International Journal of Informatics Society

17/06 Vol. 9 No.1 ISSN 1883-4566

Informatics Society

Aims and Scope

The purpose of this journal is to provide an open forum to publish high quality research papers in the areas of informatics and related fields to promote the exchange of research ideas, experiences and results.

Informatics is the systematic study of Information and the application of research methods to study Information systems and services. It deals primarily with human aspects of information, such as its qu ality and value as a resource. Informatics also referred to as Information science, studies t he structure, algorithms, behavior, and interactions of natural and a rtificial systems that store, process, access and communicate information. It also develops its own conceptual and theoretical foundations and utilizes foundations developed in other fields. The advent of computers, its ubiquity and ease to use has led to th e study of info rmatics that has computational, cognitive and social aspects, including study of the social impact of information technologies.

The characteristic of informatics' context is amalgamation of technologies. For creating an informatics product, it is necessary to integrate many technologies, such as mathematics, linguistics, engineering and other emerging new fields.

Guest Editor's Message

Haruo Hayami

Guest Editor of Twenty-Fifth Issue of International Journal of Informatics Society

We are delighted to have the twenty-fifth issue of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Tenth International Workshop on Informatics (IWIN2016), which was held at Riga, Latvia, Aug. 28-31, 2016. The workshop was the tenth event for the Informatics Society, and was intended to bring together researchers and practitioners to share and exchange their experiences, discuss challenges and present original ideas in all aspects of informatics and computer networks. In the workshop 26 papers were presented in eight technical sessions. The workshop was successfully finished with precious experiences provided to the participants. It highlighted the latest research results in the area of informatics and its applications that include networking, mobile ubiquitous systems, data analytics, business systems, education systems, design methodology, intelligent systems, groupware and social systems.

Each paper submitted IWIN2016 was reviewed in terms of technical content, scientific rigor, novelty, originality and quality of presentation by at least two reviewers. Through those reviews 15 papers were selected for publication candidates of IJIS Journal, and they were further reviewed as a Journal paper. This volume includes five papers among the accepted papers, which have been improved through the workshop discussion and the reviewers' comments.

We publish the journal in print as well as in an electronic form over the Internet. We hope that the issue would be of interest to many researchers as well as engineers and practitioners over the world.

Haruo Hayami is a full professor in the Department of Information Media at Kanagawa Institute of Technology. He researches and educates database and groupware. He has published journal or international conference papers of 50 or more, and 18 books. He received his Bachelor's and a Master's degrees in applied physics from Nagoya University, in 1970 and 1972, respectively. He received a Doctor's degree in computer engineering from Nagoya Institute of Technology in 1993. He worked at NTT laboratories from 1972 to 1998 and moved to Kanagawa Institute of Technology from NTT at 1998. He received 40th Anniversary Paper Award from IPSJ (Information Processing Society of Japan) and Marvin L. Manheim Award from WfMC (Workflow Management Coalition). He is fellows of IPSJ and WfMC.

Verifying Timed Anonymity of Security Protocols

Yoshinobu Kawabe† and Nobuhiro Ito†
†Department of Information Science, Aichi Institute of Technology, Japan
{ kawabe, n-ito }@kwb.aitech.ac.jp

Abstract - On the Internet anonymity should be provided for many services and protocols. For example, an electronic voting system should guarantee to prevent the disclosure of who voted for which candidate. Trace anonymity is an extension of the formulation of anonymity by Schneider and Sidiropoulos, and we presented an inductive method based on simulation techniques of I/O-automaton theory. In this study we discuss a proof technique for a timed version of trace anonymity. Even though communication patterns are indistinguishable, the sender's identity might be disclosed by detecting the timing of message emission. The sender's identity also might be disclosed by detecting the occurrence of timeout. To deal with such timing features, this study employs timer variables which range over non-negative real numbers. This means that we must deal with an infinite-state system for timed anonymity, but I/O-automaton theory provides various proof techniques for infinite-state systems that incorporate theorem-proving. Our proof method for timed anonymity is based on the conventional I/O-automaton theory, and the existence of an anonymous simulation is shown with two steps. In the first step, we prove the anonymity of an untimed system, and then we extend the anonymity result for the corresponding timed system incrementally.

Keywords: Timed systems, Anonymity, Verification, Formal Method, I/O-automaton

1 INTRODUCTION

We say a security protocol is anonymous if an adversary who can observe all the occurrences of events from the protocol cannot determine who is the "actor" of the events. There are many studies to describe and verify the anonymity of security protocols formally; for example, in [1] a proof technique that incorporates theorem-proving is introduced.

In this paper we discuss the anonymity of timed systems. Recently various real-time systems are used on the Internet. To establish the reliability of timed systems, there have been many studies based on formal methods that modeled and verified the correctness of timed systems [2][3]. To establish anonymity, we should deal with patterns of communication such as the number of messages or the existence/nonexistence of a message. However, even though communication patterns are designed to be indistinguishable, the sender's identity might be disclosed by detecting a timing of message emission. Also, the sender's identity might be disclosed by detecting the occurrence of a timeout. That is, the detection of timing information leads to the disclosure of who is an actor.

We describe a timed system with an I/O-automaton-based formal specification language [4]. This enables us to employ a proof method developed for the anonymity of untimed systems. By introducing a timer variable, we must deal with an infinite-state system. However, I/O-automaton theory [5][6] does not assume finiteness of the number of states or trace length, and it provides a proof technique called a simulation-based method that can handle infinite-state systems directly. In this paper we discuss how to apply a simulation-based method for proving the anonymity of timed systems.

This paper is organized as follows. In section 2, we first describe the basic notion of anonymity, and a proof technique developed in [1]. Then, we present a simple motivating example in Section 3. Also, a timed system is described in IOA language. After showing a basic idea for proving timed anonymity in Section 4, we have discussions in Section 5.

1.1 Related Work

There are studies (e.g. [7]-[9]) that analyze the anonymity of security protocols or communication systems, with dealing with the delay patterns. There are also studies on modeling the anonymity of real-time systems [10][11]. However, it has not been investigated well how to design and verify timed anonymous systems formally.

For untimed systems, a formal modeling of anonymity [12] was introduced based on the applied π-calculus [13][14], and it provides a good framework for verification. However, the notion of timed anonymity is not dealt with. As another approach, an epistemic-logic-based technique is known for modeling the anonymity of multi-agent systems [15]. It might be possible to extend the technique for timed systems by employing temporal logics [16] such as CTL or LTL. However, the anonymity of timed multi-agent systems should be proven by a human prover manually.

Based on Petri net theory [17] or timed I/O-automaton theory [18], there are many studies to verify properties of timed systems. It seems possible to formalize the notion of timed anonymity in such theories directly. However, for that case we must prove both of "the symmetry of communication patterns" and "the symmetry of timing" at the same time; this may be a burden to a protocol designer. In this study, we firstly prove the symmetry of communication patterns only. And then, the result is extended to the timed anonymity incrementally.

A communication system can be described as a state machine. Thus, for finite systems we can employ SAT/SMT-solvers [19]-[21] or model checkers [22][23]; especially, UPPAAL [22] is a well-known model checker for timed systems where we can check properties described in a temporal logic. However, we can see that timed systems are essentially infinite-state. The untimed version of I/O-automaton theory does not assume the finiteness of automata, and it provides techniques to prove the trace inclusion of infinite-state systems, which can be applied with a theorem-proving approach

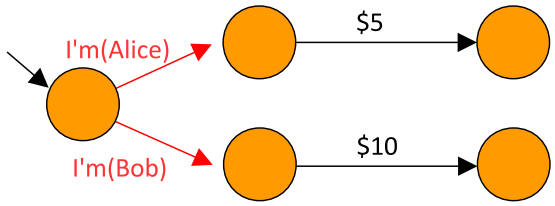


Figure 1: D1

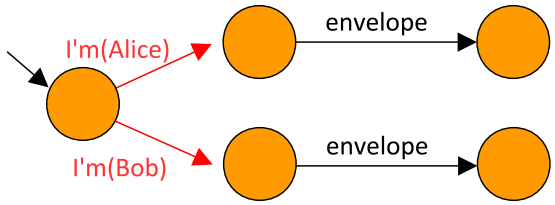


Figure 2: D2

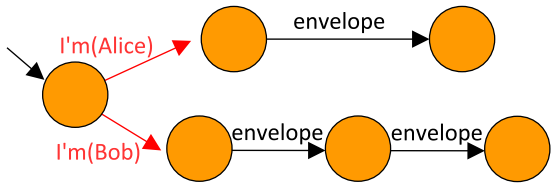


Figure 3: D3

[24]. Hence, in this study we discuss the timed anonymity property with the untimed I/O-automaton theory.

2 PRELIMINARIES

This section explains how to formalize (untimed) anonymity by I/O-automaton. We assume that readers are familiar with the basic notions and notations for I/O-automaton theory and an I/O-automaton-based formal specification language; see also Appendix A.

2.1 Basic Notion of (Untimed) Anonymity

We explain the basic notion of anonymity with the following example.

Example 1 (Donating anonymously) *There are two people, Alice and Bob, and we assume that only one of them has made an anonymous donation. Alice was going to contribute $5, while Bob was going to contribute $10.*

I/O-automaton D1 in Fig. 1 describes the above situation. Actions $5 and $10 of D1 are external actions to represent a donation. After the occurrence of I'm(Alice) or I'm(Bob), either $5 or $10 occurs. Here, I'm(Alice) and I'm(Bob) are actions that specify the donor. For convenience, we call I'm(Alice) and I'm(Bob) *actor actions*. We can see that D1 is anonymous if an adversary who observed all the occurrences of the non-actor actions cannot determine which actor action of D1 will occur.

If an adversary observed that $5 was posted, then the adversary can deduce that Alice made a donation, since action I'm(Alice) can occur in D1 only when action $5 occurs. That is, D1 is not anonymous. One reason for D1 not being anonymous is that an adversary can know how much money was posted. So, we assume that a donation was posted in an envelope. Suppose f is an operation to replace external actions $5 and $10 of D1 with a fresh external action envelope, and we define D2 as f(D1) (see Fig. 2). This operation hides information on how much money was posted. With D2, an adversary who is able to detect the occurrence of envelope cannot deduce which actor action is possible. Hence, D2 is anonymous.

If Bob is going to post $10 in two envelopes each containing $5, then we cannot establish anonymity even though all the messages are encrypted. Figure 3 shows I/O-automaton D3, which describes the above setup. Here, an adversary can determine the identity of a donor by counting the number of times that envelope occurs. Therefore, D3 is not anonymous. This example shows that a system might not be anonymous even though all the messages are encrypted.

Below, we formally discuss the correctness of communication patterns with regard to anonymity. Let X be an I/O-automaton and A be a family with the following conditions: (i) $\bigcup_{A' \in A} A' \subset ext(X)$; (ii) A' and A'' are disjoint for any distinct $A', A'' \in A$. We call A a *family of X's actor actions*, and an element of $\bigcup_{A' \in A} A'$ is called an *actor action* (on A). The occurrences of different actor actions should be indistinguishable to an adversary. That is, if an eavesdropper cannot distinguish the trace set of system X and that of X's "anonymized" version, then we can see that X is anonymous. This is formalized as follows.

Definition 1 *Let X be an I/O-automaton and A be a family of X's actor actions. We define I/O-automaton $anonym_A(X)$ as follows:*

$$
\begin{aligned}
&states(anonym_A(X)) = states(X),\\
&start(anonym_A(X)) = start(X),\\
&ext(anonym_A(X)) = ext(X),\\
&int(anonym_A(X)) = int(X) \text{ and}\\
&trans(anonym_A(X))\\
&\quad = \{(s_1, a, s_2) \mid (s_1, a, s_2) \in trans(X) \wedge a \notin \textstyle\bigcup_{A' \in A} A'\}\\
&\qquad \cup \{(s_1, a, s_2) \mid (s_1, a', s_2) \in trans(X)\\
&\qquad\qquad \wedge A' \in A \wedge a' \in A' \wedge a \in A'\}.
\end{aligned}
$$

If $traces(anonym_A(X)) = traces(X)$ holds, we say X is trace anonymous on A.

2.2 How to Prove Anonymity

This section describes a proof method for anonymity [1].

Definition 2 *Assume X is an I/O-automaton and A is a family of X's actor actions. An anonymous simulation as_A of X on A is a binary relation on $states(X)$ that satisfies the following conditions:*

1. *$as_A(s, s)$ holds for any initial state $s \in start(X)$;*
2. *For any $s_1, s_2, s_1' \in states(X)$ and $a \in sig(X)$, $as_A(s_1, s_1')$ and $s_1 \xrightarrow{a}_X s_2$ implies the following:*

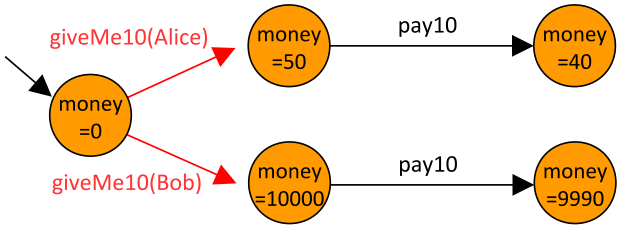


Figure 4: Automaton GMT

If $a \in A'$ holds for some $A' \in A$, for all $a' \in A'$ there is a state s_2' such that $as_A(s_2, s_2')$ and $s_1' \stackrel{a'}{\Longrightarrow}_X s_2'$; otherwise, there is a state s_2' such that $as_A(s_2, s_2')$ and $s_1' \stackrel{a}{\Longrightarrow}_X s_2'$.

I/O-automaton X is trace anonymous on A if X has some anonymous simulation as_A. It is easy to see

$$traces(anonym_A(X)) \supseteq traces(X)$$

since $anonym_A(X)$ has all the transitions of X. The other inclusion

$$traces(anonym_A(X)) \subseteq traces(X)$$

of traces can be shown since as_A is a forward simulation from I/O-automaton $anonym_A(X)$ to I/O-automaton X. A forward simulation from I/O-automaton P to I/O-automaton Q is a binary relation $r \subset states(P) \times states(Q)$ with the conditions shown in [5]; if there is a forward simulation, then we have $traces(P) \subseteq traces(Q)$ ([5], Theorem 3.10).

Proposition 1 *Let X be an I/O-automaton. If there is an anonymous simulation as_A of X on A, then*

$$traces(anonym_A(X)) = traces(X)$$

holds. □

3 ANONYMITY FOR TIMED SYSTEMS

This section discusses a formalization of timed anonymity.

3.1 Example

To explain the notion of timed anonymity, we introduce the following example.

Example 2 *There are two people, Alice and Bob. Alice has $50, while Bob has $10,000. Charlie has requested only one of them to give him $10. We do not know which person makes a payment, but one of them actually sends $10.*

I/O-automaton GMT in Fig. 4 describes the above situation. Action `giveMe10(`mem`)`, where mem is `Alice` or `Bob`, is a special action to represent the actor, and `pay10` is an action for a payment. Automaton GMT has the trace set

$$traces(\mathtt{GMT}) = \left\{ \begin{array}{l} \mathtt{giveMe10(Alice).pay10}, \\ \mathtt{giveMe10(Bob).pay10} \end{array} \right\}.$$

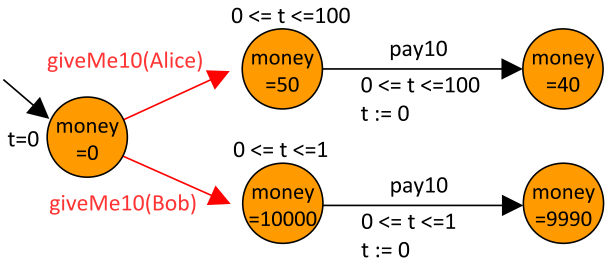


Figure 5: Timed Automaton GMTt

In this case, an adversary who observed the occurrence of action `pay10` cannot determine the preceding action. That is, both of `giveMe10(Alice)` and `giveMe10(Bob)` are possible, so the adversary never knows who made a payment. We can see that this is the same thing of the second setting in Example 1, so GMT is trace anonymous.

In Example 2, Alice possibly pays $10 even though she has only $50. In the following, we would like to consider a modified example.

Example 3 *Bob has much money ($10,000), so he can send $10 immediately. But Alice has only $50. When asked by Charlie, she thinks for a moment before sending $10.*

This is described with a timed automaton in Fig. 5. In this modeling, Alice who has only $50 might take some time up to 100 seconds before sending $10. On the other hand, Bob can make a decision within one second. From this observation, if the payment of $10 occurs after one second, then the payer is Alice. This means that even though communication patterns are indistinguishable, the sender can be identified by detecting the timing of message emission.

3.2 Describing Timed System in IOA

This section describes a timed system with an (infinite-state) untimed I/O-automaton. With examples in the previous section, we explain a basic idea of timed anonymity.

IOA [4] is a formal specification language based on I/O-automaton theory. In IOA, a state is formalized as a tuple of values. Automaton GMT in Fig. 4 is written as follows.

```
automaton GMT
  signaure
                          % AorB = { Alice, Bob }
    output giveMe10(mem: AorB)
    output pay10

  states
     money: Nat  := 0

  transitions
    % This is an actor action.
    output giveMe10(mem)
      pre money = 0
      eff if (mem = Alice) then
            money := 50;
          else
            money := 10000;
```

```
        fi

    output pay10
      pre (money = 50 \/ money = 10000)
      eff money := money - 10
```

Here, two actions `giveMe10(mem)` and `pay10` are defined in a precondition-effect style; `giveMe10(mem)` is an actor action. If we define a candidate binary relation $as_{\mathtt{GMT}}$ as:

$$as_{\mathtt{GMT}}(s, s') \iff s.\mathtt{money} = s'.\mathtt{money} \vee |s.\mathtt{money} - s'.\mathtt{money}| = 9950$$

then the binary relation satisfies the conditions to be an anonymous simulation of automaton `GMT`, where $\alpha.\beta$ represents the value of variable β at state α.

To model a timed system, in this paper we introduce special variables:

- `timer` : a timer variable for elapsing time, and
- `timerFlg` : a flag variable for activating/deactivating the timer.

Thus, we can define the following automaton `GMT2`:

```
automaton GMT2
  signaure
    output giveMe10(mem: AorB)
    output pay10
    internal timerDeactivate

   states
     money: Nat  := 0,
     timer: Real  := 0.0,
     timerFlg: Bool  := true

   transitions
     output giveMe10(mem)
       pre    ~timerFlg
           /\ money = 0
       eff if (mem = Alice) then
             money := 50;
           else
             money := 10000;
           fi
           timerFlg := true

     output pay10
       pre    ~timerFlg
           /\ (money = 50 \/ money = 10000)
       eff money := money - 10;
           timerFlg := true

     % This action is internal and it does
     % not appear in traces.
     internal timerDeactivate
       pre timerFlg
       eff timerFlg := false
```

where we can easily see that $traces(\mathtt{GMT2}) = traces(\mathtt{GMT})$ holds. The value of `timerFlg` should be `false` if either `giveMe10(mem)` or `pay10` is enabled, and `timerFlg` becomes `true` if the action is actually fired. Also, action `timerDeactivate`, which is called a time action, is enabled only if `timerFlg` is `true` and the value of `timerFlg` becomes `false`. This means that a normal action and a time action occur alternately. Note that action `timerDeactivate` does not change the value of `timer` and the time action does not appear in traces since it is internal.

By modifying `GMT2`, we can develop Fig. 5's automaton `GMTt`. Specifically, we remove `timerDeactivate` from `GMT2` and we add the following three actions.

```
output giveMe10Time
  pre    timerFlg
      /\ money = 0
  eff timerFlg := false

output pay10Time(t)
  pre    timerFlg /\ t = timer
      /\ (money = 50 \/ money = 10000)
      /\ ((money = 50)
          => (   0.0 <= timer
              /\ timer <= 100.0))
      /\ ((money = 10000)
          => (   0.0 <= timer
              /\ timer <= 1.0))
  eff timer := 0.0;
      timerFlg := false

output elapse(delta)
  pre    timerFlg /\ delta > 0.0
      /\ (money = 50 \/ money = 10000)
      /\ ((money = 50)
           => (   (   0.0 <= timer
                   /\ timer <= 100.0)
               /\ (   0.0 <= timer + delta
                   /\ timer + delta
                        <= 100.0)))
      /\ ((money = 10000)
           => (   (   0.0 <= timer
                   /\ timer <= 1.0)
               /\ (   0.0 <= timer + delta
                   /\ timer + delta
                        <= 1.0)))
  eff timer := timer + delta
```

Below we classify `GMTt`'s actions as follows:

- Normal actions (`giveMe10(mem)` and `pay10`): appear in the original automaton `GMT`; and
- Time actions (`giveMe10Time`, `pay10Time(t)` and `elapse(delta)`) : are employed for expressing timing constraints and for elapsing time.

In IOA specification `GMTt`, action `giveMe10Time` and its corresponding normal action `giveMe10(mem)` have a common condition "`money = 0`" in their precondition part. Also, actions `pay10Time(t)` and `pay10` have a common condition "`(money = 50 \/ money = 10000)`". Moreover, actions `giveMe10Time` and `pay10Time(t)` do not rewrite variable `money`. Hence, after firing `giveMe10Time` or `pay10Time(t)`, its corresponding normal action is enabled. From this observation, we can see that a one-step transition by action `pay10` in Fig. 5 is formalized with a two-

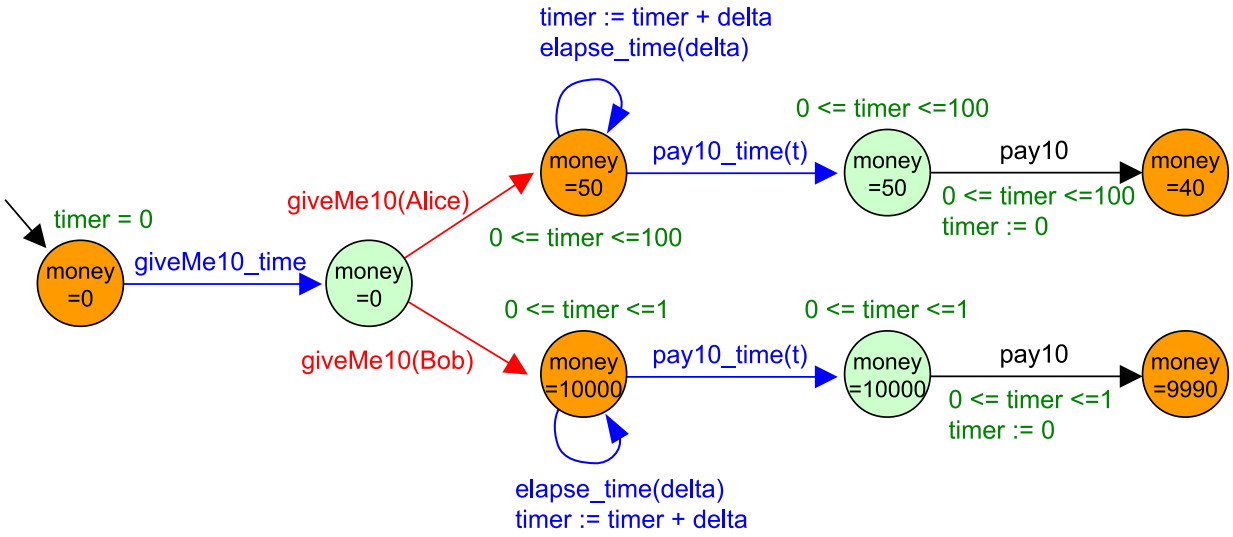


Figure 6: GMTt's Transitions (Expanded in Conventional I/O-Automaton)

step transition sequence with `pay10Time(t)` and `pay10` in IOA language; this is shown in Fig. 6[1].

`GMTt` has another time action, `elapse(delta)`, and the output action is employed for elapsing time. The precondition of `elapse(delta)` defines the timing constraint at time `timer` and at time `time` + `delta`.

4 ANALYZING ANONYMITY FOR TIMED SYSTEMS

This section analyzes the anonymity of timed systems.

4.1 Counterexample for `GMTt`'s Anonymity

Automaton `GMTt` does not have a trace

```
giveMe10Time.giveMe10(Bob).
    elapse(30).pay10Time(30).pay10
```

that represents "Bob is asked and he pays $10 after 30 seconds"; note that Bob must make a payment in one second. However, automaton `GMTt`'s corresponding anonymous system $anonym_{\{\{\texttt{Alice,Bob}\}\}}(\texttt{GMTt})$ has the above trace; that is, we cannot say:

$$traces(\texttt{GMTt}) = traces(anonym_{\{\{\texttt{Alice,Bob}\}\}}(\texttt{GMTt})).$$

This means that $anonym_{\{\{\texttt{Alice,Bob}\}\}}(\texttt{GMTt})$'s anonymity does not lead to `GMTt`'s anonymity. Therefore, `GMTt` is not anonymous.

4.2 Anonymizing `GMTt`

In this section we modify `GMTt`. Specifically, we define:

[1]In Fig. 6, there are green states and orange ones. Time actions can be enabled in orange states only, while in green states non-time actions are enabled. Also, for simplicity of depicting the automaton, a state in this figure actually represents *a collection of states*; for example, a state of "`money = 50`" represents infinitely many states where the value of `timer` ranges over $\{\, t \mid 0 \leq t \leq 100 \,\}$.

```
output pay10Time(t)
  pre    timerFlg /\ t = timer
      /\ (money = 50 \/ money = 10000)
      /\ ((money = 50)
           => (   0.0 <= timer
               /\ timer <= 1.0))
      /\ ((money = 10000)
           => (   0.0 <= timer
               /\ timer <= 1.0))
  eff timer := 0.0;
      timerFlg := false

output elapse(delta)
  pre    timerFlg /\ delta > 0.0
      /\ (money = 50 \/ money = 10000)
      /\ ((money = 50)
           => (   (   0.0 <= timer
                   /\ timer <= 1.0)
               /\ (   0.0 <= timer + delta
                   /\ timer + delta
                        <= 1.0)))
      /\ ((money = 10000)
           => (   (   0.0 <= timer
                   /\ timer <= 1.0)
               /\ (   0.0 <= timer + delta
                   /\ timer + delta
                        <= 1.0)))
  eff timer := timer + delta
```

for `pay10Time(t)` and `elapse(delta)`. That is, we replace "`timer<=100.0`" and "`timer+delta<=100.0`" in actions `pay10Time(t)` and `elapse(delta)` with conditions "`timer<=1.0`" and "`timer+delta<=1.0`", respectively. We call the resulting automaton `GMTt2`. This is to assume Alice responds in one second.

The modified automaton has an anonymous simulation:

$$\begin{aligned} as_{\texttt{GMTt2}}(s, s') \iff\ & as_{\texttt{GMT}}(s, s') \\ & \wedge s.\texttt{timer} = s'.\texttt{timer} \\ & \wedge (s.\texttt{timerFlg} \iff s'.\texttt{timerFlg}). \end{aligned}$$

This formula contains GMT's anonymous simulation relation $as_{\texttt{GMT}}$. With this binary relation, we can prove the anonymity of GMTt2 with the following steps:

1. Find an anonymous simulation for GMT;
2. Then, extend the anonymity result for GMTt2.

In the remainder of this section, we describe why the above proof is possible.

4.2.1 GMTt2's Initial State's Condition

Let $(s, t, p) \in start(\texttt{GMTt2})$ be an initial state of GMTt2, where s is a tuple that represents a state of automaton GMT, t is a value of variable timer, and p is a value of variable timerFlg. From the definition of GMTt2, we have $t = 0.0$ and $u = \texttt{true}$. Clearly, we have $as_{\texttt{GMT}}(s, s)$ implies $as_{\texttt{GMTt2}}((s, 0.0, \texttt{true}), (s, 0.0, \texttt{true}))$.

4.2.2 Step's Correspondence for Normal Actions

A normal action of GMTt2 can be enabled only if the value of variable timerFlg is false. If the action is fired, then variable timerFlg is changed to be true, but timer is not changed. Hence, we can see that for any normal action a we have:

- $(s_1, t, p) \xrightarrow{a}_{\texttt{GMTt2}} (s_1', t, p')$ and
- $as_{\texttt{GMTt2}}((s_1, t, p), (s_2, u, q))$

implies

- We have $t = u$ from the definition of $as_{\texttt{GMTt2}}$;
- We have $p = q = \texttt{false}$ and $p' = \texttt{true}$ since a is a normal action; and
- We have $as_{\texttt{GMT}}(s_1, s_2)$ and $s_1 \xrightarrow{a}_{\texttt{GMT}} s_1'$.

Thus, there exists a state s_2' of GMT such that:

- We have $s_2 \xRightarrow{a'}_{\texttt{GMT}} s_2'$ and $as_{\texttt{GMT}}(s_1', s_2')$;
- $a \in \{\texttt{giveMe10(Alice)}, \texttt{giveMe10(Bob)}\}$ implies $a' \in \{\texttt{giveMe10(Alice)}, \texttt{giveMe10(Bob)}\}$; and
- $a = \texttt{pay10}$ implies $a' = a = \texttt{pay10}$.

Therefore, for the state $(s_2', t, \texttt{true})$, we have:

- $(s_2, u, q) \equiv (s_2, t, \texttt{false}) \xRightarrow{a'}_{\texttt{GMTt2}} (s_2', t, \texttt{true})$, and
- $as_{\texttt{GMTt2}}((s_1', t, p'), (s_2', t, \texttt{true}))$.

Consequently, if binary relation $as_{\texttt{GMT}}$ is an anonymous simulation, then binary relation $as_{\texttt{GMTt2}}$ satisfies a step correspondence condition for any normal action.

4.2.3 Step's Correspondence for Time Actions

If a time action is enabled, the value of timerFlg is true. Also, variables timer and timerFlg can be changed by the time action. Hence, for any time action b, we have:

- $(s_1, t, p) \xrightarrow{b}_{\texttt{GMTt2}} (s_1', t', p')$, and
- $as_{\texttt{GMTt2}}((s_1, t, p), (s_2, u, q))$

implies

- $s_1' = s_1$, $t = u$, and $p = q = \texttt{true}$ holds;
- If b is elapse(delta) then $p' = \texttt{true}$; otherwise, $p' = \texttt{false}$; and
- $as_{\texttt{GMT}}(s_1, s_2)$ holds.

If we can prove

$$(s_2, u, q) \equiv (s_2, t, \texttt{true}) \xRightarrow{b}_{\texttt{GMTt2}} (s_2, t', p')$$

for state (s_2, t', p'), then $as_{\texttt{GMTt2}}((s_1, t', p'), (s_2, t', p'))$ holds. Hence, $as_{\texttt{GMTt2}}$ satisfies the conditions to be an anonymous simulation of GMTt2 for action b.

4.3 Further Analysis for GMTt2

We consider the transition

$$(s_1, t, p) \equiv (s_1, t, \texttt{true}) \xrightarrow{b}_{\texttt{GMTt2}} (s_1, t', p') \equiv (s_1', t', p')$$

shown in the previous section and a transition

$$(s_2, t, \texttt{true}) \xrightarrow{b}_{\texttt{GMTt2}} (s_2, u', q')$$

by time action b. From the definition of each time action, we have $u' = t'$ and $q' = p'$. Moreover, the condition sequence

$$(s_2, u, q) \equiv (s_2, t, \texttt{true}) \xRightarrow{b}_{\texttt{GMTt2}} (s_2, t', p')$$

is actually a one-step transition

$$(s_2, u, q) \equiv (s_2, t, \texttt{true}) \xrightarrow{b}_{\texttt{GMTt2}} (s_2, t', p')$$

since GMTt2 does not have any internal actions. Hence, for GMTt2, we can prove the anonymity by proving:

> For any GMTt2's time action b and any states s_1, s_2 with $as_{\texttt{GMT}}(s_1, s_2)$, if action b is enabled at state (s_1, t, p) then b is also enabled at (s_2, u, q).

Specifically, it suffices to show the following three formulae with a theorem proving tool [24], where enabled(s, a) is true if action a is enabled at state s:

```
(as(s1, s2) /\ enabled(s1, giveMe10Time))
 => enabled(s2, giveMe10Time),
(as(s1, s2) /\ enabled(s1, pay10Time(t)))
 => enabled(s2, pay10Time(t))
```

and

```
(as(s1, s2) /\ enabled(s1, elapse(delta)))
 => enabled(s2, elapse(delta)).
```

5 DISCUSSION

This section discusses the formal proof approaches for timed anonymity.

5.1 Untimed Automaton vs. Timed Automaton

In this study, we described a timed system as an infinite-state system with a conventional I/O-automaton, and we applied the proof method for anonymity [1] directly. As another approach, it seems possible to redefine the anonymity proof technique of [1][25] in timed automaton [26] or in timed I/O-automaton [18]. In this section we compare the approaches.

Timed automaton models are designed for dealing with timing features of computation; so, several constraints are introduced to verify timing properties properly. For example, an execution sequence where only time actions occur infinitely often and normal actions do not occur is regarded as unfair, and unfair execution sequences are prohibited. However, for anonymity verification we might not need such a condition; even though there is an unfair execution sequence by actor Alice in a security protocol, we can discuss the anonymity if the security protocol has its corresponding (unfair) execution sequence by actor Bob.

The untimed I/O-automaton model does not support such conditions, but it has various verification tools and proof methods, and we can use them to prove anonymity. This as an advantage of using untimed I/O-automaton theory. However, in our approach we should introduce a parameter for real-valued times; that is, we must handle infinite-state systems. We can overcome this problem since I/O-automaton theory [5][6] does not assume finiteness of the number of states or trace length, and simulation-based proof techniques are applicable to prove the trace inclusion of infinite-state systems.

We compared the both approaches, and in this study we employed a formal specification language based on conventional I/O-automaton theory. The main reason is that various verification tools are available.

5.2 On Anonymity Proof Method for Security Protocols with Stronger Adversaries

We have introduced an automaton that has variables `timer` and `timerFlg` in this study. A similar approach is employed in [27] to deal with stronger adversaries.

The technique in [1] can only deal with eavesdroppers, and in [27] an adversary model has been introduced to handle stronger adversaries, which may change the protocol's state in various ways, e.g. by sending dummy messages and by rewriting disk image of a PC. This is formalized as follows.

Definition 3 (Attacker part) *Attacker Atk of system X is quadruplet $(states(X), S_{Atk}, A_{Atk}, T_{Atk})$, where a set of attacker's states S_{Atk}, a set of attacker's actions A_{Atk} and a set of attacker's transitions T_{Atk} should satisfy:*

$$\begin{cases} sig(X) \cap A_{Atk} = \emptyset \text{ and} \\ T_{Atk} \subseteq \{((s_1, v_1), a, (s_2, v_2)) \mid s_1, s_2 \in states(X), \\ \qquad\qquad\qquad\qquad v_1, v_2 \in S_{Atk}, \\ \qquad\qquad\qquad\qquad a \in A_{Atk}\}. \end{cases}$$

With the attacker part Atk, automaton (X, Atk) is defined with:

$$\begin{aligned} &states((X, Atk)) = \{(s, v) \mid s \in states(X), v \in S_{Atk}\}, \\ &start((X, Atk)) = \{(s, v) \mid s \in start(X), v \in S_{Atk}\}, \\ &ext((X, Atk)) = ext(X) \cup A_{Atk}, \\ &int((X, Atk)) = int(X), \\ &act((X, Atk)) = act(X), \text{ and} \\ &trans((X, Atk)) \\ &\quad = \{((s_1, v), a, (s_2, v)) \mid (s_1, a, s_2) \in trans(X), v \in S_{Atk}\} \\ &\qquad \cup T_{Atk}. \end{aligned}$$

For automaton (X, Atk), a state $(s, v) \in states((X, Atk))$ has two parts. The second part v is a state of the attacker, and the protocol part X does not change the second part. We can see that, in analogy with the above adversary model, timer-related variables correspond to the attacker's state v, and time actions correspond to attacker's actions T_{Atk}.

This paper has shown a basic idea to prove the anonymity of timed systems, but we have not introduced a formal definition for timed anonymity. We believe that it is possible to introduce such a formal definition as in a similar way of [27].

6 CONCLUSION

This paper discussed a method to verify the anonymity of timed systems. By describing a timed system with an I/O-automaton-based formal specification language, a proof technique for anonymity of untimed systems can be applied to a timed system.

This paper has shown a basic idea to prove the anonymity of timed systems with a small example. As described in Section 5.2 , the formalization of timed anonymity is not complete, and it is an important future work. Also, it is another interesting future work to deal with a larger example such as Mixnet [28]. This is a well-known protocol that relaizes anonymous communication, and we can see that this is a larger and real application. In this study, we have conducted a stepwise verification of anonymity with the following three steps:

1. After describing an untimed version's specification of a communication protocol, we prove the existence of an anonymous simulation;
2. We obtain a corresponding timed version's specification by introducing some time actions;
3. We can easily extend the anonymous simulation relation for the timed system, and we prove that the extended binary relation satisfies the step's correspondence condition for all of the time actions.

We believe that the above approach is also applicable to various real applications, including Mixnet, and it is an interesting future work.

ACKNOWLEDGEMENT

This study is supported by the Grant-in-Aid for Scientific Research (C), No.26330166, of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

REFERENCES

[1] Y. Kawabe, K. Mano, H. Sakurada and Y. Tsukada, "Theorem-proving anonymity of infinite-state systems," *Inf. Proc. Lett.*, Vol. 101, No. 1, pp. 46-51 (2007).

[2] K. van Hee and N. Sidorova, "The Right Timing: Reflections on the Modeling and Analysis of Time," *PETRI NETS 2013*, LNCS 7927, Springer, pp.1-20 (2013).

[3] M. Wehrle and S. Kupferschmid, "Mcta: Heuristics and Search for Timed Systems," *FORMAT 2012*, LNCS 7595, Springer, pp.252-266 (2012).

[4] A. Bogdanov, "Formal verification of simulations between I/O-automata," Master's thesis, MIT (2000).

[5] N. A. Lynch and F. Vaandrager, "Forward and backward simulations — part I: Untimed systems," *Inform. and Comput.*, Vol. 121, No. 2, pp. 214-233 (1995).

[6] N. A. Lynch, *Distributed algorithms*, Morgan Kaufmann Publishers, (1996).

[7] R. E. Newman, V. R. Nalla and I. S. Moskowitz, "Anonymity and Covert Channels in Simple Timed Mix-firewalls," In *PET 2004*, LNCS 3424, Springer-Verlag, pp. 1-16 (2004).

[8] V. C. Perta, M. V. Barbera and A. Mei, "Exploiting Delay Patterns for User IPs Identification in Cellular Networks," In *PETS 2014*, LNCS 8555, Springer-Verlag, pp. 224-243 (2014).

[9] P. Palmieri and J. Pouwelse, "Paying the Guard: an Entry-Guard-based Payment System for Tor," In *FC 2015*, LNCS 8975, Springer-Verlag, pp. 437-444 (2015).

[10] A. Serjantov and R. E. Newman, "On the Anonymity of Timed Pool Mixes," In *Proceedings of the Workshop on Privacy and Anonymity Issues in Networked and Distributed Systems*, Kluwer, Athens, Greece, pp. 427-434 (2003).

[11] G. Tóth and Z. Hornák, "Measuring Anonymity in a Non-adaptive, Real-time System," In *PET 2004*, LNCS 3424, Springer-Verlag, pp. 226-241 (2004).

[12] S. Kremer and M. Ryan. "Analysis of an electronic voting protocol in the applied Pi calculus," In *ESOP '05*, LNCS 3444, Springer-Verlag, pp. 186-200 (2005).

[13] M. Abadi and C. Fournet. "Mobile values, new names, and secure communication," In *POPL '01*, ACM Press, pp. 104-115 (2001).

[14] M. Abadi and C. Fournet. "Private authentication," *TCS*, Vol. 322, pp. 427-476 (2004).

[15] J. Y. Halpern and K. R. O'Neill. "Anonymity and information hiding in multiagent systems," *J. of Computer Security*, Vol. 13, No. 3, pp. 483-514 (2005).

[16] E. M. Clarke Jr., O. Grumberg and D. Peled, *Model Checking*, MIT Press (1999).

[17] W. Reisig, *Understanding Petri Nets*, Springer (2013).

[18] D. Kaynar, N. Lynch, R. Segala and F. Vaandrager, "The Theory of Timed I/O Automata," *Synthesis Lectures on Computer Science*, Morgan Claypool Publishers (2010).

[19] R. M. Podorozhny, S. Khurshid, D. E. Perry and X. Zhang, "Verification of multi-agent negotiations using the alloy analyzer," In *IFM '07*, LNCS 693, Springer-Verlag, pp.501-517 (2007).

[20] Alloy, `http://alloy.mit.edu/alloy/`, Retrieved (2016).

[21] The Yices SMT Solver, `http://yices.csl.sri.com/`, Retrieved (2016).

[22] UPPAAL, `http://www.uppaal.org/`, Retrieved (2016).

[23] NuSMV home page, `http://nusmv.fbk.eu/`, Retrieved (2016).

[24] J. F. Soegaard-Andersen, S. J. Garland, J. V. Guttag, N. A. Lynch and A. Pogosyants. "Computer-assisted simulation proofs," In *CAV '93*, LNCS 697, Springer-Verlag, pp. 305-319 (1993).

[25] I. Hasuo, Y. Kawabe and H. Sakurada, "Probabilistic anonymity via coalgebraic simulations," *TCS*, Vol. 411, No. 22-24, pp. 2239-2259 (2010).

[26] R. Alur and D. Dill, "A theory of timed automata," *TCS*, Vol. 126, pp. 183-235 (1994).

[27] Y. Kawabe and H. Sakurada, "An adversary model for simulation-based anonymity proof," *IEICE Trans.*, Vol. E91-A, No. 4, pp. 1112-1120 (2008).

[28] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *CACM*, Vol. 24, No. 2, pp. 84-90 (1981).

A I/O-AUTOMATON AND IOA LANGUAGE

I/O-automaton theory is a formal system to describe and analyze distributed algorithms. This section provides a brief overview of I/O-automaton theory and IOA formal specification language. A formal definition of an I/O-automaton is given in Section A.1.

A.1 I/O-Automaton Theory

An I/O-automaton X is a tuple

$$(sig(X), states(X), start(X), trans(X))$$

where $sig(X)$ is a set of actions, $states(X)$ is a set of states, $start(X) \subset states(X)$ is a set of initial states, and

$$trans(X) \subset states(X) \times sig(X) \times states(X)$$

is a set of transitions. There are three sorts of actions — input, output and internal. We use $in(X)$, $out(X)$ and $int(X)$ for the sets of input, output and internal actions, respectively. We assume that $in(X)$, $out(X)$ and $int(X)$ are disjoint. We define $ext(X)$ as the union of $out(X)$ and $in(X)$, and an element of $ext(X)$ is called an external action. For simplicity, this paper only deals with I/O-automaton X satisfying $in(X) = \emptyset$; that is, we assume that $ext(X) = out(X)$.

Transition $(s, a, s') \in trans(X)$ is written as $s \xrightarrow{a}_X s'$; we also write $s \rightarrow_X s'$ if a is internal. We define the relation $\twoheadrightarrow_X$ as the reflexive transitive closure of $\rightarrow_X$. For any $a \in sig(X)$ and $s, s' \in states(X)$, we write $s \stackrel{a}{\Longrightarrow}_X s'$ for $s \twoheadrightarrow_X s_1 \xrightarrow{a}_X s_2 \twoheadrightarrow_X s'$ with some $s_1, s_2 \in states(X)$ if a is external, or for $s \twoheadrightarrow_X s'$ if a is internal.

```
automaton channel(i, j: ID)
  signature
    input  send(const i, const j, r:Req)
    output recv(const i, const j, r:Req)

  states
    queue: Seq[MES] so that queue = empty

  transitions
    input send(i, j, r)
      eff queue := packet(i, j, r) -| queue
    output recv(i, j, r)
      pre queue ~= empty
          /\ last(queue) = packet(i, j, r)
      eff queue := init(queue)
```

Note: We employ several pre-defined datatypes or operators, such as datatype `Seq` for a sequence, operators `:=` (substitution), `-|` (appending an element to the head of a sequence), `|-` (appending an element to the tail), `last` (a sequence's tail element) and `init` (deleting the tail element).

Figure 7: Formalizing a communication channel

For any initial state $s_0 \in start(X)$ and transition sequence $\alpha \equiv s_0 \xrightarrow{a_1}_X s_1 \xrightarrow{a_2}_X \cdots \xrightarrow{a_n}_X s_n$, the sub-sequence of $a_1 a_2 \cdots a_n$ that consists of all the external actions is called the *trace* of α. We write $traces(X)$ for the entire set of X's traces. In I/O-automaton theory, various properties of a distributed system can be defined as conditions of a trace set (see Sec. 8.5.3 of [6]), and a proof technique with *forward simulations* is available to show the inclusion of trace sets of two I/O-automata.

Definition 4 *A forward simulation $f \subset states(X) \times states(Y)$ from automaton X to automaton Y is a binary relation with the following:*

1. *For any initial state a of X, there is some initial state b of Y and $f(a, b)$ holds;*
2. *For any reachable states a_1, a_2 of X, any reachable state b_1 of Y and any action π of X, if $f(a_1, b_1)$ and $a_1 \xrightarrow{\pi}_X a_2$ hold then there is a state b_2 that satisfies $f(a_2, b_2)$ and $b_1 \stackrel{\beta}{\Longrightarrow}_Y b_2$ with $\beta = trace(a_1 \xrightarrow{\pi}_X a_2)$.*

Proposition 2 (Th. 3.10 of [5]) *$traces(X) \subseteq traces(Y)$ holds if there is a forward simulation from X to Y.* □

A.2 IOA Specification Language

In this paper we specify systems in IOA language [4], which is a formal specification language based on I/O-automaton theory. Figure 7 is an IOA example that models a communication channel from agent `i` to agent `j`. IOA specification `channel(i, j)` consists of the following portions:

1. `signature`: declares actions and their sorts;
2. `states`: declares variables. In this example, a variable `queue` for a message sequence is declared with the initial value `empty` and a sort `Seq[MES]`;
3. `transitions`: defines a body for each action, and the body is described in a precondition-effect style. There are two actions in the above example:

   Input action `send(i, j, r)`: attaches message `packet(i, j, r)` to the head of `queue`. Input actions do not have preconditions;

   Output action `recv(i, j, r)`: is executable when `queue` is not `empty` and the last element of `queue` is `packet(i, j, r)`. The effect is to remove `packet(i, j, r)` from `queue`.

A state is formalized as a tuple of values for which a sort is declared in the `states`-part of IOA specification. For example, state set $states$(`channel(i, j)`) of I/O-automaton `channel(i, j)` is

$$\{\, (queue) \mid queue \text{ is a value of sort } \texttt{Seq[MES]} \,\}.$$

(Received October 17, 2016)

Yoshinobu Kawabe received his B.E., M.E. and D.E. degrees in information engineering from Nagoya Institute of Technology in 1995, 1997 and 2003, respectively. He joined NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation in 1997. In 2002, he was a visiting research scientist at MIT Laboratory for Computer Science. Since 2008, he has been with Aichi Institute of Technology, where he is a professor in the Department of Information Science. His research interests include term rewriting systems, process algebras, network programming languages, formal methods and security verification. He is a member of ACM, JSSST, IPSJ and IEICE.

Nobuhiro Ito was born in Aichi, Japan, in 1970. He received the B.E., M.E. and D.E. degrees from Nagoya Institute of Technology, in 1994, 1996 and 1999, respectively. He has been a professor in the department of Information Science, Aichi Institute of Technology, Japan, since 2015. His current research interests are protocols and algorithms for teamwork and contributions for real world by disaster simulations.

Development and Evaluation of a Near-Miss Map System utilizing Driver's Emotions

Yoshia Saito*

*Faculty of Software and Information Science, Iwate Prefectural University, Japan
y-saito@iwate-pu.ac.jp

Abstract – To realize a safe car society, it is necessary to support drivers not only in mechanical aspects but also human aspects. We propose a system for automatically generating a near-miss map utilizing driver's emotion. Existing near-miss map systems only use conditions of the car such as heavy braking to detect unsafe locations. However, the accuracy of the near-miss map is not so high because there are many false negative and false positive errors. To decrease the false positive errors and false negative errors for generating a high-accuracy near-miss map, we introduce human emotions into the detection of unsafe locations and realize a high-accuracy near-miss map system. In this paper, we report the design and implementation of our prototype system and evaluation results of the prototype system in terms of false negative and false positive errors.

Keywords: Emotion, Near-Miss Map

1 INTRODUCTION

There are 77 million cars in Japan [1] and 1.1 billion cars in the world [2]. The cars are useful and essential for people to live a daily life. On the other hand, there were 540 thousand traffic accidents in Japan. 4100 people died as the victim of the traffic accidents and 670 people were injured [3]. The cars also pose a significant danger to people and we must decrease the risk. As mechanical approaches, recent cars equip driving assistance technologies such as ESC (Electronic Stability Control) and ABS (Antilock Brake System). In addition, more advance technologies which include pre-crash safety are coming into practical use in recent years such as ADAS (Advanced Driving Assistant System) and DSSS (Driving Safety Support Systems). With assistance of these technologies, the traffic accidents decreased from 950 thousand in 2003, when there were the highest number of the accidents, to 540 thousand in 2015. The decrease ratio, however, draws a shallow slope in recent years and it needs other solutions than just the mechanical ones.

It is said that a reason caused majority of the traffic accidents is human error. To decrease the traffic accidents, we need to focus on human and support drivers introducing more human approaches. As the human approaches, it is recognized that the driver's emotion affect the driving performance. The typical example is "Road Rage" [4]. The road rage is a term used to denote aggressive behaviors by drivers when they are cut into their line, overtaken by the others and got angry by other reasons. It causes fatal and injury accidents. In this way, the human emotions are important for the driving safety. Many researchers study influence of the driver's emotion and apply the driver's emotion to the driving support system [11]-[16].

In this research, we develop a system which creates a high-accurate near-miss map utilizing the driver's emotions. The near-miss map is a map which shows unsafe locations gathering information. The information gathering of the unsafe location in the existed system is realized by 2 different ways. The first is a manually operated way which gathers from human (e.g. questionnaire survey) the second is an automatically operated way which gathers from sensors (e.g. heavy braking detection). The manually operated way requires a great deal of time and the automatically operated way has an accuracy problem of unsafe location detection. We introduce driver's emotion to the automatically operated way in order to improve the accuracy of unsafe location detection. In our proposed system, a smartphone is used to recognize driver's surprise and fear emotions by facial expression and detect an unsafe location using the driver's emotions in conjunction with sensed car conditions. The information of unsafe location is sent to the server via the Internet. The other drivers which come by near the location can receive warning to be aware of the risk of traffic accidents.

The paper is organized as follows. In the next section, we describe human emotion researched in psychology and applied studies of the human emotions in the automobile field. Section 3 shows existed near-miss map system and their problems as a preliminary study. We propose a near-miss map system utilizing driver's emotion in section 4 and implement a prototype system in section 5. Section 6 evaluates the prototype system and reports the experiment results. Section 7 gives some conclusions and our future work.

2 RELATED WORK

As related work, we explain human emotions at first. After that, we describe existed researches which utilize human emotions in automobile field and the effectiveness to apply human emotions to driving assistance systems.

2.1 Human Emotions

Human emotions have been researched in the field of psychology. Ekman defined six basic emotions by researching facial expressions of human [5]. The basic emotions are "happiness", "surprise", "fear", "sadness", "anger", and "disgust". Seven emotions added "neutral" to the six basic

emotions are frequently used in the field of facial expression recognition with image processing technologies [6]-[8].

In some researches applied human emotions to information systems, the emotion model defined by Plutchik is used. Plutchik defines dimensions which include eight basic emotions “joy”, “trust”, “fear”, “surprise”, “sadness”, “disgust”, “anger”, and “anticipation” [9]. These emotions presents a circumplex model. In this model, similar emotions are placed on the neighborhood and opposite emotions are placed on the opposite side. Plutchik also defines eight combinations of two basic emotions “love”, “submission”, “awe”, “disapproval”, “remorse”, “contempt”, “aggressiveness”, and “optimism”.

Parrot classify human emotions and define them as hierarchical tree structure [10]. The primary emotions are “love”, “joy”, “surprise”, “anger”, “sadness”, and “fear”. Other a lot of emotions more than one hundred are also defined as secondary emotions and tertiary emotions.

In the above emotion models, there are several common basic emotions such as “surprise”, “anger”, “sadness”, and “fear”. From these common basic emotions, we focus on the “surprise” and “fear” emotions in our research because these emotions can be appeared on driver’s facial expression in near-miss situation.

2.2 Emotions in Automobile Field

There are a lot of researches to detect emotions of drivers using sensor devices and grasp conditions of the drivers. Jones researched a method to detect driver’s emotion using his/her speech and developed a system for emotion recognition in a car [11]. Riener proposed a method to detect driver’s emotions using his/her heart rate variability [12]. Haak tried to detect driver’s emotion by analyzing his/her brain signals [13]. Moreover, Anzengruber developed a system which can detect driver’s emotions using his/her face surface temperature and evaluated the system on a driving simulator [14]. In this manner, we can detect driver’s emotions utilizing information collected by various sensor devices.

There are also applied researches utilizing the detected driver’s emotions. Jeon studied effects of driver’s angry and fear emotion on his/her driving performance [15]. He found angry drivers made more mistakes than fear drivers and fear drivers had heavier workload than angry drivers. Cilfford described traffic accidents cloud be decreased if a navigation system changes emotional expression of voice guidance [16]. The energetic voice of the navigation system was effective when driver was happy and subdued voice was effective for upset drivers in the experiments. In this manner, these researches indicate the driver’s emotions affect his/her driving performance significantly. To grasp not only car conditions but also driver’s conditions using his/her emotions is necessary and utilization of the driver’s emotions will be effective to develop a system which creates a high-accurate near-miss map.

3 PRELIMINARY STUDY

In this section, we describe several existed near-miss maps and their map generating methods. Then, we clarify a problem of the generating methods.

3.1 Existed Near-Miss Maps

Many local governments and organizations create and offer near-miss maps for citizens and communities. To create and offer the near-miss maps, it is necessary to gather information of unsafe locations and warn the information to people. In this research, we focus on information gathering of unsafe locations.

There are two methods for the information gathering to create near-miss maps. The first method is manual information gathering. It gathers information manually by means of questionnaire surveys and interview researches (called "manual information gathering" hereafter). Most near-miss maps [17]-[20] are created by the manual information gathering. This method can gather high-accurate information. However, a great deal of time and effort is required and it is difficult to gather information in real time. The second method is automatic information gathering. It gathers information automatically by means of analyzing sensor data (called "automatic information gathering" hereafter). Typical example is the SAFETY MAP of Honda Internavi [21]. The automatic information gathering detects a heavy braking and bad road using acceleration sensors and so on. This method does not need time and effort. However, accuracy of the map is not so high because it uses only car conditions although it does not always show the driver feels a sense of danger at that time.

3.2 Manual vs. Automatic Information Gathering

To determine whether there is difference between manual information gathering and automatic information gathering, we compare the existed near-miss maps. In this preliminary study, we compared a near-miss map at the station of Muikamachi, Minamiuonuma city of Niigata in Japan [20] with a map of the SAFETY MAP [21] at the same location. The former map is created by manual information gathering (called "manual near-miss map" hereafter). The latter map is created by Honda Internavi which detects -0.2G acceleration and estimates a heavy breaking (called "automatic near-miss map" hereafter).

By comparing these two near-miss maps, we found the manual near-miss map said unsafe locations but not unsafe locations in automatic near-miss map. This problem is called “false negative error”. The term, false negative error is a result that indicates a given condition is not fulfilled but the actual condition is fulfilled. There are many false negative errors in the automatic near-miss map. Thus, these false negative errors mean the driver felt a sense of danger but did not brake hard. Especially, these errors were found around locations where city areas prohibited speeding.

On the other hands, we found the automatic near-miss map said unsafe locations but not unsafe location in manual near-miss map. This problem is called “false positive error”. The term, false positive error is a result that indicates a given condition is fulfilled but the actual condition is not fulfilled. There are many false positive errors in the automatic near-miss map. Thus, these false positive errors mean the driver did not feel a sense of danger but braked hard.

Table 1: False negative and false positive error

		Condition (Correct Answer)	
		Positive	Negative
Test Outcome	Positive	True Positive	**False Positive**
	Negative	**False Negative**	True Negative

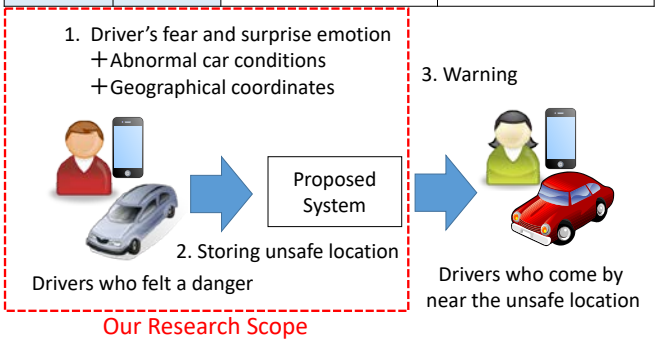


Figure 1: The model of our proposed system

Especially, these errors were found around locations where long straight road with a good view allowed speeding.

Table 1 shows supplementary explanation for the false negative and false positive errors. The false negative and false positive errors must be solved in order to increase accurate of the automatic near-miss map. As mentioned above, car conditions such as heavy brakes are not enough to detect actual unsafe locations where the driver feels a sense of danger. If we apply the driver's emotions to the automatic information gathering, more high-accurate near-miss map can be created without time and effort.

4 PROPOSED SYSTEM

We propose a system which creates a high-accurate near-miss map utilizing the driver's emotion.in order to realize a safe car society. Figure 1 shows the model of our proposed system. The proposed system utilize driver's fear and surprise emotions and abnormal car conditions to gather information of unsafe locations. By using not only car conditions but also driver's conditions, it improves accuracy of the automatic information gathering for unsafe locations. The proposed system uses the information of unsafe locations and warns of a danger to drivers who come by near the locations in order to encourage them to drive carefully. In this research, our research scope is the mechanism of information gathering for unsafe locations in a high accuracy.

The proposed system use general-purpose smartphones and does not need any dedicated devices so that more people can use the system. Therefore, we use only general sensors such as a camera and an acceleration sensor equipped in the general-purpose smartphones. Figure 2 shows the architecture of the proposed system. The procedure of proposed system is as follows.

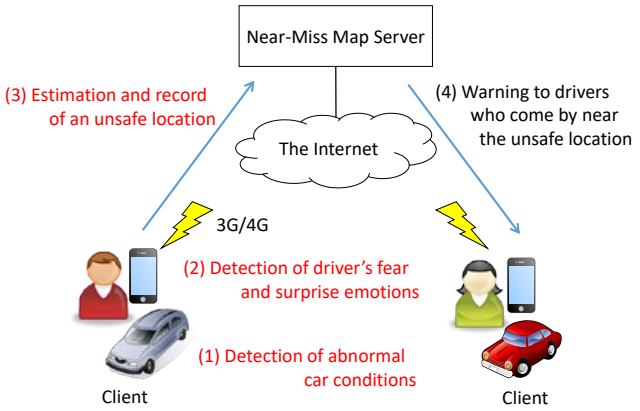


Figure 2: The architecture of the proposed system

(1) Detection of abnormal car conditions
(2) Detection of driver's fear and surprise emotions
(3) Estimation and record of an unsafe location
(4) Warning to drivers who come by near the unsafe location

In next section, we implement a prototype system using the above procedure.

5 IMPLEMENTATION

In this section, we describe implementation of functions (1) detection of abnormal car conditions, (2) detection of driver's fear and surprise emotions, (3) estimation and record of an unsafe location, and (4) warning to drivers who come by near the unsafe location. The function (4) is out of our research scope but we just implement it.

5.1 Detection of Abnormal Car Conditions

We detect heavy brakes to detect abnormal car conditions. An acceleration sensor equipped in a smartphone is used for the function of heavy brake detection. The function detects a certain measure of acceleration. In the prototype system, a smartphone is mounted on dashboard of a car and senses variation of the acceleration. When the variation value exceeded by -0.25G which is defined by reference to the other existed system for automatic near-miss map, the function recognizes it as a heavy brake.

5.2 Detection of Driver's Fear and Surprise Emotions

For the detection of driver's fear and surprise emotions, we use clmtrackr [22],[23] which is a library of facial expression recognition. This library presents each emotion as a probability value from 0.0 to 1.0 by the facial expression. It uses a camera of the mounted smartphone and gets driver's facial expression images in real time. From the image, the library calculate the probability value for each emotion as shown in Fig. 3.

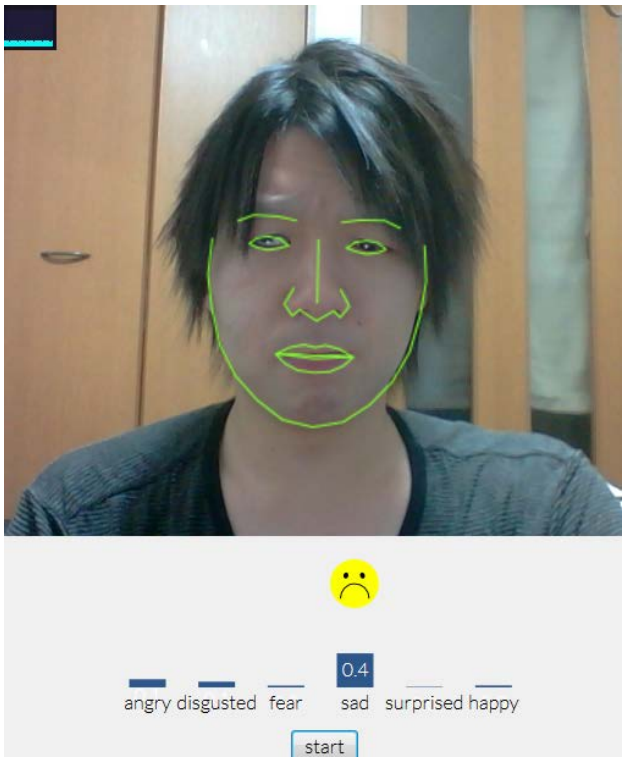


Figure 3: The detection of each emotion by facial expression image in real time

5.3 Estimation and Record of an Unsafe Location

We conducted a preliminary experiment to assess a reference probability value of fear or surprise emotion for estimation of unsafe locations and set it to 0.3 with heavy brakes. However, it could not detect unsafe locations when the heavy brake was not detected in a city areas prohibited speeding. In such a case, we use a high probability value of fear or surprise emotions with no heavy brake to estimate unsafe locations in low-speed areas. The high probability value for fear and surprise emotions is used for 0.6 which is twice as much as 0.3 with heavy brakes. After the detection, information of the unsafe location is sent to a server on the Internet via 3G/4G wireless network. The client communicates with the server using WebSocket and the information includes geographical coordinates of an unsafe location, probability values of the driver's emotions and acceleration data.

5.4 Warning to Drivers

Figure 4 shows a user interface of the prototype system on a smartphone in order to detect unsafe locations and give a warning to the driver. The user interface presents a driver's current location on the map by using GPS equipped in the smartphone. The current location is updated in real-time. Unsafe locations are also displayed on the map by getting unsafe location information from the server. The driver's facial expression image is taken from a camera equipped in

Figure 4: The user interface of the prototype system

the smartphone and driver's emotions are detected in real time. The acceleration value is displayed for heavy brakes. The client communicates with the server every 5 seconds. It sends current location information and receives unsafe location information. When the driver comes close to an unsafe location, the client give a warning to the driver with alarmed sound and text. After a fixed time, the alarm is stopped.

6 EVALUATION

We evaluate the prototype system in terms of false-negative and false-positive errors. At first, we explain the experimental methodology. Then, we show the results and discuss about accuracy of the near-miss map.

6.1 Experimental Methodology

In the experiment, we create a manual near-miss map created by a questionnaire and an automatic near-miss map created by the prototype system. After that, we compare them if the automatic near-miss map achieves low false-negative and false-positive errors.

Figure 5 shows the driving route in the experiment. The route includes residential sections around our university, straight road where the driver can put on speed, sloping, and various types of roads. It is 17 km long and takes time for approximately 30 minutes to go around the route.

At first, we created a manual near-miss map by a questionnaire. The subjects are 30 students, who have driver license, of our university. Before conducting the questionnaire, we developed a system for the questionnaire survey which presents a driving video on the route and current car location on a map as shown in Fig. 6. The subjects used the system and answered unsafe locations and their reasons. Figure 7 shows the format of the questionnaire. In this time, there were no dangerous scene in the video and we asked subjects to remember their experience when passed the locations. We defined the unsafe locations where the subjects of 10 percent

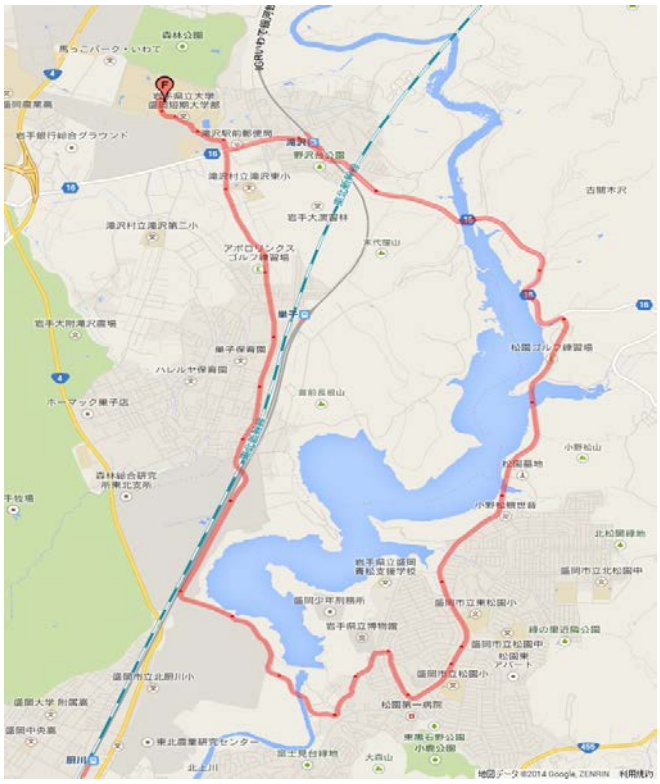


Figure 5: The driving route in the experiment

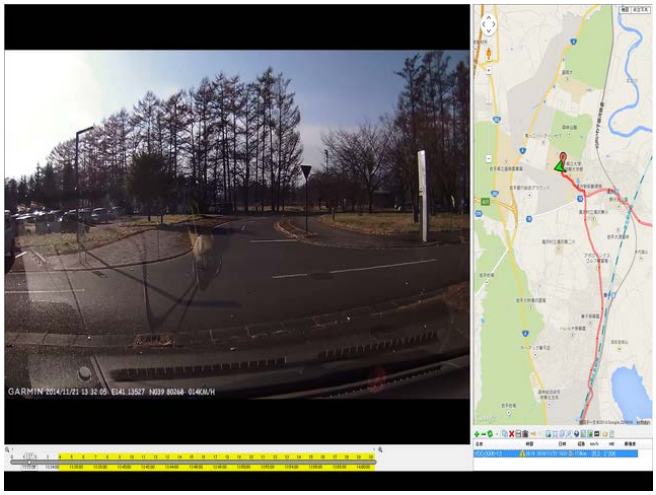


Figure 6: A system for the questionnaire

(3 subjects) answered it was unsafe in the questionnaire.

Secondly, we conducted an experiment on the road with a real car which mounted a smartphone on the dashboard for the prototype system. In this experiment, the subjects were 20 students, who have driver license, of our university. We suppose the proposed system is utilize driver's emotion. However, for the safety of the subjects, they rode in the front passenger seat of the car and the author drove the car. We also record the driving video for analytical use. Figure 8 shows the experimental environment in the car.

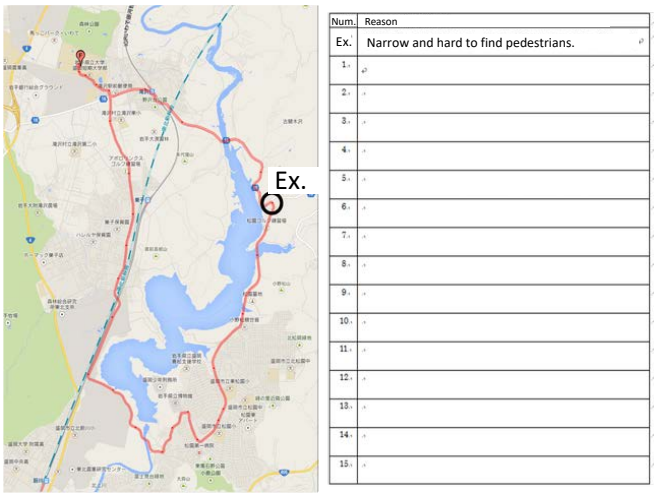


Figure 7: The format of the questionnaire

Figure 8: The experimental environment in the car

6.2 Results and Discussions

Figure 9 shows the result of the manual near-miss map created by the questionnaire. The pins show the unsafe locations defined by the questionnaire. In this evaluation, we use this map as a correct data which presents unsafe locations.

Figure 10 shows the locations where the heavy brake was detected on the road with a real car. We can see there are many false positive and false negative errors comparing with the manual near-miss map. It is difficult to create a near-miss map only using heavy brake detection.

Figure 11 shows the location where the heavy brake was detected with greater than 0.3 probability value of fear/surprise emotions. This result can be reduce false positive errors. However, there are false negative errors yet comparing with the manual near-miss map.

Figure 12 shows the automatic near-miss map created by the prototype system. This is a map which are added locations greater than 0.6 probability value of fear/surprise

Figure 9: The manual near-miss map created by the questionnaire

Figure 10: The locations of heavy brakes

Figure 11: Heavy brake with greater than 0.3 probability value of fear/surprise emotions

emotions to Fig. 11.

Figure 13 shows the comparison result of these two maps. The circles show the locations of false-positive errors and the triangles show the locations of false-negative errors. Table 2 shows the result of false-positive and false-negative error rate.

The locations of false-positive errors were three points. The one point was detected by heavy brakes and 0.3 probability value of fear/surprise emotions. The other two points were detected by 0.6 probability value of fear/surprise emotions. The rate of the false-positive errors was 14 % which was on 3 errors of 21 detected locations and the prototype system achieved low false-positive error rate.

The locations of false-negative errors were eight points. The rate of the false-negative errors was 36 % which was on 8 errors of 22 actual unsafe locations. The prototype system should be improved in terms of the unsafe locations. Possible reasons for the false-negative errors are low accuracy of the emotion detection. Since the emotion detection of facial expression is difficult and has a limit of accuracy, the prototype system could not detected the subject's fear/surprise emotions. One of the solutions is to use together with other methods for emotion detection such as utilization of driver's heart rate, body temperature and so on.

Figure 12: The automatic near-miss map created by the prototype system

Table 2: Percentage of false-positive and false-negative error in the evaluation

False-positive	14 %
False-negative	36 %

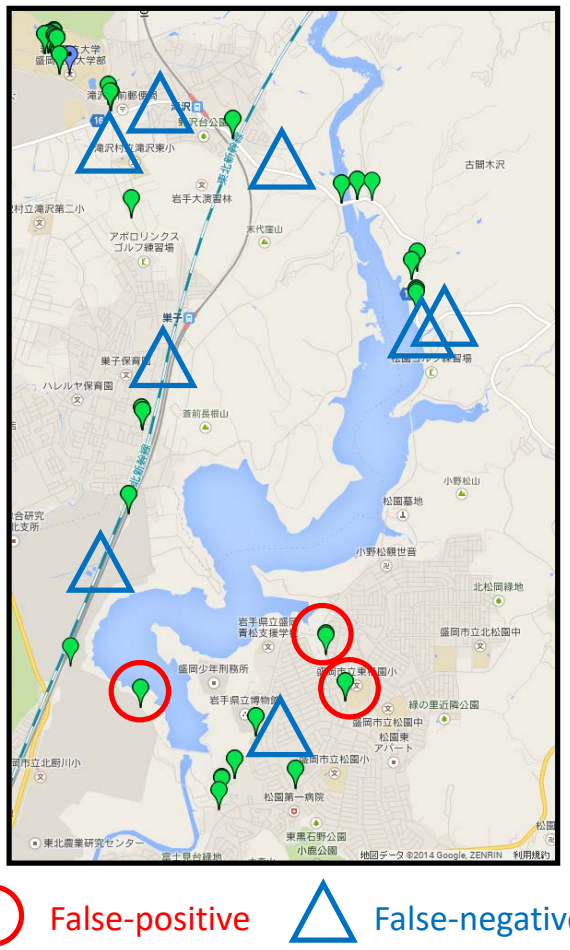


Figure 13: The result compared between the manual and automatic near-miss maps

7 CONCLUSION

In this paper, we proposed a high-accurate near-miss map system utilizing driver's emotions and developed the prototype system. From the evaluation experiment, we found our proposed system could achieve false-positive error rate although it required to improve false-negative error rate. For the future work, we will introduce other methods for emotion detection to reduce false-negative error rate.

REFERENCES

[1] Automobile Inspection & Registration Information Association, https://www.airia.or.jp/publish/statistics/ub83el00000000wo-att/01.pdf

[2] Japan Automobile Manufacturers Association., http://www.jama.or.jp/world/world/index.html

[3] National Police Agency in Japan, http://www.e-stat.go.jp/SG1/estat/List.do?lid=000001150496

[4] T. Galovski and E. Blanchard, “Road rage: a domain for psychological intervention?,” Aggression Violent Behavior, Vol. 9, Issue 2, pp. 105-127 (2004).

[5] P. Ekman, “Facial Expression and Emotion,” American Psychologist, pp. 384-392 (1993).

[6] M. J. Lyons, J. Budynek, and S. Akamatsu: “Automat-ic Classification of Single Facial Images,” IEEE Transactions on Pattern Analysis and Machine Intel-ligence, Vol. 21, No. 12, pp. 1357-1362 (1999).

[7] D. Datcu and L. Rothkrantz, “Facial expression recognition in still pictures and videos using active appearance models: a comparison approach,” Proceedings of the 2007 international conference on Computer systems and technologies (CompSysTech), No. 112, pp. 1-6 (2007).

[8] E. Costantini, F. Pianesi and M. Prete, “Recognising emotions in human and synthetic faces: the role of the upper and lower parts of the face,” Proceedings of the 10th international conference on Intelligent user interfaces (IUI), pp. 20-27 (2005).

[9] R. Plutchik, “The nature of emotions, American Scientist,” pp.344–350 (2001).

[10] W. Parrott, “Emotions in social psychology,” Psychology Press (2001).

[11] C. M. Jones and I. Jonsson, “Automatic recognition of affective cues in the affective cues in the speech of

car drivers to allow appropriate responses," OZ-CHI'2005, pp. 1-10 (2005).
[12] A. Riener, A. Ferscha and M. Aly, "Heart on the road: HRV analysis for monitoring a driver's affective state," AutomotiveUI'09, pp. 99-106 (2009).
[13] P. van den Haak, R. van Lon, J. van der Meer and L. Rothkrantz, "Stress assessment of car-drivers using EEG-analysis," CompSysTech'10, pp. 473-477 (2010).
[14] B. Anzengruber and A. Riener, " "FaceLight" – Potentials and Drawbacks of Thermal Imaging to Infer Driver Stress," AutomotiveUI'12, pp. 210-216 (2012).
[15] M. Jeon, J. Yim and B. N. Walker, "An Angry Driver Is Not the Same As a Fearful Driver: Effects of Specific Negative Emotions on Risk Perception, Driving Performance, and Workload," AutomotiveUI'11, pp. 137-140 (2011).
[16] C. Nass, Ing-Marie Jonsson, Helen Harris, Ben Reaves, Jack Endo, Scott Brave and Leila Takayama, "Improving Automotive Safety by Pairing Driver Emotion and Car Voice Emotion," CHI'05, pp. 1973-1976 (2005).
[17] A near-miss map for elementary schools of Saitama city in Japan, http://www.city.saitama.jp/kita/001/001/003/p020992.html
[18] A near-miss map of Sodegaura city in Japan, https://www.city.sodegaura.lg.jp/soshiki/doboku-kensetsu/hiyarihattotizu.html
[19] A near-miss map of Shizuoka city in Japan, http://dataset.city.shizuoka.jp/dataset/anhoko
[20] A near-miss map of Minamiuonuma city in Japan, http://www.adclub.jp/common/hiyarihatto/minamiuonuma_niigata.html
[21] Honda SAFETY MAP, http://safetymap.jp/
[22] clmtrackr, https://github.com/auduno/clmtrackr
[23] J. M. Saragih, Simon Lucey and Jeffrey F. Cohn, "Deformable Model Fitting by Regularized Landmark Mean-Shift," International Journal of Computer Vision, Volume 91, Issue 2, pp. 200-215 (2011).

(Received October 18, 2016)

Yoshia Saito received his Ph.D. degree from Shizuoka University, Japan, in 2006. He had been an expert researcher of National Institute of Information and Communications Technology (NICT) from 2004 to 2007, Yokosuka, Japan. He was a lecturer from 2007 to 2011 at Iwate Prefectural University and he is currently an associate professor at the University. His research interests include computer networks and Internet broadcasting. He is a member of IPSJ, IEEE, and ACM.

GPU Acceleration of EDA-based Algorithms to Learn Bayesian Networks

Takashi Mori[†*], Yuma Yamanaka[†], Takatoshi Fujiki[†], and Takuya Yoshihiro[‡**]
[†]Graduate School of Systems Engineering, Wakayama University, Japan
[‡]Faculty of Systems Engineering, Wakayama University, Japan
[*] s171053@sys.wakayama-u.ac.jp
[**] tac@sys.wakayama-u.ac.jp

Abstract - Bayesian Network is a graphical model that expresses causal relationship among events, which is regarded to be useful in decision making in various practical scenes. A number of algorithms to learn Bayesian Network structure from data have been proposed so far, but since the problem to learn Bayesian Network structure is proved to be NP-hard, it takes considerable time to learn sub-optimal structures. As one of the efficient approximation algorithm to obtain good (but not optimal) solution in practical time, EDA-based genetic algorithms are used. However, it still takes time to compute the solution on a CPU. Thus, in this paper, we propose a method to accelerate the EDA-based algorithm by designing parallel execution of the algorithm using GPUs. Through evaluation, we show that the proposed algorithm runs about 14-times faster than the original one executed on CPU. We also compare the quality of Bayesian network models created by major approaches in the literature, and found that EDA-based algorithm is superior to the others. We concludes that the proposed algorithm to learn Bayesian network structures is good in both quality and computational speed.

Keywords: Bayesian Networks, GPU, EDA, PBIL

1 INTRODUCTION

Bayesian Networks (BNs) are regarded as useful graphical models used to analyze causal relationship among events. There are so many practical fields in which Bayesian Networks are effectively utilized, such as bioinformatics research, medical analyses and diagnosis, computer security, system diagnosis and monitoring, etc. Recently, because we are surrounded by so much data coming from the Internet, sensors embedded to the environment, or various information systems, the importance of Bayesian Networks as analytic tools is continuously growing larger and larger.

A large number of studies have been dedicated to learn good BNs efficiently in the literature. It is well-known that BN structure learning can be formulated as an optimization problem that optimizes a model score defined as an information criterion. However, because it is proved to be NP-hard [1], to solve the problem approximately within practical time is significantly important. One traditional method is K2 [2], which introduce the constraint on variable order to reduce the search space. The variable order is the constraint on events $n_1, n_2, \ldots, n_k$ where n_i can be a parent of n_j only if $n_i \prec n_j$. However, because the order constraint such that $n_i \prec n_j$ cannot be defined frequently (i.e., it is possibly defined only in the apparent case, for example, n_i occurs before n_j in time), there are many cases in practice in which order constraint is not applicable.

To solve the optimization problem without order constraint, many studies tried to find sub-optimal BN models. Recently, algorithms based on genetic algorithms (GAs) are well-studied, as shown in the survey article [6]. Among them, we in this paper focus on a kind of GA-based algorithm called EDA (Estimation of Distribution Algorithms), in which the distribution of model scores on the graph space is estimated in order to find better-score Bayesian Network models efficiently. Note that EDA is one of the representative methods to compute BN models efficiently and approximately.

Specifically, we treat a EDA-based algorithm called PBIL (Probability-Based Incremental Learning) [12], which is reported to be the best to learn BNs among EDAs-based algorithms [7]. The problem here is that the above algorithms including PBIL-based one take significant time to learn BNs. To cope with the large data available today, acceleration of those algorithms to run within shorter time is important.

In this paper, we present a method to accelerate a PBIL-based BN-learning algorithm to run much faster using GPU computation. Our method incorporates parallel computation on GPU based on the coalesce access technique to accelerate the calculation of model scores such as AIC. We evaluated the proposed method using well-know bench-mark data sets, and found that the algorithm achieve about 14-times faster running speed than the original CPU-executed algorithm using consumer-class GPU hardware. We further compared the quality (i.e., optimality) of the solusions between major algorithm to learn Bayesian Networks in order to confirm the performance of PBIL-based algorithms. As a result, we found that the PBIL-based algorithm is superior to the other major approaches to learn BNs. From above, we conclude that the algorithm proposed in this paper computes good solutions within a short time by utilizing GPUs.

This paper is organized as follows. In Sec. 2, we describe the major approaches of algorithms to learn BNs in the literature. In Sec. 3, we introduce the optimization problem to learn BN structures and give the specific description of PBIL-based BN-learning algorithm. After we concisely explain the GPU architecture related to our work in Sec. 4, we describe the proposed method to accelerate the PBIL-based algorithm using GPU computation in Sec. 5. In Sec. 6 we evaluate the running speed and the quality of solution of the proposed method, we finally conclude the work in Sec. 7.

2 RELATED WORK

In this section, we concisely introduce the major approaches to learn BNs presented in the literature.

Local search is one of the most basic approaches to solve large-space optimization problems, including BN learning. Greedy hill climbing (GHC), which is the most basic one, explores the search space by moving to the best-score point among the vicinity of the current point. In learning BNs, GHC is usually applied to search the graph space. There are several variants in this sort of algorithms such as MMHC[4].

Simulated annealing (SA) is also a well-known local-search-based algorithm used in learning BNs [5], which is different from GHC in that SA does not always move to better-score point; SA has a parameter T called temperature, which is decreased gradually as iteration proceeds. In each iteration, SA selects randomly a point in the vicinity, and moves to it if the score of the new point is better, otherwise moves to it with the probability determined by T.

Genetic algorithms (GAs) are frequently used in NP-hard problems to obtain approximate solutions in a practical time. In GAs, *crossover* and *mutation* operators are applied to obtain a set of next-generation individuals. For learning BNs, one major strategy is called K2GA[3][8][9], in which GA is applied to evolve the order constraint from which K2 heuristic generates a graph structure. There are other ways to use GA to learn BNs such as graph evolution [10] and co-evolution algorithms [11] that evolve directly the graph structure.

As a kind of GA, EDA-based algorithms also have applied to learn BNs. In EDA, the distribution of scores in the search space is estimated using a set of individuals in a generation, and update the probability vector to generate individuals of the next generation. PBIL (Probability Based Incremental Learning), which is one of EDA, is first proposed in [12], first applied to learn BNs in [13], and reported to outperform other EDAs in learning BNs[7]. PBIL stops running when the probability vector converges. To avoid terminating and continue searching, several mutation operators such as bitwise mutation (BM)[14], transpose mutation (TM)[7], and probability mutations (PM)[15] have been proposed. Later, efficient repetition technique called PBIL-RS is proposed in [16] and reported to outperform the above mutation operators.

A few parallel algorithms to learn Bayesian Networks have been proposed in the literature. Nikolova, et al., proposed a parallel algorithm that searches graph space in parallel using conditional independence tests [19]. However, since this algorithm runs on multi-CPU platforms, it is not comparable to our algorithm that runs on GPU. Of course, utilizing muiti-CPU hardware is a possible choice. However, our choice, i.e., using GPUs, has an advantage that we can use well-populated and cheeper hardware to accelerate the structure learning of Bayesian networks. Linderman[20] and Wang[21], respectively, proposed an algorithm that accelerate MCMC sampling in the ordering space. Althrough MCMC-based algorithm could be a powerful choice to learn Beysian Networks, they are usually applied to the case in which each node (i.e., events) takes a real value and its statistical distribution is assumed. However, since in this paper we treats the case of discrete values, especially binary values in many cases, MCMC-based methods are too heavy and time-consuming. No parallel algorithm for GPU that is suitable to treat discrete-value cases has not been proposed so far.

3 PRELIMINARIES

3.1 Problem Formulation

A Bayesian Network model is a graphical model that represents the causal relationship among events. A Bayesian Network model has a structure represented by a directed graph where events are denoted by nodes while causal relationships are denoted by directed edges. In many cases (including this work), each node takes multinomial discrete values, and conditional probabilities among them are expressed by a model. See Fig. 1 for a concise example. Nodes n_1, n_2, and n_3 represent distinct events, where they take 1 if the corresponding events occur, and take 0 if the events do not occur (in this case we show a binomial case for conciseness). Edges $n_1 \rightarrow n_3$ and $n_2 \rightarrow n_3$ represent causal relationships, which mean that the probability of occurrence for each n_3 value depends on the values of n_1 and n_2. If edge $n_1 \rightarrow n_3$ exists, we call that n_1 is a parent of n_3 and n_3 is a child of n_1. Because nodes n_1 and n_2 do not have their parents, they have own prior probabilities $P(n_1)$ and $P(n_2)$. On the other hand, because node n_3 has two parents n_1 and n_2, it has a conditional probability $P(n_3|n_1, n_2)$. In this example, the probability that n_3 occurs is 0.890 under the assumption that both n_1 and n_2 occur. Note that, from this model, Bayesian inference is possible: if n_3 is known, then the posterior probability of n_1 and n_2 can be determined, which enables us to infer more accurately the occurrence of events.

The Bayesian Networks model can be learned from the data obtained through the observation of events. Let $N = \{n_i\}$, $(1 \leq i \leq |N|)$ be a set of events, and $O = \{o_j\}$, $(1 \leq j \leq |O|)$ be a set of observations, where $|N|$ is the number of events and $|O|$ that of observations. Let $o_j = (x_{j1}, x_{j2}, \ldots, x_{j|N|})$ be j-th observation, which is a set of observed values x_{ji} on event n_i for all $i (1 \leq i \leq |N|)$. We try to learn a good Bayesian Network model m from the given set of observations. Note that, good Bayesian Network model m is the one that creates data sets similar to the original observation O. As an model score (i.e., evaluation criterion) to measure the level of fitting between m and O, several information criteria such as AIC (Akaike's Information Criterion) [17] are used. Formally, the problem of learning Bayesian Networks that we consider in this paper is defined as follows:

Problem 1: From the given set of observations O, find a Bayesian Network model m that has the lowest model score.

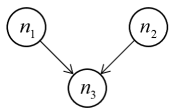
3.2 PBIL

In PBIL, an individual creature m is defined as a vector $m = \{e_1, e_2, \ldots, e_L\}$, where $e_i (1 \leq i \leq L)$ is the i-th element that takes a value 0 or 1, and L is the number of elements that consist of an individual. Let $P = \{p_1, p_2, \ldots, p_L\}$ be a probability vector where $p_i (1 \leq i \leq L)$ represents the probability to be $e_i = 1$. The algorithm of PBIL is described as follows:

(1) As initialization, we let $p_i = 0.5$ for all $i = 1, 2, \ldots, L$.

(2) Generate a set M that consists of $|M|$ individuals according to probability vector P, i.e., element e_i of each

n_1	$P(n_1)$
0	0.79
1	0.21



n_2	$P(n_2)$
0	0.88
1	0.12

n_1	n_2	$P(n_3 \mid n_1, n_2)$	
		0	1
0	0	0.999	0.001
0	1	0.710	0.290
1	0	0.360	0.640
1	1	0.110	0.890

Figure 1: An Example of Bayesian Network Models

P		Parent Node					
		n_1	n_2	…	n_i	…	$n_{\|N\|}$
Child Node	n_1	0.0	0.5	…	p_{i1}	…	0.5
	n_2	0.5	0.0	…	p_{i2}	…	0.5
	⋮	⋮	⋮	⋱	⋮	…	⋮
	n_j	p_{1j}	p_{2j}	…	p_{ij}	…	p_{Nj}
	⋮	⋮	⋮	⋮	⋮	⋱	⋮
	$n_{\|N\|}$	0.5	0.5	…	p_{iN}	…	0.0

Figure 2: A Probability Vector

individual is determined by the corresponding probability p_i.

(3) Compute the score for each individual $m \in M$.

(4) Select a set of individuals M^{Topk} whose members have evaluation scores within top k in M, and update the probability vector according to M^{Topk}. Specifically, the formula applied to every p_i to update the probability vector is shown as follows.

$$p_i^{\mathrm{new}} = ratio(i) \times \alpha + p_i \times (1 - \alpha), \qquad (1)$$

where p_i^{new} is the updated value of the new probability vector (p_i is replaced with p_i^{new} in the next generation), $ratio(i)$ is the function that represents the ratio of individuals in M^{Topk} that include edge i (i.e., $e_i = 1$), and α is the parameter called learning ratio.

(5) Repeat steps (2)-(4) until P converges.

By merging top-k individuals, PBIL evolves the probability vector such that the good individuals are more likely to be generated. Different from other genetic algorithms, PBIL does not include "crossover" between individuals. Instead, it evolves the probability vector as a "parent" of the generated individuals.

3.3 PBIL-based Bayesian Networks Learning

In this section, we describe a PBIL-based algorithm that learns BN models. Because our problem (i.e. Problem 1) to learn BN models is a little different from the general description of PBIL shown in the previous section, a little adjustment is required. In our problem, individual creatures correspond to each BN model. Namely, with the set of events N, an individual model is represented as $m = \{e_{11}, e_{12}, \ldots, e_{1|N|}, e_{21}, e_{22}, \ldots, e_{|N|1}, e_{|N|2}, \ldots, e_{|N||N|}\}$ where e_{ij} corresponds to the edge from an event n_i to n_j, i.e., if $e_{ij} = 1$, the edge from n_i to n_j exists in m, and if $e_{ij} = 0$ it does not exist. Similarly, we have the probability vector P to generate individual models as $P = \{p_{11}, p_{12}, \ldots, p_{1|N|}, p_{21}, p_{22}, \ldots, p_{|N|1}, p_{|N|2}, \ldots, p_{|N||N|}\}$ where p_{ij} is the probability that the edge from n_i to n_j exists. A probability vector can be regarded as a table as illustrated in Fig. 2. Note that, because BNs do not allow self-edges, p_{ij} is always 0 if $i = j$. The process of the BN-learning algorithm is basically obtained from the steps of the general PBIL, as described in the following (See also Fig. 3 that illustrate these steps).

(1) Initialize the probability vector P as $p_{ij} = 0$ if $i = j$, and $p_{ij} = 0.5$ otherwise, for each $i, j (1 \leq i, j \leq |N|)$.

(2) Generate M as a set of $|M|$ individual models according to P.

(3) Compute the evaluation scores for all individual models $m \in M$.

(4) Select a set of individuals M^{Topk} whose members have top-k evaluation values in M, and update the probability vector according to the formula (1).

(5) Repeat steps (2)-(4) until P converges.

Same as the genearal PBIL, the BN-learning algorithm evolves the probability vector so that we can generate better individual models. However, there is a constraint specific to BNs, that is, a BN model is not allowed to have cycles in it. To consider this constraint in the algorithm, step 2 is detailed as follows:

(2a) Consider every pair of events (i, j) where $1 \leq i, j \leq |N|$ and $i \neq j$, create a random order of them.

(2b) For each pair (i, j) in the order created in step (2a), determine the value e_{ij} according to P; every time e_{ij} is determined, if e_{ij} is determined as 1, we check whether this edge from n_i to n_j creates a cycle with all the edges determined to exist so far. If it creates a cycle, let e_{ij} be 0.

(2c) Repeat steps (2a) and (2b) until all the pairs in the order are processed.

These steps enable us to learn good BN models within the framework of PBIL.

3.4 PBIL-RS

Note that PBIL introduced above does not include mutation operators. Therefore, naturally, it easily converges to a local minimum solution. To avoid converging to the local minimum solution and to continuously improve the solution after that, several mutation operators have been proposed such as Bitwise Mutation (BM) [14], Transpose Mutation (TM) [7], and Probability Mutation (PM) [15]. PBIL-RS (PBIL-Repeated Search) [16] is also a method to avoid converging to local minimum solution, which, when it detects convergence,

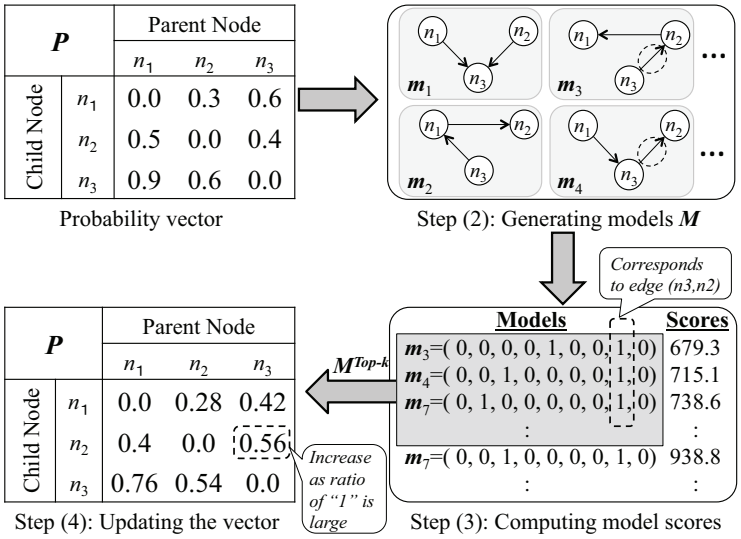


Figure 3: Overview of PBIL

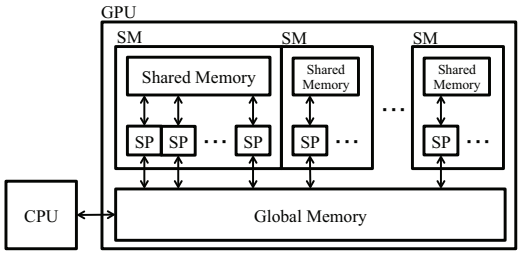


Figure 4: GPU Architecture

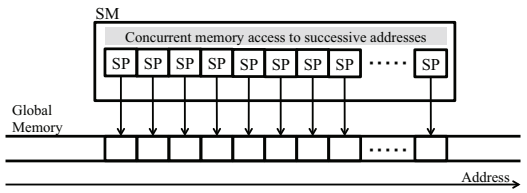


Figure 5: Coalesce Access

spreads the search area again. PBIL-RS is shown to find better solutions compared to the mutation-based methods such as BM, TM, and PM by repeating spreading and converging [16].

4 GPU ARCHITECTURE

4.1 GPU Structure

GPU (Graphics Processing Unit) is an arrayed processor that is originally developed to accelerate graphical computing, which currently is used to accelerate general scientific computation. A GPU structure is illustrated in Fig. 4. A GPU has a hierarchical structure where it consists of multiple (tens to hundreds of) Streaming Multi-processors (SMs) and a SM further includes multiple (tens to hundreds of) Streaming Processors (SPs). Since each SP in a SM concurrently executes a fragment of program code, a GPU executes a number of fragmented codes in parallel, which potentially results in significant performance.

To exchange data between a CPU and a GPU, a memory called *global memory* is prepared in the GPU that can be accessed by both the CPU and the GPU. Also, to accelerate parallel computation, each SM has a small high-speed memory called *shared memory* that can be accessed by all SPs in the same SM. A thread runs in each SP, and the threads in the same SM runs the same bytecode in parallel with different values of variables. Thus, memory accesses of threads are expected to occur simultaneously. To optimize the efficiency of the parallel access is the key issue to design algorithms for GPU. Note that, if the number of threads to execute exceeds the number of SPs in a SM, a single SP executes multiple threads in turn until all of them are executed.

4.2 Coalesce Access to Global Memory

To have as much performance gain as possible from GPU, one of the most basic techniques is to consider the efficient access to the global memory called *coalesce access*, which is illustrated in Fig. 5. Coalesce access is a synchronized parallel access technique in which threads in a SM simultaneously access the successive addresses in the global memory to achieve high-throughput memory access. When the access is scheduled completely to the successive addresses, the SM read/write the memory block in a single action that completes the access of all SPs. To utilize coalesce accesses is an important technique in designing GPU algorithms.

5 ACCELERATING PBIL WITH GPU

5.1 Overview

The method we propose in this paper extends PBIL and its family algorithms (such as PBIL-RS and the mutation extensions) to run in significantly shorter time by means of parallel computation of GPU. We re-designed Step (3) of PBIL described in Sec. 3 for GPU execution to compute the model scores for a collection of models M. Because, in PBIL, hundreds of models are to be computed in a single generation to estimate a distribution of model scores, introducing parallel computation in Step (3) is significantly effective.

Specifically, we detailed the step (3) in the following.

(3a) Transporting data from CPU to the global memory.

(3b) For each model m, we compute the evaluation score by executing the following substeps (3b-1) and (3b-2).

(3b-1) Counting the occurrences in the observation set that match each value pattern.

(3b-2) Computing evaluation scores from the counts.

(3c) Transporting the computed scores back to CPU.

As written in Step (3b), we first count the number of occurrences in the observation data O that match each value pattern of events. Here, value patterns are defined on each

event as a set of values taken by the event and its parent events. For definition, see Fig. 1 again. The value patterns on event n_3 is the combination of values of n_3 and its parents n_1 and n_2. Since those three variables take binomial values (i.e., 0 or 1), we have 8 value patterns such as $(n_1, n_2, n_3) = (0,0,0), (0,0,1), \ldots, (1,1,1)$. For conciseness, we denote it by $n_1n_2n_3 = \{000, 001, \ldots, 111\}$. In general, the set of value patterns for event n_i is denoted by $V_i = p_1p_2 \ldots p_rn_i = \{0..00, 0..01, \cdots, 1..11\}$ where the parents of n_i in model m is $\mathrm{pa}(n_i) = \{p_1, p_2, \ldots, p_r\}$ and r is the number of parents of n_i.

Then, our task in step (3b) is to compute the number of occurrences $\mathcal{N}_{iv}$ in O that takes value pattern v, for every event $i \in N$ and value pattern $v \in V_i$. Although there are several information criterion such as AIC, BIC and MDL that are used as model scores in learning Bayesian Network structure, they are all computed from the counts of value patterns, as shown in Sec. 5.3.

The basic strategy for counting value patterns is to assign a SM to a single model m, and to use all SPs in the SM in parallel to count all value patterns for all events in m. By assigning a model m to a single SM, we compute the model score of m in the SM. Each SM in a GPU processes models one by one in parallel to compute model scores for all models in M. In the following subsections, we describe the algorithm to process m within a SM to show how to make efficient manipulation of data, especially to gain from the coalesce access of global memory.

5.2 Data Structure

In Step (3a), we transport the data required to compute model scores to the global memory. We declare three arrays that represent the following sets, respectively, in the global memory.

(i) The observation set O.

(ii) The model set M.

(iii) An array to record the computed model scores.

The pseudo code to define these data items is shown in the following.

```
u_int8_t observation[N][O];
boolean model[M][N][N];
float modelScore[M];
```

Here, variables `M`, `N`, `O` represents $|M|$, $|N|$, and $|O|$, respectively. We represent the observation set O as a two-dimensional array `observation` where the 1st dimension is events and the 2nd observations. We represent The model set M as a three-dimensional array `model` where 1st dimension is used for the index of models and 2nd and 3rd dimensions are used to describe each model. Each model is expressed as an adjacency matrix such that `model[m][i][j]` is true if there is a directed link from event i to j in m. The array `modelScore` is used to retain the value computed by the algorithm.

We use the Shared Memory to place the counters that retain the counts of every value patterns for all events in a model, as follows.

```
u_int16_t counter[Vi];
```

Here, `Vi` denotes the number of value patterns on event $i \in N$. Note that `Vi` is determined depending on the number of parents of i in the model m, and the number of their multinomial values. Thus, this array may exceed the capacity of the shared memory. (Consider that, if p_j may take a value from $w(p_j)$ distinct values, `Vi` $= w(i)w(p_1)w(p_2)\ldots w(p_r)$.) If the shared memory can afford to store this array in size, we execute fast counting algorithm that we call *case-1* shown in Sec. 5.4, and otherwise, we use alternative algorithm that we call *case-2* shown in Sec. 5.5.

5.3 Model Scores

Note that we count the number of observations in each case to compute evaluation scores. Although there are several information criterion such as AIC, BIC and MDL, used as model scores in learning Bayesian Network structure, they all are computed from the number of observations in each case. For instance, AIC is computed with the following formula:

$$AIC = -2l(\theta|O) + 2k, \tag{2}$$

where θ denotes the parameter set, $l(\theta|O)$ denotes the likelihood of θ under observation O, and the term k represents the number of parameters. Here, we further show that the function $l(\cdot)$ is represented by the following formula

$$l(\theta|O) \propto \sum_{i \in N} \sum_{v \in V_i} (\mathcal{N}_{iv}) \log \theta_{iv}, \tag{3}$$

where i denotes an event, v denotes a value pattern, $\mathcal{N}_{iv}$ denotes the number of occurrences in observations O that match i and v, and θ_{iv} is a parameter computable from $\mathcal{N}_{iv}$. This means that, by counting the number of observations for each value pattern, we can compute the evaluation score of the model m such as AIC. Note that other information criteria used in Bayesian Networks such as BIC, MDL, etc., also can be computed through counting the matching value patterns in the observation set O.

5.4 Computing Model Scores (Case 1)

If the size of the array `counter[Vi]` is within the capacity of shared memory, the procedure described here (i.e., *case-1*) is applied. Before computing the model score of the given model $m \in M$, the SM responsible to this task computes $\mathcal{N}_{iv}$ for all $i \in N$ and $v \in V_i$. In the procedure, the SM proceeds each node i sequentially. Thus, we now fix $i \in N$ and focus on computing $\mathcal{N}_{iv}$ for all $v \in V_i$.

Our strategy to do this is to process occurrences in O in parallel using SPs. Thus, we designed the procedure such that each thread reads a single occurrence of O and increments the corresponding value in `counter`. Namely, we have as large number of threads as the occurrences of O. By the scheduler of GPU that assigns threads to SPs, we can read occurrences from successive addresses of the global memory as shown in Fig. 6 (remember that the addresses of occurrences for $n_i, p_1, p_2, \ldots, p_r$ are successive in array `observation`, respectively), gaining from coalesce access.

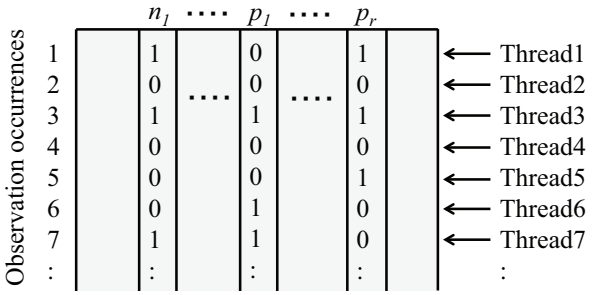


Figure 6: Counting Value Patterns (Case 1)

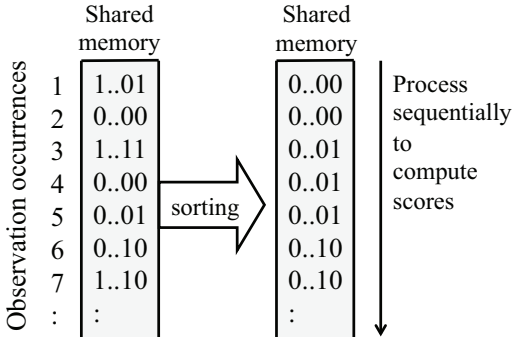


Figure 7: Counting Value Patterns (Case 2)

As the result of the above procedure executed for all events $i \in N$, we can obtain $\mathcal{N}_{iv}$ for all $i \in N$ and $v \in V_i$.

After computing $\mathcal{N}_{iv}$ for all $i \in N$ and $v \in V_i$, we compute the model score of m from them. This is simply done by computing the value according to formula (2). To compute it in parallel, we assign a SP for each $i \in N$ and sum up the model scores on the shared memory using a technique called *parallel reduction*. We finally store the computed score into the array `modelScore` in the global memory.

5.5 Computing Model Scores (Case 2)

If the size of the array `counter[Vi]` exceeds the capacity of the shared memory, we have to choose a less efficient algorithm. In the Alarm Network used in the evaluation of this paper, we can use the *case-1* algorithm only when the number of parents r is less than 7, where each event takes 4 distinct values and we have about 48KBytes shared memory.

In the *case-2* algorithm, instead of the array `counter[Vi]`, we define the array `patternValue[O]` as follows.

```
u_int16_t patternValue[O];
```

We simply use this array by storing pattern values of each occurrences of O. Retrieval of the pattern values from global memory can be done using coalesce access in the similar way to *case-1*. After retrieving the pattern values, we sort the values as shown in Fig. 7. Note that a GPU-specific sorting algorithm called *bitonic sort* can be used for high-throughput sorting. Then, we trace through the array sequentially in order to count each value pattern and sum up the model score. Although it takes a little longer than *case-1*, we can compute the model score in relatively short time.

Table 1: Evaluation Environment

OS		CentOS 5.0
CPU		Intel Core i7 4770k (3.50GHz)
Memory		32GBytes
GPU	Model	nVidia GeForce GTX TITAN Black (0.98GHz)
	# of SM	15
	# of SP per SM	192 (2880 SPs in total)
	Global Memory	6143 MBytes
	Shared Memory	49152 Bytes per SM
	GPU Library	CUDA 6.0
Compiler		g++ 4.1.2

6 EVALUATION

We evaluate the proposed method in terms of both running time and quality of the output model.

First, to clarify the performance of the proposed method to accelerate computational speed, we compare the running time of the proposed algorithm executed on GPU with its base PBIL-based algorithm executed on CPU. Note that their output is the same, only running time is different.

Second, as for the quality of the output model, we clarify how good is the BN models computed by PBIL-based algorithm. The quality of BN models can be measured by the model score such as AIC. By comparing the model scores obtained from major BN learning approaches, we show that PBIL-based algorithm is the most excellent among them.

By combining the results of running time and model quality, we would clarify that the proposed method is better than other major algorithms in the literature as it outputs higher-quality BN models within shorter running time.

6.1 Computational Time

We compare the running time of the proposed algorithm executed on GPU with its base algorithm PBIL-RS executed on CPU. We implemented both algorithms in C++ language with CUDA library for GPU processing. The execution environment is shown in Table. 1. We used Alarm Network [18] including 37 nodes as the base BN model; we generate an observation set including 1024 occurrences based on Alarm Network and learn BN models using each algorithm.

Figure 8 shows the computational time as generation proceeds. We see that the proposed algorithm that runs on GPU is about 14-times faster than PBIL-RS that runs on CPU only. In this figure, we also show a variant of the proposed algorithm seen as "case-2 only" that always runs *case-2* algorithm instead of *case-1* even if the number of parents is small. This variant takes about 1.6-times longer than the both-algorithm case, which indicates that *case-1* algorithm is considerably faster than *case-2*. To see the difference more precisely, we show the execution ratio of *case-1* and *case-2* in each generation in Fig. 9. Because mostly *case-1* algorithm is executed with 100% ratio, we can estimate that the *case-1* algorithm is about 1.6-times faster than *case-2*.

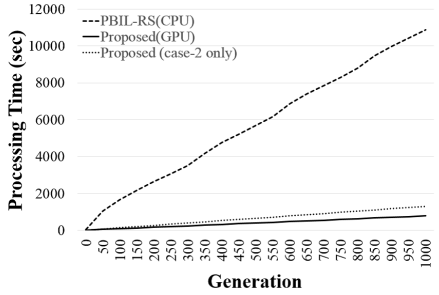


Figure 8: Execution Time (CPU vs. GPU)

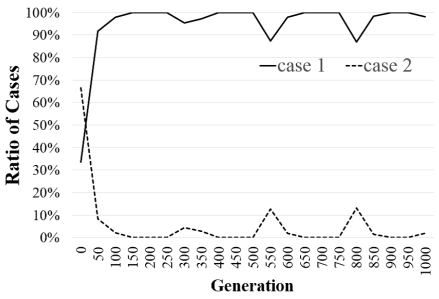


Figure 9: Ratio of Case 1 and 2

6.2 Model Quality

Next, we compare the quality of major BN-learning algorithms. As major algorithms to learn BN structure via optimization over model scores, we selected greedy hill climbing (GHC), simulated annealing (SA), PBIL, PBIL-RS, and K2GA. Note that GHC, SA, and PBIL are the type of algorithms that stop running when converged, and other two continue running until they are stopped by users.

Specific behaviors and parameter values are as follows:

GHC, in each iteration, always moves to the best model within the vicinity where we define that the two models are within their vicinity if one is generated by adding or deleting one edge from the other.

SA selects the next model in the vicinity and moves there if the score of the next model is better than the current one, and otherwise, stay at the current model with probability $e^{-\Delta/T}$ where Δ is the difference of the scores of the current and next models, and T is a parameter called temperature. T is initialized as 2500 and decreased at each iteration by multiplying $r = 0.99999$. As the vicinity to move in the next iteration, we apply the operation for every pair of nodes to add the edge (or delete if the edge already exists) in probability 0.004 (or 0.01 for deletion). These parameter values are determined to have the best performance through preliminary tests.

PBIL has several parameters as described in Sec. 3. In this evaluation, the number of individuals in each generation $|M|$ is 1000, the number of selected individuals k is 10, and learning ratio α is 0.1, all of which are determined to have the best performance through preliminary tests.

PBIL-RS is not specifically described in this paper due to space limitation, but all the definitions and parameter values are the same as [16].

K2GA applies GA to search the ordering space where K2 heuristic is used to convert from a node order to a graph structure. We use a typical crossover and mutation operator. Note that there are various crossover and mutation operators, but Ref. [3] reported that the performance is not so much different.

Table 2: Final Scores (Average)

Method	Final Score	Execution Time
PBIL	8777.7	3716 (Sec)
Simluated Annealing	9191.9	71590.7 (Sec)
Greedy Hill Climbing	8841.5	3142.1 (Sec)
PBIL-RS	8746.1	10881 (Sec)
K2GA	10011.9	302266.8 (Sec)

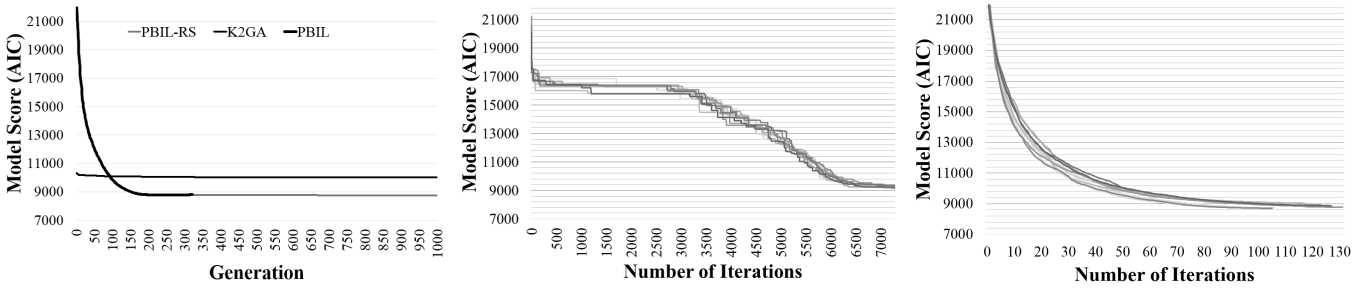
We show the comparison result of the GA-based algorithms K2GA, PBIL, and PBIL-RS in Fig. 10(a), which is the average of 10 repetitions. It is apparent that PBIL-based ones compute far better models than K2GA. Note that PBIL and PBIL-RS make almost the same curve, but PBIL stops around 320th generation due to convergence to a local minimum solution, whereas PBIL-RS continues running and find better models even after that.

As for non-GA algorithms, we show the score transition of 10 repetition of SA and GHC in Figs. 10(b) and 10(c), respectively. We see that all executions of each method make a similar curve until they converge and stop running. From those curves in Fig. 10(a)(b)(c), PBIL makes the steepest curve, meaning that the speed to approach better solutions is the highest among them.

Finally, in Table 2, we present the final scores and the running time of each algorithm. As for PBIL-RS and K2GA, we use the values of 1000th generation where improvement is scarcely seen in both algorithms, and for others we use the final values when they stop running. Among three algorithms that stop running, PBIL takes the best score, and PBIL-RS improves it by taking more execution time. Although GHC runs in a short time due to its simplicity, the score is lower than PBIL by more than 60. We also notice that the score of GHC would not improve by using more time, whereas the score of PBIL is continuously improved with time by using the technique of PBIL-RS. From above, we conclude that, in terms of scores, PBIL family would be the best algorithm among those compared.

7 CONCLUSION

We proposed a method to accelerate PBIL-based BN learning algorithms using GPU computation. We make the most of the coalesce access technique of GPU computation to reduce computation time using consumer level hardware. Through evaluation, we confirmed that the proposed method achieves about 14-times faster in running speed than the original PBIL-RS executed on only CPU. Also, we compared the quality of the computed BN models among several major approaches in the literature. As a result, we found that PBIL-based algorithms outperform other algorithms such as K2GA, greedy hill climbing, and simulated annealing. From above, we conclude that the proposed algorithm have an excellent performance in both speed and quality of solutions.



(a) PBIL(-RS) vs. K2GA (b) Simulated Annealing (SA) (c) Greedy Hill Climbing (GHC)

Figure 10: Score Transition

ACKNOWLEDGMENT

This work was partly supported by "the Program for Promotion of Stockbreeding" of JRA (Japan Racing Association).

REFERENCES

[1] D.M. Chickering, D. Heckerman, and C. Meek, "Large-Sample Learning of Bayesian Networks is NP-Hard," Journal of Machine Learning Research, Vol.5, pp.1287–1330 (2004).

[2] G.F. Cooper and E. Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data," Machine Learning, Vol.9, pp.309–347 (1992).

[3] P. Larrañaga, C.M.H. Kuijpers, R.H.Murga, and Y.Yurramendi, "Learning Bayesian Network Structures by Searching for the Best ORdering with Genetic Algorithms," IEEE Transactions on Systems, Man, and Cybernetics, Vol.26, No.4 (1996).

[4] I. Tsamardinos, L.E. Brown, and C.F. Aliferis, "The max-min hill-climbing Bayesian Network structure learning algorithm," Machine Learning, Vol.65, Issue.1, pp.31–78 (2006).

[5] A.S. Hesar, "Structure Learning of Bayesian Belief Networks Using Simulated Annealing Algorithm," Middle-east journal of Scientific Research, Vol.18, No.9, pp.1343–1348 (2013).

[6] P. Larrañaga, H. Karshenas, C. Bielza, and R. Santana, "A review on evolutionary algorithms in Bayesian Network learning and inference tasks," Information Sciences, Vol.233, No.1, pp.109-125 (2013).

[7] D.W. Kim, S. Ko, and B.Y. Kang, "Structure Learning of Bayesian Networks by Estimation of Distribution Algorithms with Transpose Mutation," Journal of Applied Research and Technology, Vol.11, pp.586–596 (2013).

[8] W.H. Hsu, H. Guo, B.B. Perry, and J.A. Stilson, "A Permutation Genetic Algorithm for Variable Ordering in Learning Bayesian Networks from Data," In Proc. GECCO'02, pp.383–390 (2002).

[9] E. Faulkner, "K2GA: Heuristically Guided Evolution of Bayesian Network Structures from Data," In Proc. IEEE CIDM'07, pp.18–25 (2007).

[10] A.P. Tonda, E. Lutton, R. Reuillon, G. Squillero, and P.H. Wuillemin, "Bayesian Network Structure Learning from Limited Datasets through Graph Evolution," In Proc. EuroGP'12, pp.254–265 (2012).

[11] O. Barriére, E. Lutton, and P.H. Wuillemin, "Bayesian Network Structure Learning Using Cooperative Coevolution," In Proc. GECCO'09, pp.755–762 (2009).

[12] S. Baluja, "Population-Based Incremental Learning: A method for Integrating Genetic Search Based Function Optimization and Competitive Learning," Technical Report CMU-CS-94-163 (1994).

[13] R. Blanco, I. Inza, and P. Larrañaga, "Learning Bayesian Networks in the Space of Structures by Estimation of Distribution Algorithms," International Journal of Intelligent Systems, Vol.18, pp.205–220 (2003).

[14] H. Handa, "Estimation of Distribution Algorithms with Mutation," Lecture Notes in Computer Science, Vol.3448, pp.112–121 (2005).

[15] S. Fukuda, Y. Yamanaka, and T. Yoshihiro, "A Probability-based Evolutionary Algorithm with Mutations to Learn Bayesian Networks," International Journal of Artificial Intelligence and Interactive Multimedia, Vol.3, No.1, pp.7–13 (2014).

[16] Yuma Yamanaka, Takatoshi Fujiki, Sho Fukuda, and Takuya Yoshihiro, "PBIL-RS: An Algorithm to Learn Bayesian Networks Based on Probability Vectors," In Proc. IWIN'2015 (2015).

[17] H. Akaike, "Information theory and an extension of the maximum likelihood principle," In Proc. ISIT'73, pp.267-281 (1973).

[18] I.A. Beinlich, H.J. Suermondt, R.M. Chavez, and G.F. Cooper, "The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks," In Proc. AIME'89, Vol. 38, pp.247-256 (1989).

[19] O. Nikolova and S. Aluru, "Parallel Bayesian Network Structure Learning with Application to Gene Networks," In Proc. CS'12, Article No.63 (2012).

[20] M. D. Linderman, R. Bruggner, V. Athalye, T. H. Meng, and N. B. Asadi, "High-throughput Bayesian Network Learning using Heterogeneous Multicore Computers," In Proc. ICS'10, pp.95-104 (2010).

[21] Y. Wang, W. Qian, S. Zhang, X. Liang, and B. Yuan "A Learning Algorithm for Bayesian Networks and Its Efficient Implementation on GPUs," IEEE Transactions on Parallel & Distributed Systems, Vol.27, No.1 (2016).

(Received October 10, 2016)

Takashi Mori received the B.E. degree from Wakayama University in 2016. He is currently a Master-course student in Wakayama University. He is interested in data mining, machine learning, and GPU computing. He is a student member of IPSJ.

Yuma Yamanaka received his B.E. and M.E. degrees from Wakayama University in 2015 and 2017, respectively. He is currently with KYOCERA Communication Systems, Co., Ltd. He is interested in data mining, machine learning, and Bayesian modeling.

Takatoshi Fujiki received his B.E., M.E. and Ph.D. degrees from Wakayama University in 2010, 2012 and 2017, respectively. He is currently with HIBIYA Resource Planning, CO., LTD. He is interested in data mining, machine learning, and bioinformatics.

Takuya Yoshihiro received his B.E., M.I. and Ph.D. degrees from Kyoto University in 1998, 2000 and 2003, respectively. He was an assistant professor in Wakayama University from 2003 to 2009. He has been an associate professor in Wakayama University from 2009. He is currently interested in the graph theory, dis- tributed algorithms, computer networks, medial applications, and bioinformatics, and so on. He is a member of IEEE, IEICE, and IPSJ.

Electronic Smell Picture Book for Children Using Pulse Ejection

Shohei Horiguchi†, Sayaka Matsumoto†, Hiroshi Shigeno‡, and Ken-ichi Okada‡

†Graduate School of Science and Technology, Keio University, Japan
‡Faculty of Science and Technology, Keio University, Japan
{shohei, matsumoto, shigeno, okada}@mos.ics.keio.ac.jp

Abstract - Human get a lot of information through the five senses. Information that can be gained using smell, one of the five senses, has a deep connection with memory and affect; therefore, presentation of scents with visual information has an effect that enhances realistic sensations. In addition, olfaction is also used to detect dangers such as rotten food or gas leaks. For these reasons, sense of smell is very important in daily life. It is important to use smell many times because it is formed primarily in childhood. However, there is little opportunity to use it compared with eyesight and hearing. In this paper, our objective was developing the electronic picture book with which children can experience scents repeatedly. It is also possible to present stimulations of other senses through wind, vibrations and sounds. In addition, it can read to children automatically with recorded voice and we can change effects presented for children. We conducted three experiments in order to confirm children could feel scents by using our book and evaluate usability of our electronic smell picture book. As a result, we found that children could feel scents by using our book and it had a good usability. This study is expected to give opportunities using senses, and to support growth of children.

Keywords: Electronic picture book, Children, Pulse Ejection, Olfactory display, Human computer interaction.

1 INTRODUCTION

Humans have five senses (eyesight, hearing, smell, touch, and taste), all of which are very important and are processed by the sensory system. In five senses, olfactory stimulation has a strong, direct connection to the limbic system, which controls emotion and memory. Therefore, videos accompanied by scents increase viewer concentration and retention of the video [1]. Olfaction is also used to detect dangers such as rotten food or gas leaks. For these reasons, olfaction is an essential sense in daily life. It is formed mostly in childhood owing to feeling scents many times. Typically, there are many educational toys for children concentrating on sight and hearing. On the other hand, there is little opportunity to experience smell. Hence, it is important to use sense of smell many times in childhood. Furthermore, it is also important for growth of children to be read picture books by parents. It results in development of linguistic competence and imagination. In our study, we focused on the experiences of scents and a picture book, and our objective was to develop the electronic picture book that infants can feel odors and other additional stimuli repeatedly. This book consists of the olfactory display which can present scent by pulse ejection and a tablet device. It can provide stimulations of other senses through wind, vibrations and sounds. Moreover, children can be read automatically by recorded voice. Parents can record voice used in automatic storytelling and add some effects to it by using edit mode of our book. We conducted three experiments. First one conducted for children for which in order to confirm they could feel four effects, especially smell, by using our book. Second and third ones were conducted for adults for which in order to evaluate the usability of our book's edit mode. In Section 2, we introduce related works about the plays for children and an electronic picture book. In Section 3, we propose our electronic picture book using olfactory display. Section 4 explains implementation of the book, and Section 5 assesses its usability. Finally, in Section 6, we present our conclusions.

2 RELATED WORK

Humans recognize an environment around them owing to using information that can be gained using five senses. These senses are formed primarily in childhood through experiences a lot of new things for them. Experiences using olfaction, touch and taste are important to enhance sensibilities of them [2]. In particular, children get many experiences thorough plays. Typically, there are a lot of plays giving stimulations to children's senses. As examples of such plays, there are building blocks, clay, simple instruments, paintings, picture books and so on. Building blocks and clay enhance a child's touch. The sound made by playing an instrument stimulates children's ears. It is said that music is essential for children's growth [3]. "I'm toy music station" sold as one of the intellectual toys contains nine instruments [4]. When children draw pictures, they use many colors. Humans have their likes and dislikes in colors naturally. Though color sense is an innate instinct, it is affected by the environment in which humans have grown up [5]. It is known that humans are more sensitive to familiar colors than unknown colors. The more we have come in touch with colors in childhood, the more we can recognize a variety of color. Therefore, there are many toys for children contain a lot of colors. Additionally, picture books also contain some colors, and stimulate a child's color sense. It is said that picture books develop a child's imagination and language ability [6]. There are many picture books recommended for a variety age, hence they play an important role for growth of a child [7]. On the other hand, the method of using picture books is not only reading by oneself but storytelling by parents. Storytelling by parents encourages not only the communication between parent and their child but also the development of the child [8], [9]. Keiko, D., et al. reported that children had been given storytelling by parents for a year

Figure 1: Wearable Olfactory Display

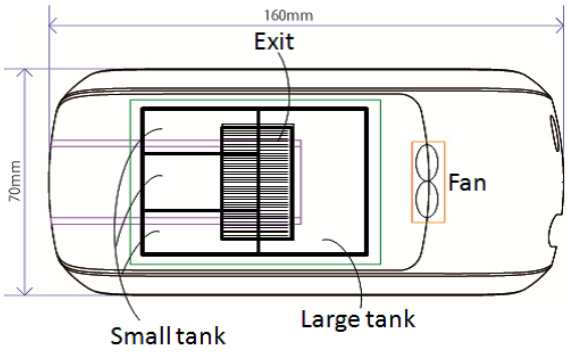


Figure 2: Overhead View of Olfactory Display

got better score in the confabulation test than children of the control group [10]. In recent years, electronic picture books come into vogue. They are composed of not only pictures but sounds or animations. They also have a function of automatic storytelling. It is reported that children showed more interest in electronic picture book than normal picture book when they use automatic storytelling [11]. There are these toys stimulating eyesight, hearing and touch without olfaction. Additionally, it is known that children have the least number of opportunities to use the sense of smell in five senses [12]. Therefore, children need to feel more odors. There is a smell book as one of the few toys that a child can experiences olfactory stimulations [13]. This book presents scent by breaking capsule including perfume. For this reason, children can never feel the odor after a capsule broken.

3 ELECTRONIC PICTURE BOOK USING SCENTS

It is important that human experiences many things about the five senses in the infancy when they are said to be the most sensitive. Children use sight, hearing and touch in play positively. However, there are fewer toys using sense of smell than toys using other senses. Therefore, it is said that children have the fewest opportunities to use olfaction in the five senses in daily life. On the other hand, it is hoped that the sensitivity of children is raised by smelling a fragrance. In this paper, we focus on smell and picture books which are familiar

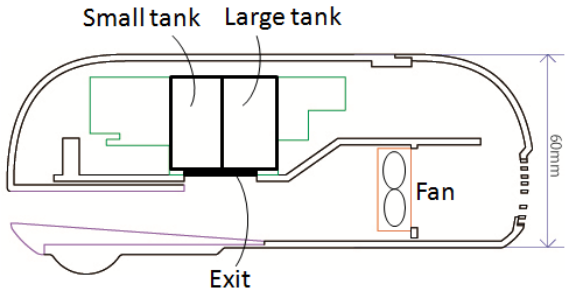


Figure 3: Sectional View of Olfactory Display

with children, and our objective was to develop the application of electronic smell picture book that children could feel four effects, especially smell, by using olfactory display. It is only once that children can experience a fragrance in the existing smell book because an odor is shown by breaking a capsule enclosing it on the illustration. Our electronic smell picture book can present an odor repeatedly by using the olfactory display which has technique of scent presentation in short duration. In addition, we use a tablet device for our book in order to make operation of it easy for children. Olfactory display is worn around the neck. Therefore, our application can smell a scent while operating the tablet. We assume that the target age of our book is 2-6 years old. The 2-3 years old children are often read by parents, contrary to this, the 4-6 years old children often read a book themselves. There are two methods of reading we prepare in order to entertain both younger and older children. One mode is that children use the automatic storytelling with recorded voice of parent to read. It is said that the voice of parents makes their children's mental condition comfortable. Another one is that they read themselves. These modes can decrease a burden to parents. Moreover, our application can provide stimulations of other senses through picture, wind, vibrations and sounds. It can stimulate four senses, olfaction, sight, hearing and touch, in total. In spite of reading by children, parents can record the their voice used in automatic storytelling and add some effects to recorded voice by using edit mode. To evaluate this application, we conduct three experiments about reading part and editing part. First one conducted for children for which in order to confirm they could feel four effects, especially smell, by using our book. Second and third ones were conducted for adults for which in order to evaluate the usability of our book's edit mode.

4 APPLICATION

4.1 Olfactory Display

The olfactory display we used in our electronic smell picture book is "Fragrance of Jet for Mobile." It is worn around the neck as shown in Fig. 1, and only the user can smell the emitted odors. Figure 2 and 3 show the plane and the slide views of it, respectively. This device adopts the thermal method used in ink-jet printers to emit odors. It has an ejection head, storing one large tank and three small tanks. Each tank stores an odorant, thus four kinds of odors can be con-

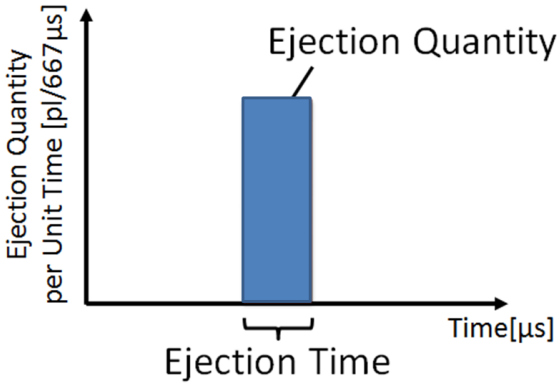


Figure 4: Pulse Ejection

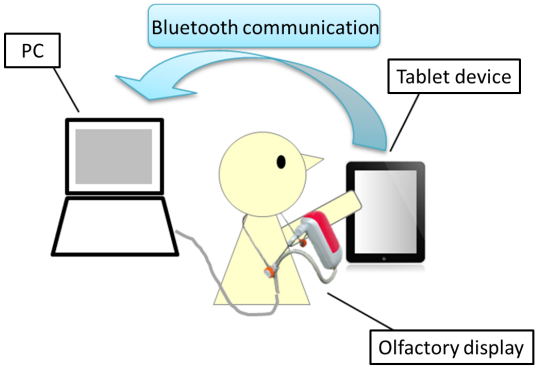


Figure 5: Conceptual Scheme of Application

tained. Odorants are emitted in picoliter (pl) quantities from small holes in the head on the wind by a fan. The average ejection quantity from the large tank's hole and each of the small tank's hole are 7.3 pl and 4.7 pl respectively. There are 255 holes in the head connected to the large tank and 127 minimum holes in the head connected to the small ones. Since this device can emit odorants from multiple holes at one time, the ejection intensity is controlled by the number of holes, of which the range is 0-255 in large tank or 0-127 in small tanks. In addition, the time of ejection can be controlled by 667 microseconds, the unit time. Hence, shown in Fig. 4, the total ejection quantity for one pulse is determined by the ejection time and the ejection quantity per unit time, which is determined by the average ejection quantity of using tank (7.3 pl or 4.7 pl) and the number of holes. The device is capable of this pulse ejection, allowing it to emit odors in a manner to avoid sensory adaptation [14]. Since pulse ejection can also prevent odors from lingering in the air, the device can also emit different odors for each page and switch odors quickly. It is therefore suitable for presenting odors with picture book.

In this paper, we use the wearable olfactory display in order that a child can smell odors while operating a tablet device. Almost children cannot stay still. Therefore, if a normal free-standing, rather than wearable, olfactory display is used by a child, position of nose would move to the area where he or she can not smell scents.

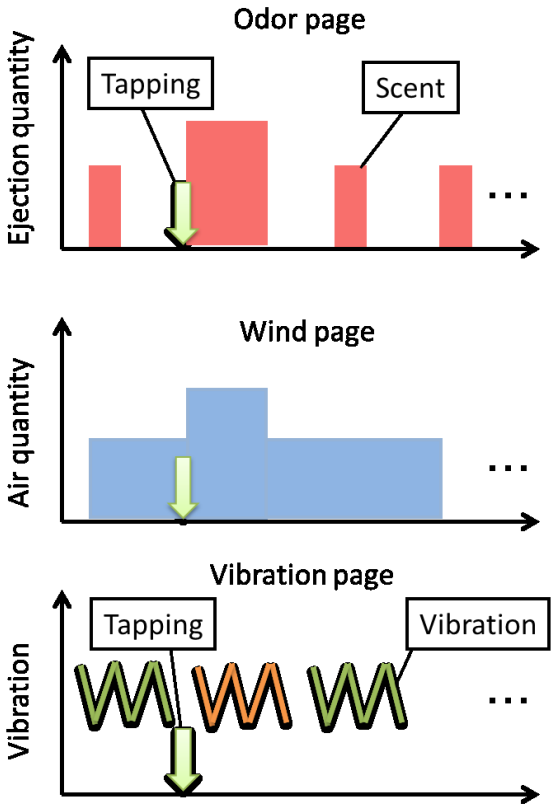


Figure 6: Presentation Methods of Odor, Wind, and Vibration

4.2 Electronic Picture Book

We suppose that our application is used by a child who is 2-6 years old. Therefore, we developed the application of an electronic smell picture book which such children can use easily. It is comprised of the olfactory display, a tablet device (Nexus9 [15]), and a computer which controls all devices. To send information on scent between a tablet and a computer, we use the Bluetooth communication system. Figure 5 shows a conceptual scheme of application. The application is implemented by two programs. One program is to send information about scent presentation which contains kinds of odor and ejection time from a tablet to a computer. Another one is that a computer orders the olfactory display to present scent.

4.2.1 Method of Presentation

Our book can present four effects: scents, sounds, wind, and vibrations. Scents and wind are presented by using olfactory display. A tablet presents sounds and vibrations. In this paper, we made an original picture book, rather than published one, in order that children can experience four effects. In our picture book, we prepared five kinds of pages presenting stimulations: we named them an odor page, a sound page, a wind page, a vibration page, and a mixed stimuli page. There are two presentation methods of odors in the odor page. They are showed in Fig. 6. First method is presenting an odor repeatedly by using pulse ejection while the odor page is displayed in order that the child can feel it unconsciously and concentrate on reading. Another method is a scent is empha-

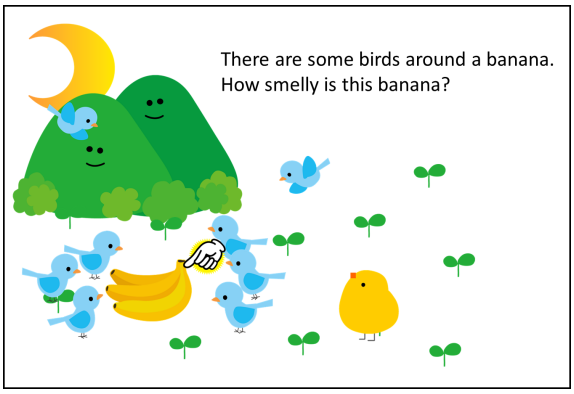


Figure 7: Odor Page of Picture Book

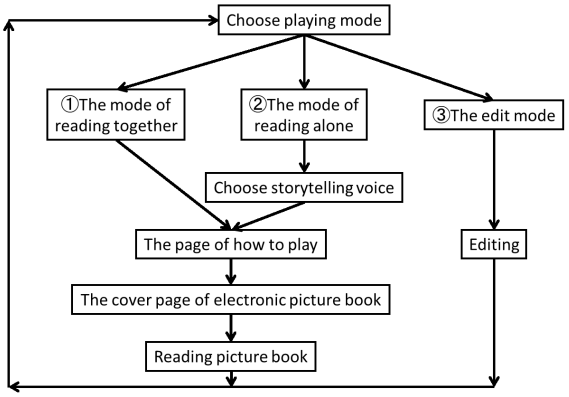


Figure 8: Operating Flow of Application

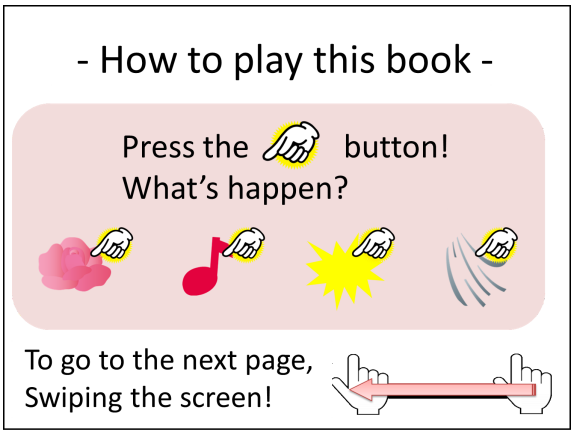


Figure 9: Screen of the Page Introducing How to Play

sized momentarily when the child taps the display of a tablet. The example of odor page is showed in Fig. 7. While the child see this page, scent of banana is presented to him or her. In addition, when the child taps the mark of hand on banana's picture, scent of banana is emphasized instantly. In the wind page, there are similar two presentation methods like in the odor page. They are showed in Fig. 6. The wind blows continuously from the olfactory display while the wind page is displayed. Moreover, when the child taps mark of hand in the page, strong wind blows momentarily. In the vibration page, there are similar two presentation methods too. They are showed in Fig. 6. The tablet vibrates continuously while the vibration page is displayed. When the user taps illustration of hand in the page, other kind of vibration is presented. Contrary to these pages, the sound page has only one method of presentation. When the child taps the mark of hand, a tablet produces the sound.

4.2.2 Three Modes of the Application

The operating flow of this application is showed in Fig. 8. There are three modes: the mode of reading together, the mode of reading alone, and the edit mode in our electronic book. The mode of reading together is used to read to the child by parent. The mode of reading alone is used when the child reads alone. In the edit mode, parents can record their voice which used by the automatic storytelling, and add effect to book. When the application starts, the user needs to choose among three modes at first.

- The Mode of Reading Together

At first, in the mode of reading together, the page which explains how to read the electronic smell book is showed in Fig. 9. In this page, the child can experience four effects: odor, sound, vibration, and wind. For example, if the user touches the mark of hand on the note, he or she could hear the sound effect. After feeling four effects, the user can go next page by sliding the screen with a finger from right to left. The next page is the cover page of picture book. After that the user can read a book by sliding the screen. If the child wants to go back previous page, he or she need to slide the screen from left to right. There are some marks of hand in various place of picture book. The user can get some effects by tapping the marks. When tapping the mark, a tablet vibrates in order to teach the user the timing of presentation. When "finish" button is pressed in the last page, the screen return to the page of choosing modes.

- The Mode of Reading Alone

In the mode of reading alone, the child can use automatic storytelling with recorded voice in order that he or she can read electronic smell book alone. The user can choose whether using automatic storytelling or not. First screen of this mode have "not use storytelling" button and "use storytelling" button. When "not use storytelling" button is pressed, explanation of how to play is displayed. When "use storytelling" button is pressed, new two buttons: "lady's voice" button and "parents' voice" button are appeared. Lady's voice is prepared at first. Parents' voice means recorded voice by parents. The user can choose which voice using in storytelling. After the user choose the voice, the explanation page of how to play is displayed. The automatic storytelling begins whenever the user slides page of picture book.

- The Edit Mode

We prepare the edit mode for parents. Parents can record their voice for storytelling, and conform effects to new recorded

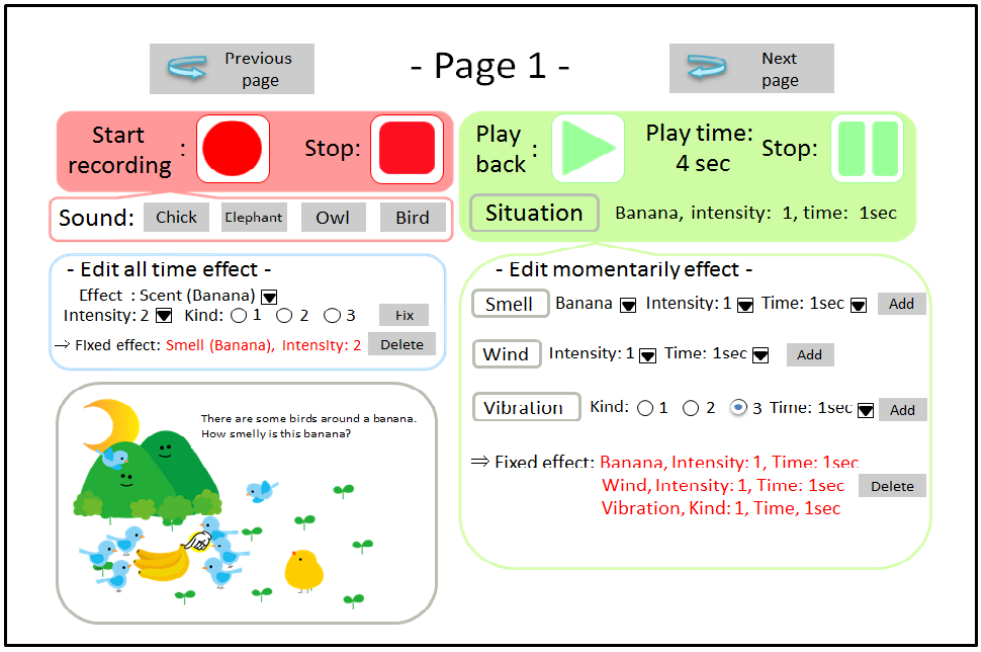


Figure 10: Screen of the Edit Mode

voice. Screen of the edit mode is shown in Fig. 10. Target of edit page is displayed at the lower left of the screen. There is a page number at the upper part of the screen. In both sides of the page number, there are "Previous page" button and "Next page" button to change the editing page. There are "Start recording" button, red "Stop" button, "Play back" button, and green "Stop" button under the page number. When "Start recording" button is pressed, recording is started. To stop the recording, parent needs to press red "Stop" button. The user can listen to their recorded voice by pressing "Play back" button. If parents want to add sound effect to storytelling, they need to push the sound buttons under "Start recording" button during recording the voice. In addition, if they want to add other effect, they need to use the area of "Edit all time effect" and the area of "Edit momentarily effect". The area of "Edit all time effect" is used to add the effect which is presented while reading target page. They choose the kind of effect from pull-down menu. If odor or wind is chosen, it is necessary to select the intensity of effect. In contrast, if vibration is chosen, the user needs to decide the kind of it. Next, when "Fix" button is pressed, effect the user chooses is fixed. The area of "Edit momentarily effect" is used to add the effect which is emphasized temporarily. If parents want to emphasize the effect, they need to press "Add" button while playing back the recoded voice. If the user wants to delete the fixed effect, he or she could delete it by tapping "Delete" button under the "Add" button. This application can use four odors(banana, apple, rose, pineapple), two levels of wind, three vibrations, and four sound as the effect.

5 EVALUATION

We conducted three experiments for evaluation of the application. First experiment evaluated whether the mode of reading alone could be used by children and whether they could feel four effects, especially smell, by using our book. Second and third experiments assessed whether adults could edit the effect by using the edit mode.

5.1 Experiment for Picture Book

5.1.1 Experiment Outline

We conducted the experiment in order to investigate whether children could use our electronic smell book and whether they could feel four effects, especially smell, by using our book. Subjects were 12 children (5 boys and 7 girls) who went to Hiyoshi Benesse nursery school. Their ages were from 5 to 6 (mean: 5.75, SD: 0.43). In this experiment, we used the original electronic smell picture book which has eight pages: four odor pages (banana, apple, rose, pineapple), a sound page, a wind page, a vibration page, and only picture page. We ordered subjects to read our book by using the automatic storytelling. We measured the number of tapping and the playing time of each page during experiment. After finishing experiment, we asked a question: "Which kind of pages did you enjoy better?" as a questionnaire. In addition, we asked another question: "Which scent did you like better?" to the subjects chose scent page in first question. Subjects could choose multiple pages or scents in these questions. When subjects used our application, they wore olfactory display, and took a seat. If it was difficult for them to wear olfactory display, we set it on the table. An experimental environment is showed in Fig. 11. The experiment for each subject took about ten minutes.

Table 1: Result of Tap Number in Four Odor Pages

Odor	Banana	Apple	Rose	Pineapple
Tap number	1.83 ± 1.85	1.67 ± 0.99	1.42 ± 0.79	2.00 ± 1.76

Table 2: Result of Tap Number in Four Effect Pages

Effect	Odor	Sound	Wind	Vibration
Tap number	1.73 ± 1.08	1.33 ± 0.65	1.67 ± 0.89	2.00 ± 1.13

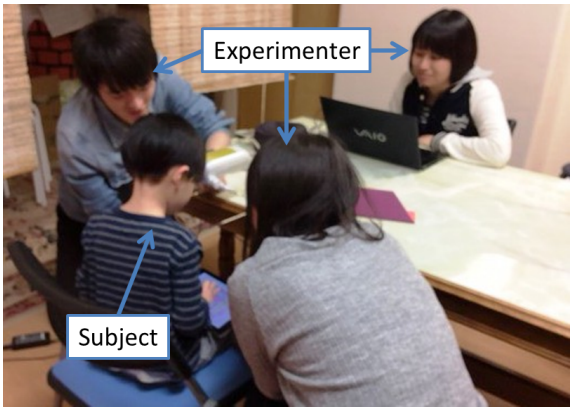


Figure 11: Experimental Environment

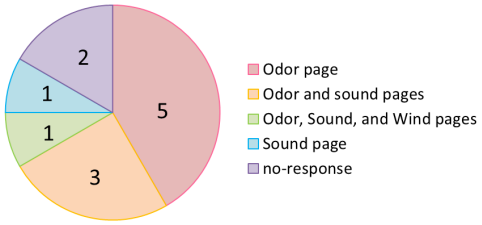


Figure 12: Result of First Questionnaire for Children

5.1.2 Result of Experiment for Picture Book

First, we considered about the number of tapping. The averages and standard deviations for the number of tapping in each odor page showed in Table 1. We analyzed whether or not the average was different based on the one-way analysis of variance. Then, there was not a significant difference ($p = 0.78 > 0.05$). Therefore, we calculated the tap number's average of odor page from four odor pages. Moreover, we showed the averages and standard deviations for the number of tapping in each effect page in Table 2. We analyzed the result in Table 2 by using the one-way analysis of variance. As a result, there was not a significant difference ($p = 0.41 > 0.05$). Hence, subjects could feel four effects on the same level by using our application.

Secondly, we considered about the playing time. The averages and standard deviations for the playing time in each odor page showed in Table 3. We analyzed whether or not the average of the playing time was different based on the one-way analysis of variance. Then, there was not a significant difference ($p = 0.78 > 0.05$). Therefore, we calculated the playing time's average of odor page from four odor pages. In addition, we showed the averages and standard deviations for the playing time in each effect page in Table 4. We analyzed the result in Table 4 by using the one-way analysis of variance. As a result, there was a significant difference ($p = 0.0007 < 0.05$). Hence, we used Tukey's test as multiple comparison. The result of it indicated that there is a significant difference between odor page and only picture page ($p < 0.05$). Accordingly, we found that children could enjoy odor pages better than only picture page.

Thirdly, we considered about the result of questionnaire which is showed in Fig. 12. We found that nine out of twelve subjects chose the odor page as an interesting page. For this reason, children tended to be interested in the odor page. We asked second question to nine children chose the scent page in first question. Their answers of second question is showed in Table 5. Only one child chose three scents, so that parameter is eleven. From Table 5, we found that odor of pineapple was most interesting for children. We thought that turn of pages affected this result. Page of pineapple was first page of picture book. Hence, pineapple could be tended to be impressive.

From the above results, we found that all subjects could felt various effects by tapping screen and have read our book, and they tended to like scent effect. We confirmed that the electronic odor picture book, we developed, could be used by children. Therefore, we thought our objective was achieved.

5.2 Experiments for the Edit Mode

5.2.1 First Experiment for the Edit Mode

We conducted an experiment in order to investigate whether adults could use the edit mode of our application. 15 subjects (10 men and 5 women) participated in this experiment. The participants were graduate and undergraduate students who were majoring in information engineering. At first, subjects practiced editing the effect of the picture book after lectured how to use the application by us. Next, we gave each subject six tasks: recording voice, adding wind, adding sound, adding vibration, adding scent of banana, and adding scent of rose. However, we also gave the rule in this experience. It is that

Table 3: Result of Playing Time in Four Odor Pages [sec]

Odor	Banana	Apple	Rose	Pineapple
Playing Time	12.9 ± 9.50	10.3 ± 5.60	10.8 ± 6.75	12.5 ± 6.78

Table 4: Result of Playing Time in Five Effect Pages [sec]

Effect	Odor	Sound	Wind	Vibration	Only picture
Playing time	11.6 ± 5.81	7.32 ± 4.62	10.4 ± 3.82	9.50 ± 3.21	4.57 ± 1.58

Table 5: Result of Second Questionnaire

Scent	Banana	Apple	Rose	Pineapple
Number of children	2	2	2	5

Table 6: Score of Questionnaire

	Score
Controllability	4.13 ± 0.52
Comprehensibility	4.60 ± 0.51

subjects could try only one time per one task. We evaluated the usability of the edit mode by a percentage of correct edit. After finished experiment, we asked some questions about usability of the application as a questionnaire.

5.2.2 Result of First Experiment

A percentage of correct edit was 97.8 % over 95 %. There were only two false edit: forgetting to press "fix" button and miss choice of effect. These mistakes could be caused by the rule we gave to subjects. In fact, the user can edit many times, in consequence, we thought that the user could decrease the miss edit. Next, we considered about the result of a questionnaire showed in Table 6. We used the five rated evaluation (1: bad - 5: good). The averages of score are showed in Table 6. The results indicated that our application had a good usability owing to both scores over 4. From the above results, we confirmed that the edit mode of our electronic book could be used by adults.

5.2.3 Second Experiment for the Edit Mode

We conducted second experiment about adding scent to recorded voice using edit mode. 15 subjects (10 men and 5 women) participated in this experiment. They were same persons participated first experiment of edit mode. In this experiment, subjects used the area of "edit momentarily effect" in edit mode in order to add scent to recorded voice. We gave each subject a task: adding scent confirming with specific timing of recorded voice. For example of specific timing, when recorded voice says last word: "na?" of "How smelly is this banana?", scent of banana have to be presented. An evaluation point of this experiment was a time lag between specific timing and timing of adding scent. We calculated it by using internal function of our application. Our application could record the timing of adding effect, so that we could get information of the time lag.

5.2.4 Result of Second Experiment

We calculated an average of subjects' time lags and standard deviation. They were 0.67 ± 1.18 sec under 1 sec. Therefore, we found that time lags of timings were very small, and our application can add effect along arbitrary user's timing. When users add the momentarily effect using our application, a guideline of adding effect was only recorded voice hearing from their ears. Hence, it was difficult for users to match the timing only using hearing. We thought that if there is visual guideline like a time bar in the edit mode, the time lag would be small. In the future, we will want to implement it in our application.

6 CONCLUSION

It is important for personal development to get a lot of information thorough the five senses in childhood in order to enhance sensitivity. Though there are many toys stimulating sight, hearing, and touch, there are a few things for olfaction. Children need to use the sense of smell many times. On the other hand, it is said that a storytelling is also important for a child's growth. In our study, we focused on the child's olfaction and storytelling, and our objective was to develop the application of electronic smell picture book that children could feel four effects, especially smell, by using olfactory display. The existing smell book presents a fragrance only one time, contrary to this, our application could eject a scent many times owing to the olfactory display. This electronic smell picture book can also stimulate other senses by presenting the wind, the vibration, and the sound. Moreover, it has a function of automatic storytelling using the recorded parent's voice. Therefore, a child can read this book alone. Parents

can also edit the effect of this book. We conducted the three experiments for children and adults. First one conducted for children for which in order to confirm they could feel four effects, especially smell, by using our book. Second and third ones were conducted for adults for which in order to evaluate the usability of our book's edit mode. As a result, subjects of children were able to operate this application, and to feel four stimulations. We found that children tended to be interested in the odor page of book, and, especially, liked a scent of pineapple. From this results, we thought our objective was achieved. The result of the experiment for adults indicated that they could edit the effect of the book correctly and add scents to recorded voice confirming with specific timing. From the above results, we confirmed that our application has a good usability. We hope that many children's sensitivity will be enhanced by using our electronic smell picture book.

7 ACKNOWLEDGMENTS

This work was supported in part by a Grant-in-Aid for Scientific Research (C), No. 26330229 (2014) from the Ministry of Education, Culture, Sport, Science and Technology Japan, Hiyoshi Benesse nursery school, and Takasago International Corporation.

REFERENCES

[1] M. Bowman, S. K. Debray, and L. L. Peterson, "Reasoning about naming systems", ACM Trans. Program. Lang. Syst, Vol. 15, No. 5, pp. 795–825 (1993).
[2] Development of smell, touch and taste. (in Japanese) http://www.jomf.or.jp/html/childcare_pdf/13.pdf
[3] K. Mochida and C. Kaneko, "Effect of preschool teachers for creative music expression of children", Technical Report. University of Bunkyogakuin, Vol. 10, No. 1, pp. 37–47 (2008). (in Japanese)
[4] How to choose an intellectual toy. (in Japanese) http://life.pintoru.com/intellectual-toy/
[5] W. A. Nagel, "Observations on the color-sense of a child", Jounal of Comparative Neurology and Psychology, Vol. 16, pp. 217–230 (2004).
[6] Y. Imai, et al., "A Comparative Study on Opinions about Methods of Reading Picture Books in Japan and Taiwan", Technical Report. Nara University of Education, Vol. 42, No. 1, pp. 211–223 (1993). (in Japanese)
[7] K. Sato, "Child Development and Picture Books", Technical Report. University of Ehime, Vol. 51, No. 1, pp. 29–34 (2004). (in Japanese)
[8] Y. Ishikawa, "Consideration of Picture Book: Production of the Zone of Proximal Development and Promotion of Cognitive Development in Children", The journal of Seigakuin University. Seigakuin University, Vol. 22, No. 1, pp. 165–179 (2004). (in Japanese)
[9] Y. Imai and J. Boi, "Effect of Reading Aloud Picture Books on Young Children's Comprehension of Emotions", Technical Report. Nara University of Education, Vol. 43, No. 1, pp. 235–245 (1994). (in Japanese)
[10] K. Dono, "The Effects of Experiences of Listening to Storytelling of Picture Books ('ehon-no-yomikikase') on the Development of Prosociality in Nursery School Children", Technical Report. Yasuda Woman's University, Vol. 36, pp. 81–91 (2008). (in Japanese)
[11] Report of experiment "storytelling using electronic books". (in Japanese) http://densholab.jp/page-29/page-1085
[12] S. Yamashita, "A Research about Using Olfactory Sense in Life Environment Studies", Departmental Bulletin Paper. Aichi University of education, Vol. 5, pp. 109–116 (2007). (in Japanese)
[13] Y. Kimura and C. Egawa, Otomodachi curry, Sekaibunka Corporation. (in Japanese) http://www.sekaibunka.com/book/exec/cs/12810.html
[14] A. Kadowaki, J. Sato, Y, Bannai, and K. Okada, "Presentation Technique of Scent to Avoid Olfactory Adaptation", Proc.of ICAT 2007, pp. 97–104 (2007).
[15] Nexus9 google. https://www.google.com/nexus/9/

(Received October 5, 2016)
(Revised December 13, 2016)

Shohei Horiguchi B.S. degree from the Department of Information and Computer Science of Keio University, Japan in 2015. He is currently working toward an M.S. degree in Open and Environment Systems at Keio University. His research interests include fragrance information processing work.

Sayaka Matsumoto B.S. degree from the Department of Information and Computer Science of Keio University, Japan in 2015. She is currently working toward an M.S. degree in Open and Environment Systems at Keio University. Her research interests include fragrance information processing work.

Hiroshi Shigeno received his B.S., M.E. and Ph.D. degrees in instrumentation engineering from Keio University, Japan in 1990, 1992, 1997. Since 1998, he has been with the Department of Information and Computer Science at Keio University, where he is currently a professor. His current research interests include mobile and ubiquitous computing, network architecture, and Intelligent Transport Systems. He is a member of IEEE, ACM, and IEICE.

Ken-ichi Okada received his B.S., M.E. and Ph.D. degrees in instrumentation engineering from Keio University, Japan in 1973, 1975, 1982. Since 1998, he has been with the Department of Information and Computer Science at Keio University, where he is currently a professor emeritus. His research interests include CSCW, groupware, human computer interaction, and ubiquitous computing. He has published 140 journal papers, 170 international conference papers, and 18 books entitled "Collaboration and Communication", "Designing Communication and Collaboration Support Systems", "Introduction to Groupware" and so on. He is a member of IEEE, ACM, IEICE and IPSJ. He was a chair of SIGGW, a chief editor of IPSJ Magazine, and chief editor of IPSJ Journal. Dr. Okada received the IPSJ Best Paper Award in 1995, 2000, and 2008, the IPSJ 40 the Anniversary Paper Award in 2000, IPSJ Fellow in 2002 and the IEEE SAINT Best Paper Award in 2004.

On the Generation of Human-oriented Counter-examples using a Test Automaton

Chikyu Yanagisawa†, Shinpei Ogata‡, and Kozo Okano*

†Graduate School of Engineering, Shinshu University, Japan
‡* Faculty of Engineering, Shinshu University, Japan
†15tm535d@shinshu-u.ac.jp
‡ogata@cs.shinshu-u.ac.jp
* okano@cs.shinshu-u.ac.jp

Abstract - To ensure quality in the process of software production, various techniques for software testing have been studied, but many conventional tests are known to take a lot of resources. Therefore, formal methods are attracting attention as a way of improving the quality of software. Model checking is one example of a formal method that inspects logically and exhaustively whether a given property is satisfied or not. A counter-example is a trace information to help with the localization of bugs and can be generated by a model checker when a given property does not hold. However, current model checkers often cannot generate counter-examples expected by users due mainly to their searching algorithms. We propose a method which derives an expected counter-example by combining model checking with a test automaton. The derivation method first creates a test automaton which roughly represents the behavior of the expected counter-example and then performs model checking on a parallel composition of the original automaton with the test automaton. We have applied the proposed method to a case study of water tanks of a chemical plant and confirmed its usefulness.

Keywords: counter-example, model checking, automaton

1 INTRODUCTION

With each coming year, advanced information societies need more reliable software and new techniques to develop such software. Software testing methods to ensure quality in the process of software production have been studied for many years. Conventional testing methods are, however, known to take a lot of resources. Therefore, formal methods are attracting attention as a way of improving the quality of software. A formal method is a method which describes the requirements and design of information systems (software and hardware) using a mathematical based language and provides a mechanism to infer that the system satisfies the requirements of users. This study uses model checking [2], which is one type of formal method.

Model checking inspects logically and exhaustively whether given properties are satisfied or not. It creates a model from source codes or systems and derives logical expressions from the requirements specification and inspection items to be entered into a model checker using a formal language. One application of model checking is "post-verifications" of a system. The procedure of post-verification utilizes models of the system when a fault has occurred [10]. It then determines the cause of the fault using formal methods, in contrast to conventional approaches which carry out cause isolation by log analysis of a particular system of failure.

Modern society depends on post-verifications in many situations because system faults can sometimes give serious impacts on human lives. As a specific example of a fault due to a system malfunction, we can recall the system troubles of a Japanese airline company that occurred as recently as 2016 [20]. When we perform post-verifications, a key element is the use of counter-examples. In general, a counter-example is regarded as "an example that refutes or disproves a hypothesis, proposition, or theorem."

When using counter-examples in post-verification, we treat properties of the system which must be fulfilled as "hypotheses, propositions, or theorems". Thus, a counter-example becomes an example of not satisfying important properties, and we can diagnose how a system fails by tracing it. Research has been conducted on how to generate counter-examples that are easy for humans to understand [8, 9].

The generation of counter-examples, however, may not produce ones which a user expects due to the searching algorithms used in a model checker. This trend is especially noticeable in cases where counter-examples include loop structures. In order to solve the problem, this paper proposes a method for creating test automata to guide the creation of counter-examples for a model represented by time automata [4] which are used in UPPAAL [5], an integrated tool for modeling, validation, and verification of real-time systems. To do this, our proposed method begins by creating a coarse behavior series of counter-examples represented in test automata [16–19]. A parallel composition of the test automata and the original automaton lead to counter-examples of the original model. In addition, we applied our technique to the diagnosis of a chemical plant system example and confirmed the effectiveness of the method.

Section 2 defines the time automata and test automaton used in our approach and Section 3 describes the proposed method for constructing counter-examples using test automaton components. Section 4 presents how we used our method to diagnose a system failure that occurred in the chemical plant systems example of our work. A discussion of the results is given in Section 5 and the summary and future work is given in Section 6.

2 PRELIMINARIES

2.1 Model Checking

Model checking [2, 3] of an automaton can be formulated as follows.

Definition 2.1 (Model checking)
Input1: an automaton A
Input2: a temporal logic expression p
Output: $A \models p$ or $A \not\models p$
Output(optional): If $A \not\models p$, then a counter-example CE

In general, computational tree logic (CTL) is used as a temporal logic for a timed automaton [5].

Intuitively $A \models p$ means that the behavior (all possible runs) of A satisfies the property expressed in p. Automaton A is also called a model. Thus, model checking is a process for checking whether a logic expression p holds under the model represented in A.

Typical properties are $\mathbb{AG}q$, $\mathbb{EF}q$ and so on. $\mathbb{AG}q$ and $\mathbb{EF}q$ mean that "for any path, q always holds," and "for some path, q eventually holds," respectively. $\mathbb{AG}$ and $\mathbb{EF}$ are called temporal operators.

For a state s, we can consider a property $\neg\mathbb{EF}s$, which means that starting from the initial state, the automaton cannot reach the state s.

Figure 1 represents the model checking process.

2.2 Timed Automaton

A timed automaton uses clocks to refer to time. The clocks can be regarded as precise analog clocks. Every clock is autonomous and will increase its value at the same uniform rate, independently from the behavior of the timed automaton. A timed automaton cannot control the behavior of the clocks except for a reset; it can neither put clocks forward, backward nor stop them. It can only reset some of the clocks. The reset clocks make their values 0 and immediately begin increasing their values again.

Definition 2.2 (Clock set C) *By C we denote a finite set of clocks. By x_i $(0 \leq i \leq |C| - 1)$ we denote an element (each clock) in C.*

When there is no confusion we can use literals (without an index) x, y, z, etc. to denote clocks.

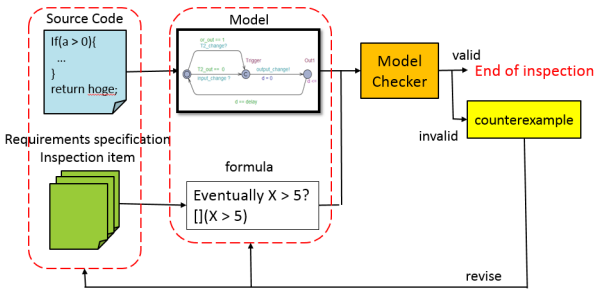


Figure 1: Model checking process.

Each clock has a time value represented as a non-negative real and the notion of "clock evaluation" is needed.

Definition 2.3 (Clock evaluation) *Clock evaluation $\nu(\in \mathbb{R}_{\geq 0}^{|C|})$ for a clock set C is a $|C|$-dimensional vector over $\mathbb{R}_{\geq 0}$.*
The i-th element ν^i of ν corresponds to the time value of clock x_i.

The term "evaluation" is originally defined in [15] as a mapping from clocks to reals. In this work, we define ν simply as a real vector.

In the following definitions we introduce two operations for expressing: 1) clock evaluation value changes according to its elapsed time, and 2) a reset by a timed automaton on some of its clocks when a transition fires.

Definition 2.4 (Operations on clock evaluation) *For a real value d, $\nu + d = (\nu^0 + d, \nu^1 + d, \ldots, \nu^{|C|-1} + d)$.*
For a set of clocks $\mathbf{r}(\subseteq C)$,
$r(\nu) = (r(\nu^0), r(\nu^1), \ldots, r(\nu^{|C|-1}))$, where

$$r(\nu^i) = \begin{cases} 0 : x_i \in \mathbf{r}, \\ \nu^i : \text{otherwise}. \end{cases} \quad (1)$$

The first operation $+d$ means that every clock increases its value uniformly and at the same rate. The second operation $r(\cdot)$ allows us to specify a subset of clocks $\mathbf{r}$ whose values are to be reset to 0.

Next we define clock constraints on C, which are used as guards and invariants of a timed automaton.

Definition 2.5 (Differential inequalities on C) *The syntax of a differential inequality* **in** *on a clock set C is given as follows:*

$$\mathbf{in} ::= x_i - x_j \sim a$$

$$\mid x_i \sim a,$$

where x_i and $x_j \in C$, a is a literal of an integer constant, and $\sim \in \{\leq, \geq, <, >\}$.
Differential inequalities $x_i \sim a$ and $x_i - x_j \sim a$ are true *iff $\nu^i \sim a$ and $\nu^i - \nu^j \sim a$ are* true, *respectively.*

Definition 2.6 (Clock constraints on C) *The syntax of a clock constraint cc on a clock set C is given as follows:*

$$cc ::= \text{true} \mid in \mid cc \wedge cc,$$

where in is a differential inequality on C.
$cc_i \wedge cc_j$ is true *iff both cc_i and cc_j are* true.
By $c(C)$, we denote the whole set of clock constraints on a clock set C.

Since a clock constraint f can be regarded as a function

$$f : C \to \{\text{true}, \text{false}\},$$

we introduce the notation of $f(\nu)$ which evaluates to true or false by evaluating each clock x_i as ν^i.

Now we can formulate a timed automaton. The semantics of a timed automaton will be defined later through a labelled transition system.

Definition 2.7 (Timed automaton) *A timed automaton $\mathscr{A}$ is a six-tuple (A, L, l_0, C, I, T), where*
A: a finite set of actions;
L: a finite set of locations;
$l_0 \in L$: an initial location;
C: a clock set;
$I : L \to c(C)$: a mapping from a location to a clock constraint, called a location invariant, or simply an invariant; and
$T \subset L \times A \times c(C) \times 2^C \times L$ is a set of transitions, where $c(C)$ is a set of clock constraints; and 2^C is a super set of sets of clocks.

Elements of the first and last L stand for the locations which the transition is starting from and going to, respectively. An element of A is an action associated with the transition. A clock constraint in $c(C)$ of the transition is called a guard. An element in 2^C is called a set of clocks to be reset.

For conciseness, we denote a transition $(l_1, a, g, r, l_2) \in T$ by $l_1 \overset{a,g,r}{\to} l_2$.

Example 1 *Figure 2 is a depiction of a timed automaton, $\mathscr{A}_L = (\{\text{press}\}, \{\text{off}, \text{dim}, \text{bright}\}, \text{off}, \{x\}, \emptyset, T)$, where* $T = \{\text{off} \overset{\text{press},\text{true},\{x\}}{\to} \text{dim},$
$\text{dim} \overset{\text{press},x\leq 10,\emptyset}{\to} \text{bright},$
$\text{dim} \overset{\text{press},x>10,\emptyset}{\to} \text{off},$
$\text{bright} \overset{\text{press},\text{true},\emptyset}{\to} \text{off}\}$.

Note that guards with value true *and empty clock resets are omitted in Fig. 2.*

Example 1 shows a timed automaton representing the behavior of a mug-light with two brightness modes. Here, we outline the behavior of this time automaton. The initial state is location "off" and the value of clock x is 0. If the "press" action fires, then the state is changed to location "dim", which means that the mag-light is dim. With this transition, the value of clock x is reset to 0. A timed automaton can stay in a location as long as its invariant is satisfied, but this example has no location invariants. The automaton can stay at location "dim" any length of time. If the value of clock x is greater than 10 units of time, the "press" action will change the location to location "off," which means the mug-light is switched off. If the value of clock x is less than or equal to 10 units of time, the "press" action will change the location to location "bright" which means press actions done in succession

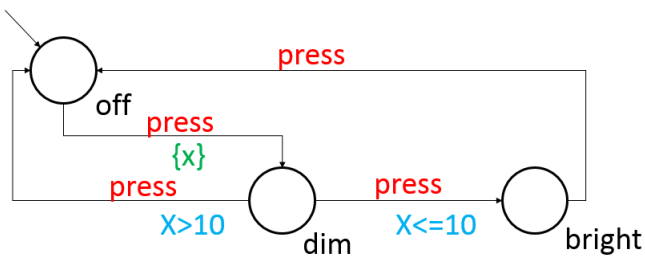


Figure 2: An example of a timed automaton representing a mag-light.

before the clock reaches 10 units of time makes the mug-light bright. At location "bright" the "press" action will change the location to location "off," regardless of the value of clock x.

The following is another example to explain the evaluation of a guard and an invariant.

Example 2 *Let C and $I(l_2)$ (a location invariant for l_2) be $\{x, y\}$ and $y > 6$, respectively.*

Consider a transition $l_1 \overset{a,x>0\wedge y\geq 3,\{y\}}{\to} l_2$.

For a clock evaluation $\nu = (8.2, 5.1)$, the values of $r(\nu)$, $g(\nu)$, and $I(l_2)(r(\nu))$ are $(8.2, 0)$, true, *and* false, *respectively, derived as follows:*
$r(\nu) = r(8.2, 5.1) = (8.2, 0)$
$g(\nu) = g(8.2, 5.1) = 8.2 > 0 \wedge 5.1 \geq 3$ =true
$I(l_2)(r(\nu)) = I(l_2)(8.2, 0) = 0 > 6$=false.

The dynamic behavior of a timed automaton can be expressed via a set of locations and a set of clock evaluations. Changes of one state to another can be the result of firing of an action or an elapse of time.

In order to define the semantics of a timed automaton, we first define a labelled transition system.

Definition 2.8 (Labelled transition system) *For a timed automaton $\mathscr{A}$, a labelled transition system (LTS) is a three-tuple (S, s_0, T), where S is a finite set of states, $s_0 \in S$ is an initial state, and T is a set of transitions, where $T \subset S \times (A \cup \mathbb{R}_{\geq 0}) \times S$.*

Here, the first and last elements in S stand for states in which the transition is starting from and going to, respectively, and A is a finite set of actions.

A transition (s, α, s') of LTS is denoted by $s \overset{\alpha}{\Rightarrow} s'$.
We can define a run of an LTS as follows.

Definition 2.9 (A Run of an LTS) *A run of a LTS (S, s_0, T) is defined as follows.*
$s_0 \overset{\alpha}{\Rightarrow} s'$ is a run of (S, s_0, T), if $s_0 \overset{\alpha}{\Rightarrow} s' \in T$.
Let σ_i be a run of (S, s_0, T), ending with state s_i. Then $s_i \overset{\alpha}{\Rightarrow} s_j \in T$, implies $\sigma_i \overset{\alpha}{\Rightarrow} s_j$ is also a run of (S, s_0, T).

The following define the semantics of a timed automaton.

Definition 2.10 (Semantics of a timed automaton) *For a given timed automaton $\mathscr{A} = (A, L, l_0, C, I, T)$, its corresponding LTS$(S, s_0, T)$ can be formalized as follows.*
$S = L \times \mathbb{R}^{|C|}_{\geq 0}$.
$s_0 = (l_0, \mathbf{0})$, where $\mathbf{0}$ is a $|C|$-dimensional vector each of whose elements is 0.

Definition 2.11 (Semantics of transition of a timed automaton) *For a transition $l_1 \overset{a,g,r}{\to} l_2$, its corresponding transition of LTS can be defined as follows.*

$$\frac{g(\nu), I(l_2)(r(\nu))}{(l_1, \nu) \overset{a}{\Rightarrow} (l_2, r(\nu))}, \qquad \frac{\forall d' \leq d \quad I(l_1)(\nu + d')}{(l_1, \nu) \overset{d}{\Rightarrow} (l_1, \nu + d)}$$

The first part is called an action transition, and the second is called a delay transition.

The first rule can be interpreted as follows. If the current clock evaluation satisfies the guard, and after some of the clocks in r are reset, and the new evaluation $r(\nu)$ also satisfies the invariant of location l_2, then $(l_1, \nu) \stackrel{a}{\Rightarrow} (l_2, r(\nu))$ can be fired.

The second rule is interpreted as follows. For some real d, and any d' such that $d' \leq d$, the obtained clock evaluation $\nu + d'$ satisfies the invariant of location l_1 and the control can stay in location l_1, but d units of time will have elapsed. In other words, the control can stay in l_1 until d units of time have elapsed.

Note that an action transition does not consume time, but a delay transition will consume time while staying in the same location.

Definition 2.12 (Run of a timed automaton) *For a timed automaton $\mathscr{A}$, a run σ is a finite or infinite run of its corresponding LTS.*

$\sigma = (l_0, \nu_0) \stackrel{\alpha_1}{\Rightarrow} (l_1, \nu_1) \stackrel{\alpha_2}{\Rightarrow} (l_2, \nu_2) \stackrel{\alpha_3}{\Rightarrow}, \ldots$,
where $\alpha \in A \cup \mathbb{R}_{\geq 0}$.

In general, for a timed automaton, we only consider runs in which delay transitions and action transitions alternately occur.

2.3 Test Automaton

In this paper, the test automaton described in [16–19] is used as guideline for deriving a desired counter-example by tracing transitions of interest in the original model.

Figure 3 represents the general usage of a test automaton. Normally, model checking uses a logical temporal expression to specify a property to check. However, complex specifications might not be able to be expressed using logical temporal expressions. A test automaton, in general, has more expressive power than a logical temporal expression and can represent a variety of complex behavior. A test automaton is typically used in a parallel composition with the original automaton in order to check a complex property. This is a useful technique, however, it is usually difficult to automatically generate a test automaton fitting the designer's need.

The test automaton TA_{id} used in this paper is just a timed automaton. We create a desirable test automaton from test automaton components using several operators in a process described in Section 3.2. This method creates a single test automaton TA_{id} for the original model M with human input and guidance. Then, it automatically generates the desired counter-example by performing model checking on a parallel composition of M and TA_{id}.

3 PROPOSED METHOD

3.1 Motivation Example

We use an example of a chemical plant system (described in more detail in Section 4) as our target for evaluation. Figure 4 shows the counter-example generated by UPPAAL when we perform model checking for the chemical plant system. Note that the model was obtained from a description of the chemical plant system which has some bugs in its design.

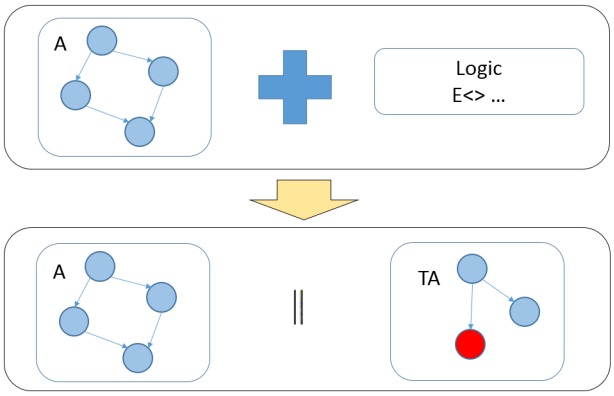


Figure 3: General usage of a test automaton.

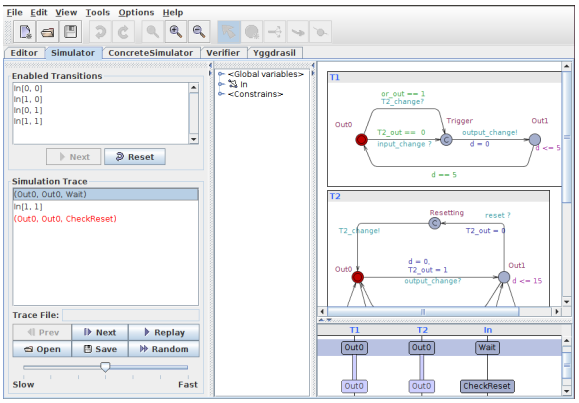


Figure 4: A counter-example generated in model checking of a chemical plant system.

We use the following expression as a property to check.

$$(In.C2 \wedge (T2\_out == 0)) - - > T1.Out1 \qquad (2)$$

This expression uses the "lead to" operation $- - >$ and the expression is equivalent to

$$\mathbb{AG}((In.C2 \wedge (T2\_out == 0)) imply \mathbb{AF}\ T1.Out1) \qquad (3)$$

which means that "when the proposition $In.C2 \wedge (T2\_out == 0)$ is satisfied, the state in which proposition $T1.Out1$ is true will surely be reached at some time."

When this property does not hold, we expect the model checker to generate a counter-example with more detailed information which can be used in subsequent failure diagnosis. In particular, for this example we expect to see as a counter-example the path which cannot reach the state $T1.Out1$ starting from state $In.C2 \wedge (T2\_out == 0)$.

However, the counter-examples actually generated specify only an initial state, which are clearly insufficient for use in fault diagnosis.

In order to solve the above problem, we propose a method using test automata in Section 3.2.

3.2 Method for Generating Test Automata

Our proposed method obtains a useful counter-example by performing model checking on a parallel composition of the original model which represents the behavior of the target system and a test automaton which represents the expected behavior of a counter-example that a user expects to see.

The method begins by creating a rough sketch of a counter-example that users expect as a test automaton. The test automaton is generated by synthesizing "test automaton components". Then, by using a parallel composition of the original model and the synthesized test automaton, we obtain the desired counter-examples in the following steps.

Let M be an original automaton (or automata) to be verified.

1. A user with some domain specific knowledge of the desired system considers an outline of the counter-example to be obtained. The necessary counter-examples are predicted from the system documents and faults report documents.
2. A single test automaton T is composed by synthesizing test automaton components. Weaving of test automaton components is done by using rename and fusion operators.
3. Model checking is performed on a parallel composition of M and T with property P which states "$M||T$ will not reach the final state". If P does not hold, a counter-example is obtained.

In creating the parallel composition, the original model M should be changed as little as possible. However, in order for it to communicate with the test automaton, the following modifications may be required. Note that these modifications can be performed automatically.

If M has a transition with some event $a!$, $a?$, or some variable x which is also used in the test automata, then the following modifications are performed.

1. If M has a transition with event $a!$ $(a?)$, then add a transition which has a synchronization signal $s!$ to a test automaton. Figure 5 shows the modification. Here the location "C" denotes a committed location which is a location where the automaton does not stay. Thus, the event $s!$ is performed immediately after the event $a!$
2. If M has a transition with a variable update $x = expr$ and x is also used in a test automaton, then add a transition which has a synchronization signal $s!$ to the test automaton. Figure 6 shows the modification. Also variable x should be declared as a global variable.

3.3 Test Automaton Components and Operators for Weaving

Figure 7 shows the general form of a test automaton component

$$TA_{id}(L_{in}, a, guard, update, reset, L_{out}, I_s, I_t)$$

Figure 5: Modification of event a!.

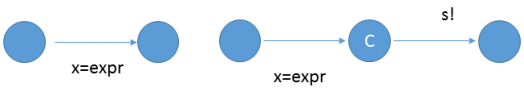


Figure 6: Modification of a transition with update variable.

where L_{in} and L_{out} are the entering location and exiting locations, respectively; $a, guard, update, reset, I_s$, and I_t are an event, a guard, an update, a clock set to reset, invariants for entering and exiting locations, respectively.

In this work, we propose three typical patterns of test automaton components shown in Fig. 8. Let T be the transition of interest in the original model. We consider two types of actions given to the test automaton when tracing the transition of the original model. The first type communicates with transition T and the second communicates with transitions other than T. When transition T fires, the instance of the component of the test automaton simply fires the corresponding transition. This is the first test automaton component. When a other transition other than T fires, we consider the following two scenarios. If the original model can return to a specified state via a transition other than T, i.e., if the original model has a loop, the test automaton must have a transition to the corresponding state. Therefore, a new component is needed. If the original model cannot return to the specified state, i.e., when the original model does not have a loop, the test automaton must have a transition to an error state. If a test automaton reaches an error state, the task of obtaining an appropriate counter-example fails. These three patterns are used as the components of the test automaton.

First we explain the simple component. Figure 8 (i) is a test automaton component to receive signal a and can be written as

$$TA_1(L_{in}, a, true, \emptyset, \emptyset, L_{out}, \emptyset, \emptyset).$$

Using the (i) type test automaton components, we can compose a test automaton which can receive consecutive signals.

For example, let us consider the test automaton components in Fig. 9.

$$TA_1 = (L_{in}, a1?, x > 0, \emptyset, \emptyset, L_{out}, \emptyset, \emptyset)$$

$$TA_2 = (L_{in}, a2?, true, \emptyset, \emptyset, L_{out}, \emptyset, \emptyset)$$

We can use a rename operator ($TA@label\backslash label2$) to change the labels of test automaton components.

For example, $TA_1@L_s\backslash L_{out}$ and $TA_2@L_s\backslash L_{in}$ will rename L_{out} of TA_1 and L_{in} of TA_2 in Fig. 9 each to L_s as shown in Fig. 10.

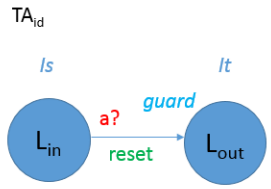


Figure 7: Test automaton component.

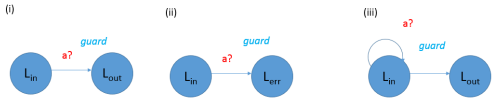


Figure 8: Typical test automaton components: (i) a normal transition, (ii) a force to an error state and (iii) a transition to itself.

The fusion operator $(+)$ is a binary operator which merges locations in both terms.

For example, $(TA_1@L_s\backslash L_{out})+(TA_2@L_s\backslash L_{in})$ will merge L_s of TA_1 and L_s of TA_2 in Fig. 10 to the result shown in Fig. 11.

Figure 8 (ii) forces the test automaton to an error state. It is used when the conditions required for a counter-example are not met.

Figure 8 (iii) stays in a location. It cannot reach location L_{out} while the event happens with a condition guard.

4 CASE STUDY

We have applied the method proposed in Section 3 to the verification of a chemical plant system reported in IPA [10] as a case-study for the "post-verification" of a system.

Figure 12 shows a schematic diagram of the system. The functionality of the system is as follows:

- When the water level is more than a prescribed alert level (40cm), the system opens a discharge valve for 5 seconds to prevent overflow. During this 5-second period and the 10 seconds following it (15 seconds total) the system will not accept a new open instruction.
- The system can also perform the same discharge operation as an instruction from a human operator.
- An instruction from the operator always takes precedence over the other instructions. The system accepts it even during the prohibited interval of 15 seconds. Operator instructions have priority over even past instructions made by the operator.

In operation, even when the water level had exceeded the alert level and the 15-second wait period for new instructions had passed, the appropriate action was not taken by the system. Furthermore, instructions from the human operator which should always have precedence were also ignored. We performed modeling using UPPAAL in order to diagnose this

Figure 9: Test automaton components 1 and 2.

Figure 10: Renamed test automaton components.

failure. Model diagnoses in Figs. 13 and 14 show the control system and a model expressed in UPPAAL, respectively. Figure 14 is derived from the specification of the chemical plant system and the model in Fig. 13.

In Fig. 13, signal $M4$ represents a manual input from the operator and signal $C2$ represents a signal which shows that the water level has exceeded the alert level. For both signals, a 1 indicates an input signal for starting the discharge and a 0 represents no input. Based on these inputs we have two timers for representing the state where the system is draining water and the 15-second wait state when new instructions are not accepted. The specific logic between the inputs and the timers is given in the "In" model (Fig. 14).

In Fig. 14, a double circle represents the initial state. A location with a C represents a committed location. A committed state cannot be delayed and the next transition must involve an outgoing edge of at least one of the committed locations. Expressions in green represent a "guard" which evaluates to a Boolean value. If a guard expression does not hold, the corresponding transition cannot fire. An expression in yellow represents a "select" and in this model, we use it to receive an input from a user (C_2: 0 or 1, M_4: 0 or 1). An expression in blue represents an "update". The model updates variables at the same time as the corresponding transitions. An expression in light blue represents a "synchronization". A synchronization label is formed as either $Expression!$ or $Expression?$. A transition with $Expression?$ fires on receiving the corresponding transition with $Expression!$.

Typically, formal methods decide whether properties which we expect to hold really hold on the model. In this case, it would be natural to consider the negation of a failure that has occurred in the system as a property to check using model checking. In our example, we must determine "whether the system is sometimes put into discharge mode when water draining is not suppressed and the water level sensor is in a state of alert level."

We check Expression (2) in Section 3.1 on the model. As shown in Fig. 4, it is confirmed that this property is not satisfied. The generated counter-example, however, is trivial and it cannot be used to explain why the property is not satisfied. Therefore, it cannot be used for the fault diagnosis of the system.

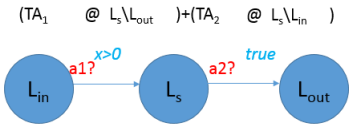


Figure 11: Fusion of test automaton components.

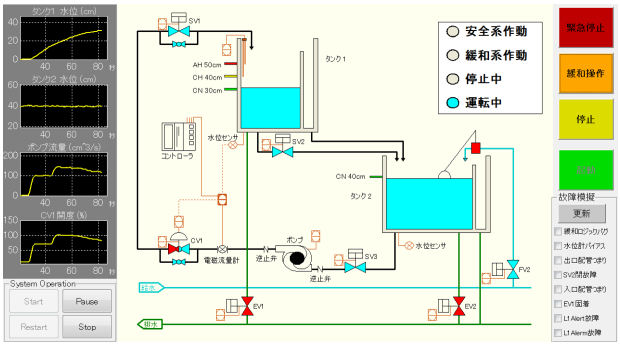


Figure 12: A schematic view of the chemical plant system example from "Fault diagnosis method for the large-scale and complex embedded system (in Japanese)"[10].

Our method creates a test automaton in a series of steps to generate a human expected counter-example. The test automaton is derived by a coarse behavior series of counter-examples in our previous work [10, 11]. Let TA_1 be a test automaton component obtained from type (i) components in Fig. 8.

$$TA_1 = TA_{id}(start, input?, C2 == 1 \&\& M4 == 0,$$
$$isT1end = 1, \emptyset, end, \emptyset, \emptyset)$$

Similarly, TA_2, TA_3, and TA_4 can be obtained from test automaton components in Fig. 8.

Figure 15 summarizes these test automaton components. A test automaton obtained by the following expression is the desired final automaton and is depicted in Fig. 16:

$$(TA_1@L_s\backslash end) + (TA_2@L_s\backslash start, L_t\backslash end)$$
$$+(TA_3@L_s\backslash start) + (TA_4@L_t\backslash start)$$

By combining the model representing the system and the test automaton, a detailed counter-example which can be used in the diagnosis of the failure is obtained(Fig. 17). The odd-numbered lines represent the states of the semantics model of the given network of timed automata. For example, (Out0, Out0, Wait, start) stands for the first timed automaton located in Out0, the second timed automaton located in Out0, and so on. In particular, the last element "start" corresponds to the automaton in Fig. 16. The even-numbered lines represent transitions of the semantics model of the given network of timed automata. For example, "In [1, 0]" on the second line and "In [1, 1]" on the 12th line represents a select input from the user. "input_change: In → T1" on the eighth line represents the fire of a transition "input_change!" on the model "In" and transition "input_change?" fires on the model "T1" also.

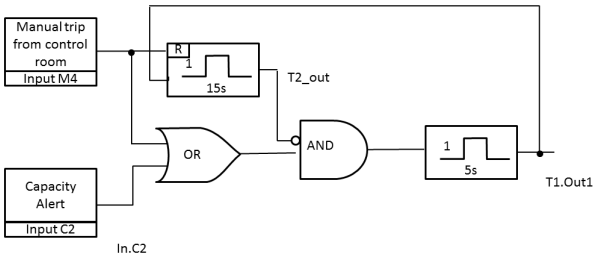


Figure 13: Control diagram of the chemical plant system.

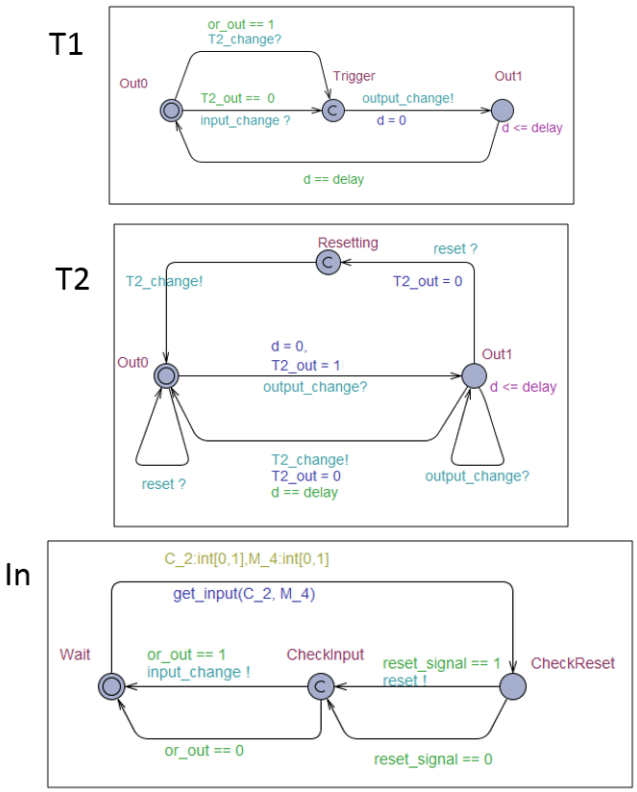


Figure 14: UPPAAL model of chemical plant system example.

5 DISCUSSION

We discuss here the coverage and time taken in generating a counter-example using test automaton components in this work.

The desired counter-example for the case-study presented in this paper was able to be constructed by the proposed test automaton components because the counter-example was generated as a single path. We can expect a variety of cases to be covered sufficiently using the same method.

The time required for counter-example generation was less than 1 second in the example of this work. Since the proposed method uses a test automaton as a guideline for creating counter-examples, it is assumed that the scalability of generation time will fall within an appropriate range even if the counter-example and scale of the time automaton become bigger. In addition, the creation of the test automaton here

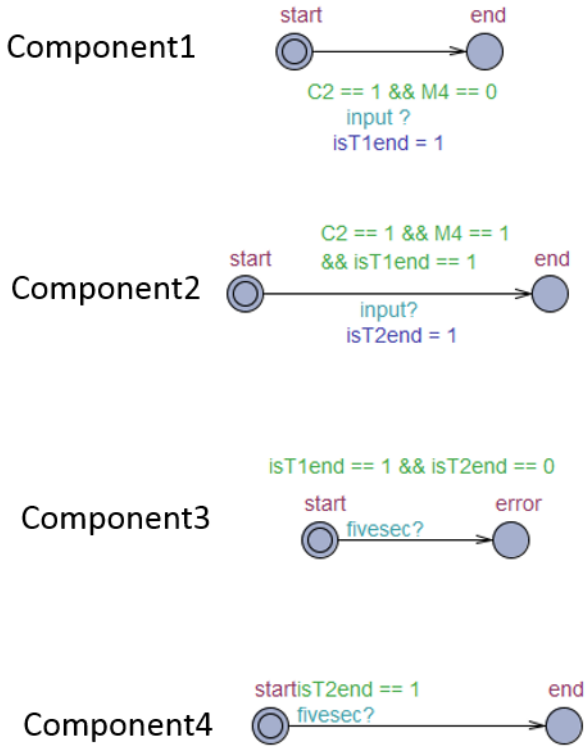


Figure 15: Components of a test automaton for the plant example.

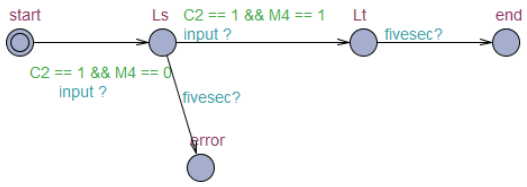


Figure 16: Test automaton.

was by the authors. Therefore, it should be evaluated in the future whether or not a novice in fault diagnosis can create a correct automaton with the prescribed process and how long it would take. This method involves only simple steps of assembling the test automaton from the parts that are provided, however, knowledge of model checking is required to create a useful test automaton. Therefore, the method targets a user with prior knowledge of model checking.

6 CONCLUSION

This paper proposed a method for generating a counter-example which meets user expectations using test automata. We applied the technique to a case study of a chemical plant system in order to verify its effectiveness.

The proposed method is not designed to generate counter-examples automatically. This is an approach that combines model checking and test automata with human guidance to

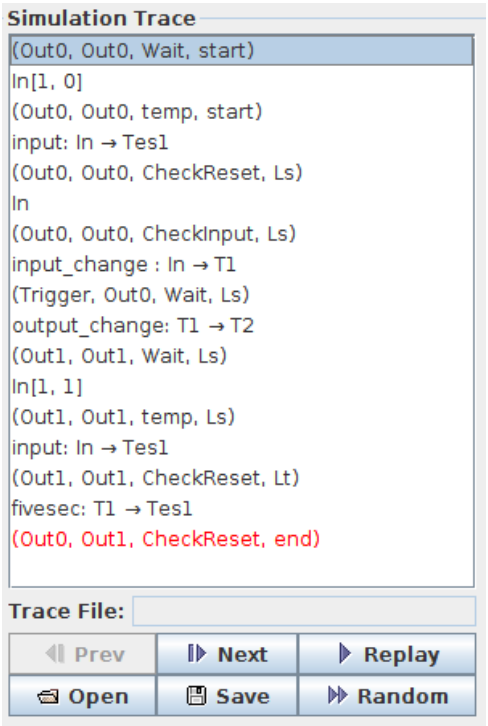


Figure 17: The counter-example generated with the proposed method.

generate a useful counter-example. Therefore, a method for generating counter-examples automatically is needed in the future and we must consider the following problems:

- We cannot fully search the model space in the case of an infinite state transition system.
- Even in a finite state transition system, state-explosion problems can occur.

Therefore, model abstraction techniques to properly reduce the number of states of a model for each property [13, 14] become important. Bounded model checking (BMC) [12] would provide another approach. The BMC technique prevents state explosion by limiting the search range of the finite state space. When BMC finds a violation on a finite state space, counter-examples are generated as finite lengths. In general, counter-examples have infinite length. However, users typically want finite counter-examples and consider a counter-example generated by BMC enough for their purposes. In our future work, we will examine BMC as a promising starting point for generating counter-examples of suitable length within a reasonable amount of time.

ACKNOWLEDGMENTS

This work is being conducted partially as Grant-in-Aid for Scientific Research S (25220003) and C (16K00094). Funding from Mitsubishi Electric Corp. is gratefully acknowledged. Finally the authors wish to acknowledge Dr. Nakajima, Professor of NII for discussion.

REFERENCES

[1] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, and M. Deardeuff, "How Amazon Web Services Uses Formal Methods," Communications of the ACM, Vol. 58, No. 4, pp. 66–73 (2015).

[2] E. M. Clarke, O. Grumberg, and D. A. Peled, Model Checking, MIT Press, (2000).

[3] K. Okano, T. Nagaoka, T. Tanaka, T. Sekizawa, and S. Kusumoto, "Parallel Multiple Counter-Examples Guided Abstraction Loop —Applying to Timed Automaton—," International Journal of Informatics Society, Vol. 8, No. 3, to appear (2016).

[4] R. Alur, "Timed Automata," Proceedings of 11th International Conference of Computer Aided Verification, (CAV '99), Vol. 1633, pp. 8–22 (1999).

[5] J. Bengtsson and W. Yi, "Timed automata: Semantics, algorithms and tools," in Lecture Notes on Concurrency and Petri Nets, Vol. 3098, pp. 87–124 (2004).

[6] G. Behrmann, A. David, and K.G. Larsen, "A Tutorial on Uppaal," Formal Methods for the Design of Real-Time Systems, Vol. 3185, pp. 200–236, (2004).

[7] T. Sekizawa, K. Okano, A. Ogawa, and S. Kusumoto, "Verification of a Control Program for a Line Tracing Robot using UPPAAL Considering General Aspects," International Journal of Informatics Society, Vol. 6, No. 2, pp. 79–87 (2014).

[8] F. Weitl and S. Nakajima, "Incremental Construction of counterexamples in Model Checking Web Documents," Proceedings of the 6th International Workshop on Automated Specification and Verification of Web Systems, EPiC Series, Vol. 18, pp. 61–75 (2010).

[9] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith, "Counterexample-guided abstraction refinement for symbolic model checking," Journal of the ACM, Vol. 50, No. 5, pp. 752–794 (2003).

[10] IPA, Fault diagnosis method for the large-scale and complex embedded system (in Japanese), https://www.ipa.go.jp/files/000045158.pdf, pp. 68–78 (2015). ⟨accessed June 05, 2016⟩.

[11] K. Okano and J. Kitamiti, Fault diagnosis method for the large-scale and complex embedded system (in Japanese), Software symposium 2015, http://sea.jp/ss2015/paper/ss2015_C1-2(2).pdf, (2015) ⟨accessed July 28, 2016⟩.

[12] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu, "Symbolic model checking without BDDs," Proceedings of 5th International Conference of Tools and Algorithms for Construction and Analysis of Systems (TACAS '99), pp. 193–207 (1999).

[13] E. Clarke, O. Grumberg, S. Jha, Y. Lu, and V. Helmut, "Counterexample-guided Abstraction Refinement," Proceedings of 12th International Conference of Computer Aided Verification (CAV '00), Vol. 1855, pp. 154–169 (2000).

[14] E. Clarke, A, Gupta, J. Kukula, and O. Strichman, "SAT based Abstraction-Refinement using ILP and Machine Learning Techniques," Proceedings of 14th International Conference of Computer Aided Verification (CAV '00), Vol. 2404, pp. 695–709 (2002).

[15] R. Alur, "Techniques for automatic verification of real-time systems," Ph.D. dissertation, Stanford University (1991).

[16] L. Ageto, P. Bouyer, A. Burgueño and K. G. Larsen, "Model-Checking via Reachability Testing for Timed Automata," Proceedings of 4th Conference of Tools and Algorithms for Construction and Analysis of Systems (TACAS '98), LNCS Vol. 1384, pp. 263–280 (1998).

[17] L. Ageto, P. Bouyer, A. Burgueño, and K. G. Larsen, "The Power of Reachability Testing for Timed Automata," Journal of the Theoretical Computer Science, Vol. 300, No. 1-3, pp. 411–475 (2003).

[18] B. Bordbar, R. Anane, and K. Okano, "An Evaluation Mechanism for QoS Management in Wireless Systems," International Workshop on Performance Modelling in Wired, Wireless, Mobile Networking and Computing 2005, Proceedings of International Conference on Parallel and Distributed Systems (ICPADS 2005), Vol. 2, pp. 150–154 (2005).

[19] B. Bordbar and K. Okano, "Verification of Timeliness QoS Properties in Multimedia Systems," Proceedings of 5th International Conference on Formal Engineering Methods (ICFEM 2003), LNCS Vol. 2885, pp. 523–540 (2003).

[20] "System trouble grounds JAL ' s domestic flights, affects travelers," The Japan Times News, http://www.japantimes.co.jp/news/2016/04/01/national/system-trouble-grounds-jals-domestic-flights-affects-travelers/, ⟨accessed Feb. 06, 2017⟩.

(Received October 10, 2016)
(Revised January 13, 2017)

Chikyu Yanagisawa received the BE degree from Shinshu University in 2015. He is a master course student of Graduate School of Science and Technology, Shinshu University. His current research interests include formal methods for software engineering.

Shinpei Ogata Shinpei Ogata is an assistant professor of Graduate School of Science and Technology in Shinshu University, Japan. He received a PhD from Shibaura Institute of Technology, Japan in 2012. His current research interests include model-driven engineering for information system development. He is a member of IEEE, ACM, IEICE, IPSJ.

Kozo Okano received his BE, ME, and PhD degrees in Information and Computer Sciences from Osaka University in 1990, 1992, and 1995, respectively. From 2002 to 2015, he was an Associate Professor at the Graduate School of Information Science and Technology of Osaka University. In 2002 and 2003, he was a visiting researcher at the Department of Computer Science of the University of Kent in Canterbury, and a visiting lecturer at the School of Computer Science of the University of Birmingham, respectively. Since 2015, he has been an Associate Professor at the Department of Computer Science and Engineering, Shinshu University. His current research interests include formal methods for software and information system design. He is a member of IEEE, IEICE, IPSJ.

Submission Guidance

About IJIS

International Journal of Informatics Society (ISSN 1883-4566) is published in one volume of three issues a year. One should be a member of Informatics Society for the submission of the article at least. A submission article is reviewed at least two reviewer. The online version of the journal is available at the following site: http://www.infsoc.org.

Aims and Scope of Informatics Society

The evolution of informatics heralds a new information society. It provides more convenience to our life. Informatics and technologies have been integrated by various fields. For example, mathematics, linguistics, logics, engineering, and new fields will join it. Especially, we are continuing to maintain an awareness of informatics and communication convergence. Informatics Society is the organization that tries to develop informatics and technologies with this convergence. International Journal of Informatics Society (IJIS) is the journal of Informatics Society.

Areas of interest include, but are not limited to:

- Internet of things (IoT)
- Smart cities, communities, spaces
- Big data, artificial intelligence
- Data science
- Computer supported cooperative work and groupware
  Intelligent transport system
- Distributed computing
- Multi-media communication
- Information systems Mobile computing
- Ubiquitous computing

Instruction to Authors

For detailed instructions please refer to the Authors Corner on our Web site, http://www.infsoc.org/.

Submission of manuscripts: There is no limitation of page count as full papers, each of which will be subject to a full review process. An electronic, PDF-based submission of papers is mandatory. Download and use the LaTeX2e or Microsoft Word sample IJIS formats.

http://www.infsoc.org/IJIS-Format.pdf

LaTeX2e

LaTeX2e files (ZIP) http://www.infsoc.org/template_IJIS.zip

Microsoft Word™

Sample document http://www.infsoc.org/sample_IJIS.doc

Please send the PDF file of your paper to secretariat@infsoc.org with the following information:

Title, Author: Name (Affiliation), Name (Affiliation), Corresponding Author. Address, Tel, Fax, E-mail:

Copyright

Publisher

Address: Informatics Laboratory, 3-41 Tsujimachi, Kitaku, Nagoya 462-0032, Japan

E-mail: secretariat@infsoc.org

CONTENTS