# International Journal of Informatics Society

04/11 Vol. 3 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 quality and value as a resource. Informatics also referred to as Information science, studies the structure, algorithms, behavior, and interactions of natural and artificial 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 the study of informatics 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

Kouji Yoshida
Guest Editor of the Seventh Issue of International Journal of Informatics Society

We are delighted to have the seventh and special of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Forth International Workshop on Informatics (IWIN2010), which was held in Edinburgh, Scotland, UK, Sept. 13-16, 2010. The workshop was held at Royal British Hotel. This workshop was the forth 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 at four technical sessions. The workshop was complete in success. It highlighted the latest research results in the area of networking, business systems, education systems, design methodology, groupware and social systems.

Each IWIN2010 paper was reviewed in terms of technical content and scientific rigor, novelty, originality and quality of presentation by at least two reviewers. From those reviews, 17 papers are selected for publication candidates of IJIS Journal. This seventh includes five papers of them. The selected papers have been reviewed from their original IWIN papers and accepted as publication of IJIS. The papers were improved based on reviewers' comments.

We hope that the issue would be of interest to many researchers as well as engineers and practitioners in this area.

We publish the journal in print as well as in an electronic form over the Internet. This way, the paper will be available on a global basis.

**Kouji Yoshida** is a professor at Shonan Institute of Technology, Japan. He had Ph.D. from Shizuoka University of Japan in 2001. He worked as a computer scientist in the Information Engineering Works at Mitsubishi Electric Corp from 1972 to 2001. He was a visiting lecturer at Nagoya Institute of Technology from 1996 to 1999. His major research includes Internet/Intranet, distance learning and information gathering on the Internet. He is a member of IPSJ and IEICE, respectively.

# A Correction Reflected Query Method of Database during Online Entry

Tsukasa Kudo†, Yui Takeda‡, Masahiko Ishino*, Kenji Saotome**, Kazuo Mutou***, and Nobuhiro Kataoka****

†Faculty of Comprehensive Informatics, Shizuoka Institute of Science and Technology, Japan
‡Mitsubishi Electric Information Systems Corporation, Japan
* Department of Management Information Science, Fukui University of Technology, Japan
** Hosei Business School of Innovation Management, Japan
*** Faculty of Science and Technology, Shizuoka Institute of Science and Technology, Japan
**** Department of Information Technology, Tokai University, Japan
kudo@cs.sist.ac.jp

***Abstract*** - The database of the mission-critical systems is updated with entry data by transaction processing, and are queried to make statistics and so on by batch processing generally. Such a batch processing had been executed at the overtime to avoid the data entry service time, because it occupied the database for hours. On the other hand, in recent years, the entry service time is being rapidly extended with the development of the Internet business. So, the methods to execute the both concurrently have been put to practical use. However, there are some cases that cannot be supported by only the conventional methods, because there are various kinds of database query and operation in the actual mission-critical system. In this paper, to support such the case, we propose a query method to query the database as of designated time reflecting the correction entered after the time. Moreover, we implemented this method into a mission-critical system, and confirmed the effect to reduce the overtime batch processing in the actual operation.

***Keywords***: temporal database, transaction time database, mission-critical system, query, integrity, batch processing

## 1 INTRODUCTION

In the mission-critical system such as the retail, the finance, the manufacture, because data are entered by many online terminals concurrently (hereinafter "online entry"), concurrency controls are executed by the transaction processing [5]. On the other hand, a great deal of data processing, such as periodic sum of entered data, is processed by the batch processing [5]. For example, in the retail system, sales information at stores is reflected into its database immediately by the transaction processing; on the other hand, the settlement of accounts is calculated by the batch processing. Here, the batch processing had been executed in night to avoid the time zone of the online entry, because it occupy the database for hours to process a great deal of data. However, in recent years, this time zone was expanded by the development of the internet business and so on. As a result, it often caused a problem that the batch processing didn't complete in the given time.

So, the method to maintain the integrity of query result of database even during the online entry had been implemented. For example, the multiversion concurrency control of database [2], by which the integrity of query result is maintained during the online entry, is used widely. Here, in the batch processing, because the restriction of the execution time is looser than the online entry, strict examinations of the entry data are executed. Therefore, error data is often found. If the batch processing is executed while online entry isn't executed, it can be executed again after the correction of error data. However, if batch processing is executed concurrently with online entry, the newly entered data is also reflected into the batch processing result. That is, the corrected result of designated time, the cash total sum of the day and so on for example, can't be provided.

For this problem, authors showed that the integrity of the snapshot of the bitemporal database can be maintained during the online entry in the actual mission-critical system, even in the case that error data were detected, by reflecting its correction into the query result [9]. Here, the bitemporal database is a kind of temporal database [7], [13], which manages both of the transaction time and the valid time. The former is the time that data is valid in the database; the latter is the time that data is valid in real world [4], [6], [11], [13]. And, its query target was the data at the designated valid time.

However, in the case of the settlement of accounts and so on, the processing target is the data that was online entered by the deadline time, which is the database status as of this time. And, if error data are detected, they have to be corrected while the processing. In this case, the multiversion concurrency control has the problem that does not support the reflection of data correction after the deadline time; the bitemporal database also has the problem not being suitable for such the system that the status of real world was not entered instantly.

Our goal in this paper is to provide the query method that maintains the integrity of query result with reflecting the data correction, even in the above-mentioned case. We summarized this and showed it in the title as "A Correction Reflected Query Method". For this purpose, we propose the correction query method, which uses the transaction time. We show that the corrected data is queried without influences of the online entry by this method. Moreover, we implemented this method into an actual mission-critical system, and confirmed the effect to reduce the overtime batch processing.

The reminder of this paper is organized as follows. In section 2, we show the problem to intend for, and in section 3,

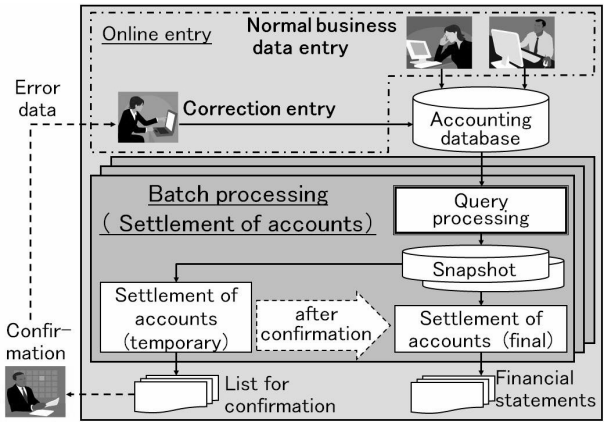



Figure 1: An example of batch processing constitution.

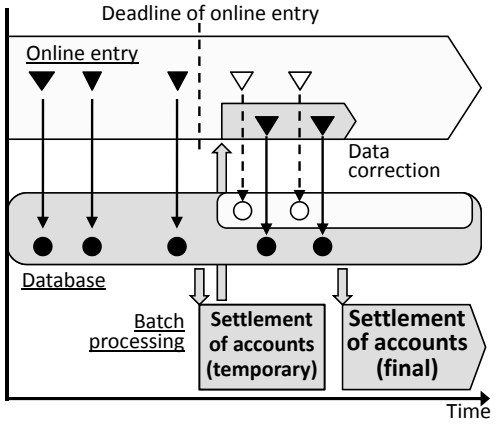



Figure 2: Settlement of accounts data by time series

we propose the query method to solve this problem. In section 4, we show an implementation case of this method in a mission-critical system, and in section 5, we evaluate the method based on the implementation result. Finally, we consider this method in section 6.

## 2 PROBLEM WITH BATCH PROCESSING

### 2.1 Constitution of Batch Processing

In the mission-critical system, a certain level integrity of online entered data is maintained by the integrity control of the database management system and the transaction processing, and by the checking function of the business application program. In this paper, we define the integrity as what the state of the real world is reflected in the database with validity and completeness [10]. By the way, the integrity confirmation with querying a large quantity of data needs to be executed by batch processing. For example, the calculations of total for the collation with the actual cash or the actual articles, or the consistency check among some tables and so on. So, in the batch processing, the first process is usually the integrity confirmation of its target data.

Figure 1 shows the example of the batch processing about the accounting system. Accounting data is accumulated in the database by the online entry, and the settlement of accounts processing is executed regularly. In this processing, temporary processing is executed first to prevent errors of the processing, in which various kinds of data check is done. And, when error data is detected, it is corrected by the online entry. In this way, after all confirmation is complete, final processing is executed to make the financial statements.

Here, the query processing in the batch processing (hereinafter "batch query processing") of Figure 1 has to be executed without undergoing influence of the online entry, though it is executed concurrently with the online entry. So, even if the correction data is entered by the online entry, it must be distinguished from the normal business entry data entered after the deadline time. Figure 2 shows the state of data of the settlement of accounts processing of Figure 1 by the time series. In figure 2, "▼" shows both of the online entry before the deadline time, and its correction entry; "●" shows the data corresponding to them. Also, "▽" shows the new online entry after the deadline time; "◯" shows the data corresponding to this. In the settlement of accounts processing, the temporary processing is executed for the confirmation about the data entered by the deadline time of Figure 2. And, the final settlement of accounts processing is executed after correction of the data error. Therefore, the target data of the settlement of accounts processing is the query result as of the the deadline time, in which only the correction entered after the time is reflected. That is, in Figure 2, only the data shown by "●" is the target for the final processing.

### 2.2 Problem about Conventional Database Query Method

We show the problem about the conventional database query method in the case of batch query processing accompanied by the data correction. In the multiversion concurrency control, the version of the database is managed with the time series. That is, the data entered after the deadline time for the correction cannot be distinguished from the normal business entry. Therefore, in the case shown in Figure 2, there is the problem that even the normal business entry data shown by "◯" become the processing target, too.

For this problem, we showed a solution utilizing the bitemporal database and confirmed that we could execute the batch processing even while the online entry in the actual mission-critical system [9]. In the bitemporal database, both histories of the valid time and the transaction time are managed, and the state of data, which once existed in the database, is accumulated as the records. That is, the both records of the state of the database and the real world are accumulated [4], [8]. For example, in the personnel management system of the company, the period that a person was in office for one duty position is shown with the valid time; on the other side, the period that its data was valid in the database is shown with the transaction time. Incidentally, the database that manages none of these times is called the snapshot database [12].

Figure 3 shows the application example of the bitemporal database to the travel expense checkout of the accounting system, in which correction data is queried on the condition that the deadline time is April 20th. In Figure 3, "$[V_a, V_d)$" shows

(1) Case of normal query

| ID | Va | Vd | Ta | Td | Amount | Result |
|---|---|---|---|---|---|---|
| 001 | 4/19 | 4/20 | 4/20 | 4/21 | 1,000 | |
| 001 | 4/19 | 4/20 | 4/21 | now | 1,500 | ● |
| 002 | 4/20 | 4/21 | 4/21 | now | 3,000 | |

(2) Case of wrong query

| ID | Va | Vd | Ta | Td | Amount | Result |
|---|---|---|---|---|---|---|
| 001 | 4/19 | 4/20 | 4/20 | 4/21 | 1,000 | |
| 001 | 4/19 | 4/20 | 4/21 | now | 1,500 | ● |
| 002 | 4/19 | 4/20 | 4/21 | now | 3,000 | ● |

Figure 3: Query of correction data by bitemporal database

the period of the valid time, i.e. one business trip period, and "$[T_a, T_d)$" shows the period of the transaction time, i.e. the period that its slip data was valid in the database of the system. Incidentally, the time is expressed by the unit of a day. And, "●" of the column " Result" shows the queried data for the query condition explained below. On April 20th, the data $ID = 001$ was entered, and on April 21st, the correction entry of the data $ID = 001$ and the new entry of the data $ID = 002$ was done. Here, when making a travel expense checkout data aggregate $D = \{d\}$ and designating the valid time $t_v$ and the transaction $t_t$, the following data is queried as the snapshot as of the above-mentioned time.

$$D_1 = \{d | d \in D, t_v \in [d[V_a], d[V_d]) \wedge t_t \in [d[T_a], d[T_d])\} \quad (1)$$

Here, $d[V_a]$ shows the instance of the attribute $V_a$ in $d$, and the others are same, too.

Therefore, as shown in (1) of Figure 3, when time were designated as $t_v = April\ 19th$ and $t_t = April\ 21st$, the data $ID = 001$ after correction is queried; the data $ID = 002$ is not queried. Here, the time "$now$" of $T_d$ shows the corresponding data is valid at the time to query [1], [14].

However, in the actual business, the state of the real world isn't always reflected into the database immediately. (2) of Figure 3 shows the case that the entry of the travel expense checkout has been late. Though the valid time period of the trip is $[4/19, 4/20)$, its data was entered on April 21th. In this case, there is a problem that the query result includes the data $ID = 002$, because it satisfies the condition of equation (1). But nevertheless it is the normal business entry data after the deadline time.

Moreover, there is the problem that some businesses don't need to manage the valid time. For example, the slips of the purchase and the payment of the accounting system are managed by the system, so their valid time as for the real world isn't managed usually. That is, the split table of the database doesn't need to take the composition of bitemporal database.

# 3 PROPOSAL OF QUERY METHOD TO REFLECT DATA CORRECTION

We propose a query method,"correction query", for the problem shown in section 2.

## 3.1 Correction Query

The correction query is the query method which result of time $t_1$ reflects only its correction entered by time $t_2$. We call the time $t_1$ "query time", and $t_2$ "correction query tim", and it becomes $t_1 < t_2$. Incidentally, in the case of Figure 2, $t_1$ corresponds to the deadline, and $t_2$ corresponds to the start time of settlement of accounts (final).

The correction query deals with the database that manages the transaction time, i.e. the transaction time database. The relation [3] of the transaction time database $R$ is expressed as following.

$$R(K, T, A) \quad (2)$$

We show each attribute as follows.

- $K = \{K_1, ..., K_m\}$
  This expresses the set of attributes constituting the primary key of the snapshot queried by the designated transaction time.

- $T = \{T_a, T_d\}$
  This expresses the time period attribute of the transaction time, which is generated by system and isn't made public to the users. Here, $T_a$ shows the time that the data was added to the database (hereinafter "addition time"), and $T_d$ shows the time that the data was logically deleted from the database (hereinafter "deletion time"). As long as the data hasn't been deleted yet, the instance of attribute $T_d$ is expressed by the above-mentioned "$now$".

- $A = \{A_1, ..., A_n\}$
  This expresses the other attributes.

We can query the snapshot at any designated transaction time, which is the state of the database at the time. When making the designated time t, the relation of this snapshot is expressed by the following equation.

$$Q(t) = \{q | q \in R \wedge q[T_a] \leq t \wedge t < q[T_d]\} \quad (3)$$

Here, $q[T_a]$ shows the instance of the attribute $T_a$ of $q$, and $q[T_d]$ is similar, too. In the correction query, both of the snapshot at above-mentioned $t_1$ and $t_2$ are queried. And, the correction query result is the data that reflected the corrections entered by the time $t_2$ into the snapshot of $t_1$.

The relation of the correction query for $R$ is expressed by the union of the following $S_1$ and $S_2$, i.e. $S = S_1 \cup S_2$. Here, $S_1$ shows the data not being changed or deleted between $t_1$ and $t_2$. So, the correction query result is the same as the snapshot of $t_1$. The corresponding data is expressed by the following equation, because it exists at the both of $t_1$ and $t_2$.

$$S_1 = \{s | s \in Q(t_1) \wedge s \in Q(t_2)\} \quad (4)$$

On the other hand, $S_2$ shows the data being changed or deleted between $t_1$ and $t_2$. So, the correction query result is the data after the change or delete. As for the change, it is expressed by the following equation, because the data before and after change is connected by the primary key attributes

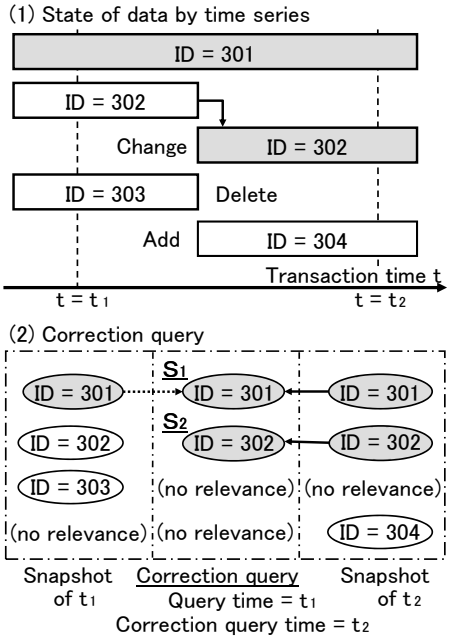



Figure 4: An example of correction query

$r[K]$ and $s[K]$. And, by this definition, the data deleted by the time $t_2$ isn't the target of the correction query.

$$S_2 = \{s | s \notin Q(t_1) \wedge \in Q(t_2) \wedge \exists r \in Q(t_1); r[K] = s[K]\} \quad (5)$$

Incidentally, the data of the correction query result is the subset of the snapshot $Q(t_2)$, which is entered by the usual transaction, so the consistency of the data is maintained.

Figure 4 shows the example of the correction query, of which query time is the transaction time $t = t_1$ and correction query time is $t = t_2$. In the entered data $ID = 301, 302\ and\ 303$, $ID = 302$ was changed, $ID = 303$ was deleted, on the other hand $ID = 304$ was added newly after the time $t_1$. (2) of Figure 4 shows the correction query result for these data. First, the data $ID = 301$ is queried based on the equation (4); $ID = 302$ after correction is queried based on (5). Second, $ID = 303$ that was deleted and $ID = 304$ that was newly added don't become the target.

## 3.2 Effect of Correction Query

We show that the problem shown in section 2.2 can be solved by the correction query. Figure 5 shows the application example of the correction query to the settlement of accounts processing, in the case of Figure 2. Here, we show the change of data of database by the time series like (1) of Figure 4. The temporary processing of the settlement of account had been executed for the data entered by the deadline time, and to correct the data, the change of $ID = 302$ and the deletion of $ID = 303$ were executed by the online entry based on the confirmation result of the temporary processing. On the one hand, the online entry of the normal business data were continued after the deadline time as same as before the time. In this example, the result of correction query, of which

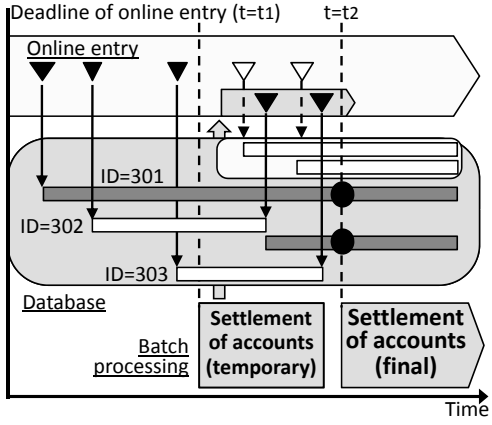



Figure 5: Correction query for settlement of accounts

the query time is the deadline time $t = t_1$ and the correction query time is the start time of the "final" settlement of account processing $t = t_2$, is the data shown by "●" in Figure 5. That is, the state of database as of the deadline time with reflecting the corrections entered after the time can be queried even during the normal business online entry without undergoing influence of this.

# 4 APPLICATION TO A MISSION-CRITICAL SYSTEM

In this section, we show the application result of the correction query to a mission-critical system, the local government system.

## 4.1 Overview of Local Government System

The local government system is a mission-critical system for the public administration business of the local government like a city hall. And, as shown in Figure 6, it consisted of various kinds of subsystems to assist the local government business. They were classified by business contents as follows.

(a) **Subsystems about Resident information**
They were used for the business, such as management and certificate of the residents who live in the city.

(b) **Subsystems about Local Tax**
They were used for the business of the local tax, such as levy and certificate about tax.

(c) **Subsystems about Welfare**
They were used for the business of welfare, such as qualification management, levy and grant.

(d) **Subsystems about City Office**
They were used for the business of the office work of local government, such as personnel management, salary computation and financial accounting.

In each subsystem, the reports were accepted at the report windows and online entered to accumulate in the database. And, the processing to query a large quantity of data was executed as the batch processing regularly or at any time. In the

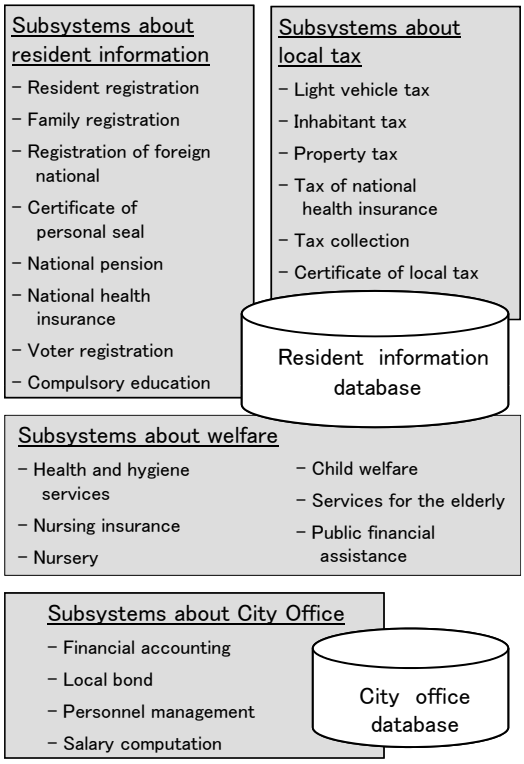



Figure 6: Composition of local government system

batch processing, the state of database as of the designated time was often queried. We show the example of batch processing like this below.

- **Population statistics:** based on the resident transfer reports, the statistics of such as the population and the number of households was made as of the end time of the first day of every month.
- **Taxation processing:** based on the reports about the local tax, the taxation processing was executed. It used the state of database as of the individually designated time.
- **Settlement of accounts processing:** based on the data of the income and the outlay, settlement of accounts processing was executed with the state of database as of the end time of every day, month and year.

## 4.2 Implementation of Correction Query

As shown in section 3.2, the correction query intends to the transaction time database. We used the commercial relational database and added the attributes of the addition time and the deletion time to each table to compose a transaction time database, depending on the necessity of the target business. Here, since transaction time is used as one of primary key attributes of the database, the unit of the transaction time had to be decided based on the frequency of data entry. In this system, data were entered from the terminals, and the data entry took several seconds at least. So, we made the unit of the transaction time 1 second. Incidentally, we made the attribute

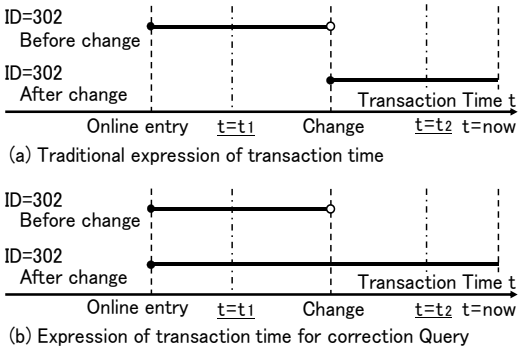



Figure 7: Implementation of transaction time

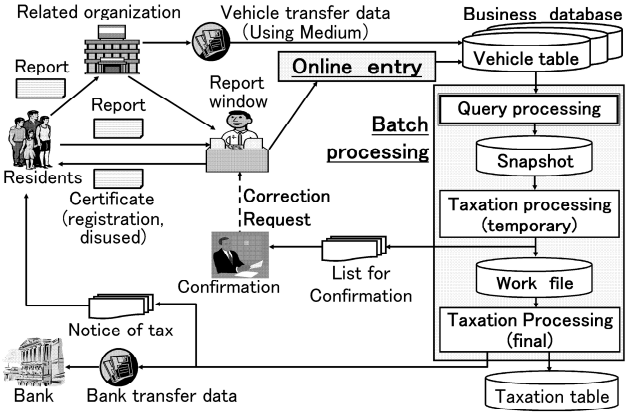



Figure 8: Dataflow of light vehicle tax business

of the transaction time the closed information in users including the records as for it, so users could query only the latest state at the query time.

As for the change records with the transaction time, the data after change was conventionally expressed in the form, of which addition time was the changed time as shown in (a) of Figure 7. In the implementing of a correction query, it was necessary to connect the data before and after correction. So, the query processing became complicated if the conventional expression was used. To solve this problem, we implemented the transaction time with the expression, in which the addition time of the data after change is the time that the data was added first, as shown in (b) of Figure 7. Incidentally, in this expression, the deletion time becomes the primary key attribute; though, in the conventional expression, the addition time is the primary key attribute.

## 4.3 Composition of Subsystem for Business

As the example of the business system, to which we applied the correction query, we show the light vehicle tax subsystem that is one of the subsystems about the local tax. The light vehicle is taxed on the light vehicles, which is owned by the residents as of April 1st that is the basic date. And, the taxation processing is executed based on the data reported by the residents.

Figure 8 shows the dataflow of the light vehicle tax busi-

(1) Data of vehicle table on 5/6

| ID | Owner | Ta | Td | 5/6 |
|---|---|---|---|---|
| 001 | Keiji, T. | 5/6 | now | ● |
| 002 | Jouto, J. | 5/6 | now | ● |
| 003 | Haisha, S. | 5/6 | now | ● |

(2) Data of vehicle table on 5/7

| ID | Owner | Ta | Td | 5/6 | 5/7 | S |
|---|---|---|---|---|---|---|
| 001 | Keiji, T. | 5/6 | now | ● | ● | ● |
| 002 | Jouto, J. | 5/6 | 5/7 | ● | | |
| 002 | Jouto, J. | 5/6 | now | | ● | ● |
| 003 | Haisha, S. | 5/6 | 5/7 | ● | | |
| 004 | Tuika, F. | 5/7 | now | | ● | |

Figure 9: Query result of vehicle table with correction query

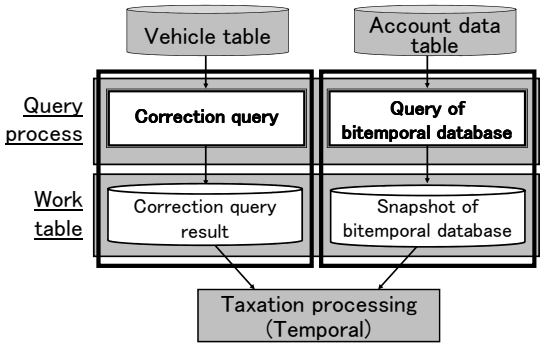



Figure 10: Combination with conventional query method

ness. The acquisition reports of the light vehicles should be reported within 15 days; the disused and transfer reports should be reported within 30 days. However, these reports are accepted in the related organizations such as the Light Motor Vehicle Inspection Organization, the Land Transport Bureau and the light vehicle stores in addition to the report windows of the local government. The data accepted at the related organizations were delivered to the local government with paper reports for online entry or with mediums for lump-sum entry. For such operation, it often takes time to reflect the transfer data of the real world into the vehicle table of the system. Therefore, the taxation processing was executed for the data entered by the deadline time, and thereafter, tax correction processing was executed monthly for the data newly entered by the corresponding deadline time.

Online entry at the report windows could not be suspended during business hours, because the light vehicles license plate issue certificates or the disuse report receipt certificates had to be published immediately reflecting the reported data. On the other hand, the taxation processing and the tax correction processing were executed by the batch processing to make the tax payment notices to the residents and the account transfer requests to the financial institutions. So, to prevent the taxation error, the checklist and the statistics documents for the confirmation were made by the temporary processing first. And, when the data error was detected, it was corrected by the online entry. After this confirmation and correction, the final processing was executed.

So, the target data for the final processing was the state of database as of the deadline time, in which only the corrections after the time were reflected. Figure 9 shows the taxation processing case, of which the deadline time was May 6th and the execution time was May 7th. We show the state of database as of May 6th in (1) of Figure 9, and the data entered by this time was the target for the processing. We show the state as of May 7th in (2) of Figure 9, in which the change of the vehicle $ID = 002$, deletion of $ID = 003$ and addition of $ID = 004$ were reflected. Here, the transaction time of $ID = 002$ was implemented with the expression shown in (b) of Figure 7. In Figure 9, "●" of column "$5/6$" shows the snapshot data of May 6th; column "$5/7$" shows the snapshot data of May 7th; column "$S$" shows the correction query result, of which the query time was May 6th and the correction query time was May 7th.

As shown in the column "$S$", the correction query result was the snapshot at May 6th, in which only the correction entered by May 7th were reflected. So, the addition data $ID = 004$ was not included.

## 4.4 Combination with Other Query Methods

In the actual mission-critical systems, it is necessary to query the database in a wide range of conditions. For example, the light vehicle tax was paid by the tax notice or the bank transfer. Here, as for the bank transfer, it was requested by the resident with its transfer period. So, the data for the bank transfer needed to have the valid time attribute, and we had to implement it as a table of bitemporal database. On the other hand, we queried the master table by the multiversion concurrency control, because it was the table of the snapshot database without managing the transactiontime. In this way, as the constitution of the table was different with the condition of the target business, it was necessary to combine various kinds of query results to make the final outputs such as the financial statements and so on.

In the application system, to solve this problem, we composed temporary files of the batch processing by the work tables, which are usually composed by the sequential access method (SAM) file. And, in the whole batch processing, we processed data by the query function of the database, to simplify each individual query procedure and maintain its performance. For example, as for the above-mentioned bank transfer, we queried the vehicle table by the correction query and queried the account data table by the snapshot of the bitemporal database on the other hand. Afterward, as shown in Figure 10, we combined these results by utilizing the query function of the database in the temporary processing executed next.

# 5 EVALUATION

## 5.1 Evaluation about Systems Operation

In the application system, online entry could not be suspended during business hours, because the certificates reflecting the entry data had to be published immediately as shown in Figure 8. On the other hand, conventionally, the batch processing using the data that took time until its entry or was not including the valid time data, could not be executed concurrently with the online entry. So, it had to be executed at the overtime like "batch processing 3" or "4" of (1) of Figure 11.

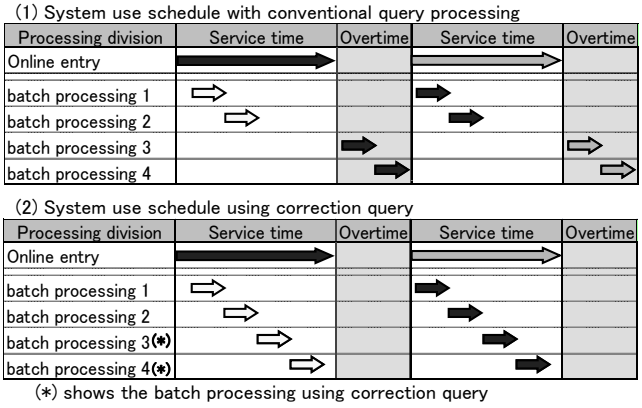



Figure 11: Reduction of overtime batch processing

Table 1: Application rate of correction query.

| No | business | total | $T_a$ | $T_d$ |
|---|---|---|---|---|
| (a) | resident | 36 | 32(89%) | 18(50%) |
| (b) | tax | 72 | 58(81%) | 31(43%) |
| (c) | welfare | 40 | 37(93%) | 24(60%) |
| (d) | office | 63 | 42(67%) | 8(13%) |
| | sum | 211 | 169(80%) | 81(38%) |

In contrast, as the batch processing like this became able to be executed concurrently with the online entry by utilizing the correction query in the application system, it could be executed during the business hours on the next day as shown in (2) of figure 11.

As a result, all the batch processing to query database were executed during the business hours, and the overtime work could be reduced. Incidentally, the confirmation and the correction entry were also executed at the same time.

## 5.2 Evaluation about Coverage

Table 1 shows the application table number and rate of the correction query in the application system. We added the addition time $T_a$ to the tables to manage the records with the transaction time; and we added the deletion time $T_d$ to the tables for the correction in addition to $T_a$. Therefore, the rate of the column $T_d$ of Table 1 is the application rate of the correction query. Here, the row number is the same as the subsystem classification number shown in section 4.1. And, it targets only the transaction table, so it excludes the following tables: the master tables such as the parameter table and the code table; the temporary data tables such as the work table; the derivation datas table such as the total sum.

Here, the table rate to have the addition time is 80%; the table rate to have the deletion time is 38%. That is, the correction query was applied to about 50% of the tables that manage the records. Here, the application rate depended on the subsystem. It was applied to only the 13% tables in the subsystems about the city office; on the other hand, it was applied to from the 43% to 60% tables in the other subsystems.

As shown in section 4.4, the queries with a wide range of conditions were necessary in the actual system operation. Table 2 shows the evaluation of the query method for these query condition. In addition, it shows the kind of the database corresponding to the query method, too. In table 2, "◯" shows that batch query processing can be executed during the online entry; "×" shows that there is the problem to execute the processing. The conventional query methods, i.e. the multiversion concurrency control and the snapshot of bitemporal database, have the problem for the query condition as of the designated transaction time with correction. By the correction query, we could execute the batch query processing even in the above-mentioned condition.

On the one hand, the multiversion concurrency control is necessary to query the tables of the snapshot database; the snapshot of the bitemporal database is necessary to query as of the designated valid time reflecting correction entry. Therefore, it is necessary to make the batch processing such a structure that can combine these query results for making the final output as shown in Figure 10.

## 5.3 Evaluation about Implementation

For the correction query was implemented in the query processing as shown in Figure 4, the online entry processing was same as before. And, as for the database table, we could implement the correction query easily, because we implemented the transaction time using the expression shown in (b) of Figure 7. For example, the correction query shown in Figure 9 could be executed by the following simple SQL.

$$select\ ID,\ Owner,\ T_a,\ T_d\ from\ Vehicle\ Table \\ where\ T_a \leq 5/6\ and\ T_d = now \quad (6)$$

In addition, there is the thing that plural history data are queried if $T_d$ is designated as the past, not now. In this case, the history data that has earliest $T_d$ becomes the query target. However, in the actual system operation, $T_d$ was usually designated at "now", that is the time when the batch processing was executed. Therefore, such operation was unnecessary.

As shown in section 5.2, it is necessary to query the database in a wide range of conditions corresponding with the business needs and to combine these results to make the final output. For this problem, in the application system, we took the constitution of batch processing, in which we used the database work table instead of the SAM file as shown in Figure 10. As a result, we could combine them easier by using SQL function. By adopting the above-mentioned constitutions, in the application case to the local government of a population of about 40 thousand, the performance deterioration of query and online entry didn't occur comparing with the conventional method.

# 6 CONSIDERLATION

By the correction query, the problem of conventional query method, that is the query condition as of the designated transaction time with correction during online entry, could be solved. As the result of having applied it to an actual mission-critical

Table 2: Evaluation of query method with query time condition.

| Target database | Query method | As of query start time | As of designated valid time with correction | As of designated transaction time with correction |
|---|---|---|---|---|
| Multiversion concurrency control | Snapshot database | ○ | × | × |
| Snapshot | Bitemporal database | ○ | ○ | × |
| **Correction Query** | Transaction time database | ○ | × | ◯ |

system, we confirmed the effect that the overtime batch processing to query the database became unnecessary. In recent years, such the operation of mission-critical systems is increasing because of the rapid development of the internet business such as the electronic commerce, the electronic government and so on, in which users directly enter their data to the systems and the online entry cannot be suspended. So, the batch processing has to be executed in the online entry service time. Therefore, we consider that the correction query is effective, by which we can execute the batch query processing without suspending the online entry.

In the actual mission-critical systems, a wide range of data management and data query are necessary based on the business needs. So, it is necessary that the database can be queried by plural methods, and the final output has to be made by combining these query results. In particular, querying the database containing records is complicated. So, the method to meintain query performance is important. Therefore, we consider that our proposal method is effective: the implementation of the transaction time by the proposed expression; the method using the work table to process the data step by step by utilizing the database function to simplify each query and combine their results to make final output.

The application rate of the correction query deeply depends on the subsystems as shown Table 1. Excepting the subsystems about the city office, because the subsystems deal the data based on the reports of real world, the wrong entry data has to be corrected as shown by the light vehicle business in section 4.3. On the other hand, as for the subsystems about the city office, the reports were often omitted in the business. For example, the slips of the financial accounting subsystem were managed in the database. So, when an approval slip was wrong, the new split was published for its adjustment. Therefore, we consider that the correction query is effective for the system that needs the internal correction to consistent its data with the state of the real world.

## 7 CONCLUSION

As for the system that takes time until the state of real world is reflected into its database, the batch processing is often executed using the data entered by the designated time. However, in this case, when the entry data is corrected, the integrity of the query result of the batch processing cannot be maintained during the online entry by the conventional query method. In this paper, we propose the correction query to query the data entered by the designated time with reflecting the corrections entered after the time. Moreover, we applied this to the mission-critical system and confirmed the effect to reduce the overtime batch processing in the actual systems operation.

Future study will focus on the development of the method, by which database can be updated with a large quantity of data in a lump during the online entry.

## REFERENCES

[1] L. Bækgaard and L. Mark, “Incremental Computation of Time-Varying Query Expressions,” IEEE Trans. knowledge and Data Eng., Vol. 7, No. 4, pp. 583–590 (1995).

[2] P. A. Bernstein and N. Goodman, “Multiversion Concurrency Control-Theory and Algorithms,” ACM Trans. on Database Sys., Vol. 8, No. 4, pp. 465–483 (1983).

[3] E. F. Codd, “Extending the database relational model to capture more meaning,” ACM Trans. on Database Sys., Vol. 4, No. 4, pp. 397–434 (1979).

[4] N. Edelweiss, P. N. Hübler, M. M. Moro and G. Demartini, “A Temporal Database Management System Implemented on top of a Conventional Database,” Proc. International Conference of the Chilean Computer Science Society, pp. 58–67 (2000).

[5] J. Gray and A. Reuter, “Transaction Processing: Concept and Techniques,” Morgan Kaufmann, San Francisco (1992).

[6] C. S. Jensen, L. Mark and N. Roussopoulos, “Incremental Implementation Model for Relational Database with Transaction Time,” IEEE Trans. knowledge and Data Eng., Vol. 3, No. 4, pp. 461–473 (1991).

[7] C. S. Jensen, C. E. Dyreson and et al., “The Consensus Glossary of Temporal Database Concept – February 1998 Version, Temporal Database: Research and Practice.” (the book grow out of a Dagstuhl Seminar, June 23–27, 1997), Lecture Notes in Computer Science 1399, Springer–Verlag, pp. 367–405 (1998).

[8] C. S. Jensen and R. T. Snodgass, “Temporal Data Management,” IEEE Trans. knowledge and Data Eng., Vol. 11, No. 1, pp. 36–44 (1999).

[9] T. Kudou, M. Ishino, K. Saotome, N. Kataoka and T. Mizuno, “Implementation of Integrity Maintenance

Method of Query Result by Bitemporal Database," International Journal of Infomatics Society, Vol. 1, No. 1, pp. 16–26 (2009).
[10] A. Motro, "Integrity = validity + completeness," ACM Trans. on Database Sys., Vol. 14, No. 4, pp. 480–502 (1989).
[11] G. Özsoyoğlu and R. T. Snodgrass, "Temporal and Real-Time Databases, A survey," IEEE Trans. knowledge and Data Eng., Vol. 7, No. 4, pp. 513–532 (1995).
[12] L. Shrira and H. Xu, "SNAP: Efficient Snapshots for Back-in-Time Execution," Proc. 21st International Conference on Data Engineering., pp. 434–445 (2005).
[13] R. Snodgrass and I. Ahn, "Temporal Databases," IEEE COMPUTER, Vol. 19, No. 9, pp. 35–42 (1986).
[14] B. Stantic, J. Thornton and A. Sattar, "A Novel Approach to Model NOW in Temporal Databases," Proc. 10th International Symposium on Temporal Representation and Reasoning and Fourth International Conference on Temporal Logic, pp. 174–180 (2003).

(Received August 24, 2010)
(Revised April 24, 2011)

**Tsukasa Kudo** received the M. Eng. from Hokkaido University in 1980 and the Dr. Eng. in industrial science and engineering from Shizuoka University, Japan, in 2008. In 1980, he joined Mitsubishi Electric Corp. He was a researcher of parallel computer architecture, an engineer of application packaged software and business information systems. Since 2010 he is a Professor of Shizuoka Institute of Science and Technology. Now, his research interests include database application and software engineering. He is a member of IEIEC, Information Processing Society of Japan and The Society of Project Management.

**Yui Takeda** received the B.E. from Keio University, Japan in 1987. In 1987, she joined Mitsubishi Electric Corp. She was an engineer of artificial intelligence and application software. Since 2001, she joined Mitsubishi Electric Information Systems Corp. Now, she manages intellectual property rights.

**Masahiko Ishino** received the master's degree in science and technology from Keio University in 1979 and received the Ph.D. degree in industrial science and engineering from graduate school of Science and technology of Shizuoka University, Japan, in 2007. In 1979, he joined Mitsubishi Electric Corp. Since 2009, he is Professor of Fukui University of Technology. Now, His research interests include Management Information Systems, Ubiquitous Systems, Application Systems of Datamining, and Information Security Systems. He is a member of Information Processing Society of Japan, Japan Industrial Management Association, Japan Society for Management Information.

**Kenji Saotome** received the B.E. from the Osaka University, Japan in 1979, and the Dr.Eng in Information Engineering from the Shizuoka University, Japan in 2008. From 1979 to 2007, he was with Mitsubishi Electric, Japan. Since 2004, he has been a professor of Hosei business school of innovation management. His current research areas include LDAP directory applications and single sign-on system. He is a member of the Information Processing Society of Japan.

**Kazuo Mutou** received the master's degree in precision engineering from Yamanashi University in 1982 and received the Ph.D. degree in mechanical system engineering from graduate school of Science and technology of Tokyo University of Agriculture and Technology in 1993, in 1982, he was with Polytechnic University. Since 2008, he is Assistant Professor of Shizuoka Institute of Science and Technology. Now, His research interests include CAD/CAE/CAM/CAT Systems, MES Systems, and Digital Manufacturing Systems, etc. He is a fellow of Society of Automotive Engineers of Japan and a member of the Japan Society for Precision Engineering, etc.

**Nobuhiro Kataoka** received the Ph.D. in information science from Tohoku University. Since he jointed Mitsubishi Electric Corporation he has been engaged development of software engineering, and computer system design. He is currently a professor at School of Information Technology and Electronics Tokai University in Japan. His research interests is modeling for Information system development.

# A Model Abstraction Technique for Probabilistic Real-Time Systems Based on CEGAR for Timed Automata

Takeshi Nagaoka†, Akihiko Ito†, Toshiaki Tanaka†, Kozo Okano†, and Shinji Kusumoto†

†Graduate School of Information Science and Technology, Osaka University
Yamadaoka 1–5, Suita City, Osaka, 565-0871, Japan

***Abstract*** - Model checking techniques are considered as promising techniques for verification of information systems due to their ability of exhaustive checking. Well-known state explosion, however, might occur in model checking of large systems. Such explosion severely limits the scalability of model checking. In order to avoid it, several abstraction techniques have been proposed. Some of them are based on CounterExample-Guided Abstraction Refinement (CEGAR) technique proposed by E. Clarke *et al.*

This paper proposes a reachability analysis technique for probabilistic timed automata. In the technique, we abstract time attributes of probabilistic timed automata by applying our abstraction refinement technique for timed automata proposed in our previous work. Then, we apply probabilistic model checking to the generated abstract model which is just a markov decision process (MDP) with no time attributes. This paper also provides some experimental results on applying our method to IEEE 1394, FireWire protocol. Experimental results show our algorithm can reduce the number of states and total execution time dramatically compared to one of existing approaches.

***Keywords***: Probabilistic Timed Automaton, CEGAR, Model Checking, Real-time System, Formal Verification

## 1 INTRODUCTION

Model checking[1] techniques are considered as promising techniques for verification of information systems due to their ability of exhaustive checking. For verification of real-time systems such as embedded systems, timed automata are often used. On the other hand, probabilistic model checking[2]–[4] can evaluate performance, dependability and stability of information processing systems with random behaviors. In recent years, probabilistic models with real-time behaviors, called probabilistic timed automata (PTA) attract attentions. As well as traditional model checking techniques, however, state explosion is thought to be a major hurdle for verification of probabilistic timed automata.

Clarke *et al.* proposed an abstraction technique called CEGAR (CounterExample-Guided Abstraction Refinement)[5] shown in Fig. 1. In the CEGAR technique, we use a counter example (CE) produced by a model checker as a guide to refine abstracted models. A general CEGAR technique consists of several steps. First, it abstracts the original model (the obtained model is called abstract model) and performs model checking on the abstract model. Next, if a CE is found, it checks whether the CE is feasible on the concrete model or not. If the CE is spurious, it refines the abstract model. The last step is repeated until the valid output is obtained. In the CEGAR loop, an abstract model must satisfy the following property: if the abstract model satisfies a given specification, the concrete model also satisfies it.

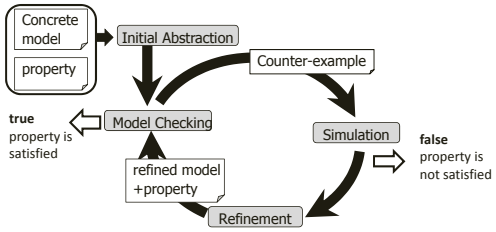



Figure 1: A General CEGAR Technique

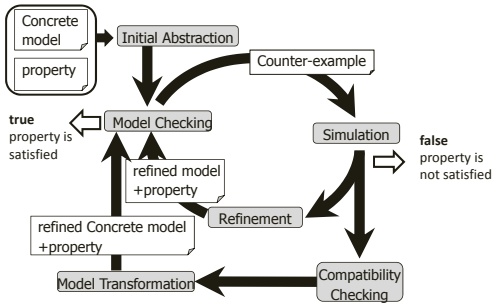



Figure 2: Our CEGAR Technique for Reachability Analysis of a Probabilistic Timed Automaton

In Paper[6], we have proposed an abstraction algorithm for timed automata based on CEGAR. In this algorithm, we generate finite transition systems as abstract models where all time attributes are removed. The refinement modifies the transition relations of the abstract model so that the model behaves correctly even if we don't consider the clock constraints.

This paper proposes a reachability analysis technique for probabilistic timed automata. In the technique, we abstract time attributes of probabilistic timed automata by applying our abstraction technique for timed automata proposed in Paper[6]. Then, we apply probabilistic model checking to the generated abstract model which is just a markov decision process (MDP) with no time attributes. The probabilistic model checking algorithm calculates a summation of occurrence probability of all paths which reach to a target state for reachability analysis. For probabilistic timed automata, however, we have to consider required clock constraints for such paths, and choose the paths whose required constraints are compatible. Since our abstract model does not consider the clock

constraints, we add a new flow where we check whether all paths used for probability calculation are compatible. Also, if they are not compatible, we transform the model so that we do not accept such incompatible paths simultaneously. The proposed procedure for the probabilistic timed automata is shown in Fig. 2.

This paper also provides some experimental results on applying our method to some examples. Experimental results show our algorithm can reduce the number of states and total execution time dramatically compared to one of existing approaches.

Several papers including Paper[3] have proposed probabilistic model checking algorithms. These algorithms, however, don't provide CEs when properties are not satisfied. Our proposed method provides a CE as a set of paths based on $k$-shortest paths search. This is a major contribution of our method. The proposed method also performs model checking considering compatibility problem. Few approaches resolve the compatibility problem. Paper [16] resolves the compatibility problem in a similar way to us. It, however, uses another approach (, which is based on a natural technique called predicate abstraction of clocks constraints) to abstract the models and the paper doesn't perform evaluation while our approach uses a quite simple abstraction technique, which remeves all of clock attributes, and this paper also shows the efficiency via performing experiments.

The organization of the rest paper is as follows. Sec.2 provides some definitions and lemmas as preliminaries. Sec.3 describes our proposed abstraction technique for the probabilistic timed automaton. Sec.4 gives some experimental results. Finally, Sec.5 concludes the paper and gives future works.

## 2 PRELIMINARY

This section gives some definitions about models used in this paper and also describes a general CEGAR technique.

### 2.1 Clock and Zone

Let $C$ be a finite set of clock variables which take non-negative real values ($\mathbb{R}_{\geq 0}$). A map $\nu : C \to \mathbb{R}_{\geq 0}$ is called a clock assignment. The set of all clock assignments is denoted by $\mathbb{R}^C_{\geq 0}$. For any $\nu \in \mathbb{R}^C_{\geq 0}$ and $d \in \mathbb{R}_{\geq 0}$ we use $(\nu + d)$ to denote the clock assignment defined as $(\nu+d)(x) = \nu(x)+d$ for all $x \in C$. Also, we use $r(\nu)$ to denote the clock assignment obtained from $\nu$ by resetting all of the clocks in $r \subseteq C$ to zero.

**Definition 2.1** (Differential Inequalities on $C$)**.** Syntax and semantics of a differential inequality $E$ on a finite set $C$ of clocks is given as follows:
$E ::= x - y \sim a \mid x \sim a$,
where $x, y \in C$, $a$ is a literal of a real number constant, and $\sim \in \{\leq, \geq, <, >\}$. Semantics of a differential inequality is the same as the ordinal inequality.

**Definition 2.2** (Clock Constraints on $C$)**.** Clock constraints $c(C)$ on a finite set $C$ of clocks is defined as follows:
A differential inequality $in$ on $C$ is an element of $c(C)$.

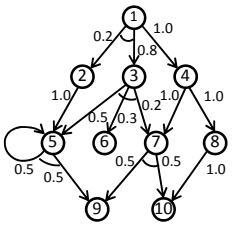



Figure 3: An Example of an MDP

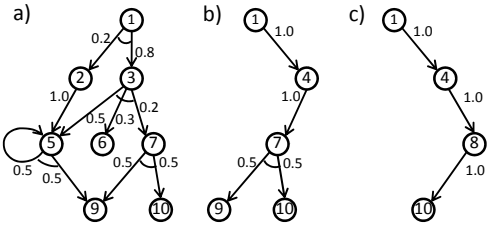



Figure 4: Examples of Adversaries

Let $in_1$ and $in_2$ be elements of $c(C)$, $in_1 \wedge in_2$ is also an element of $c(C)$.

A zone $D \in c(C)$ is described as a product of finite differential inequalities on clock set $C$, which represents a set of clock assignments that satisfy all the inequalities. In this paper, we treat a zone $D$ as a set of clock assignments $\nu \in \mathbb{R}^C_{\geq 0}$ (For a zone $D$, $\nu \in D$ means the assignment $\nu$ satisfies all the inequalities in $D$).

### 2.2 Probability Distribution

A discrete probability distribution on a finite set $Q$ is given as the function $\mu : Q \to [0, 1]$ such that $\Sigma_{q \in Q}\mu(q) = 1$. Also, $support(\mu)$ is a subset of $Q$ such that $\forall q \in support(\mu).\mu(q) > 0$ holds.

### 2.3 Markov Decision Process

A Markov Decision Process (MDP)[7] is a markov chain with non-deterministic choices.

**Definition 2.3** (Markov Decision Process)**.** A markov decision process $MDP$ is 3-tuple $(S, s_0, Steps)$, where
$S$ : a finite set of states;
$s_0 \in S$ : an initial state; and
$Steps \subseteq S \times A \times Dist(S)$: a probabilistic transition relation where $Dist(S)$ is a probability distribution over $S$.

In our reachability analysis procedure, we transform a given PTA into a finite MDP, and perform probabilistic verification based on the Value Iteration[8] technique.

Figure 3 shows an example of an MDP. In the figure, probability distributions are associated with transitions. In the figure, transitions which belong to the same distribution are connected with a small arc at their source points. The MDP has several non-deterministic choices at the state 1 and 4. For example, at the state 1, we have two choices; 1) the control moves to the state 2 with the probability 0.2 and to the state

3 with the probability 0.8, 2) the control moves to the state 4 with the probability 1.0.

### 2.3.1 Adversary

An MDP has non-deterministic transitions called action. To resolve the non-determinism, an adversary is used. The adversary requires a finite path on an MDP, and decides a transition to be chosen at the next step.

Figure 4 shows examples of resolving the non-determinism of the MDP shown in Fig. 3 by some adversaries. Figure 4. $a$) is the case where we choose the action which moves to the state 2 or state 3 at the initial state 1. On the other hand, $b$) and $c$) are the cases where we choose the action which moves to the state 4 at the initial state 1. In the case of $b$), we choose the action which moves to the state 7 when we move from the state 1 to state 4. Also, in the case of $c$), we choose the action which moves to the state 8 in the same trace.

Here, if we want to obtain the reachability probability from the state 1 to the state 10, under the adversary of $a$), we can obtain the probability 0.08 ($= 0.8 \times 0.2 \times 0.5$), which is the minimum reachability probability. On the other hand, under the adversary of $c$), we can obtain the probability 1.0 ($= 1.0 \times 1.0 \times 1.0$), which is the maximum reachability probability.

### 2.3.2 Value Iteration

A representative technique of model checking for an MDP is Value Iteration[8]. The Value Iteration technique can obtain both of maximum and minimum probabilities of reachability and safety properties, respectively. At each state, Value Iteration can select an appropriate action according to the property to be checked. Therefore, the technique can produce the adversary as well as the probability.

## 2.4 Timed Automaton

**Definition 2.4** (Timed Automaton)**.** A timed automaton $\mathscr{A}$ is a 6-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 finite set of clocks;
$I \subset (L \to c(C))$: a mapping from locations to clock constraints, called a location invariant; and
$T \subset L \times A \times c(C) \times \mathscr{R} \times L$,
where $c(C)$ is a clock constraint, called guards;
$\mathscr{R} = 2^C$: a set of clocks to reset.

A transition $t = (l_1, a, g, r, l_2) \in T$ is denoted by $l_1 \xrightarrow{a,g,r} l_2$. A map $\nu : C \to \mathbb{R}_{\geq 0}$ is called a clock assignment. We define $(\nu + d)(x) = \nu(x) + d$ for $d \in \mathbb{R}_{\geq 0}$. $r(\nu) = \nu[x \mapsto 0], x \in r$, where $\nu[x \mapsto 0]$ means the valuation that maps $x$ into zero, is also defined for $r \in 2^C$.

**Definition 2.5** (Semantics of a Timed Automaton)**.** Given a timed automaton $\mathscr{A} = (A, L, l_0, C, I, T)$, let $S \subseteq L \times \mathbb{R}^C_{\geq 0}$ be a set of whole states of $\mathscr{A}$. The initial state of $\mathscr{A}$ shall be given as $(l_0, 0^C) \in S$.

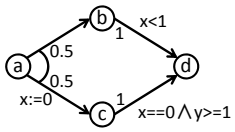



Figure 5: An Example of a PTA

For a transition $l_1 \xrightarrow{a,g,r} l_2 \ (\in T)$, the following two transitions are semantically defined. The former one is called an action transition, while the latter one is called a delay transition.

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

**Definition 2.6** (A Semantic Model of a Timed Automaton)**.** For timed automaton $\mathscr{A} = (A, L, l_0, C, I, T)$, an infinite transition system is defined according to the semantics of $\mathscr{A}$, where the model begins with the initial state.

## 2.5 Probabilistic Timed Automaton

A PTA is a kind of a timed automaton extended with probabilistic behavior. Therefore, using the PTA, we can evaluate quantitative properties such as performance of information systems based on the probabilistic model checking technique. In the PTA, a set of probabilistic distributions is used instead of a set $T$ of discrete transitions on the timed automaton.

**Definition 2.7** (Probabilistic Timed Automaton)**.** A probabilistic timed automaton $PTA$ is a 6-tuple $(A, L, l_0, C, I, prov)$, where
$A$: a finite set of actions;
$L$: a finite set of locations;
$l_0 \in L$: an initial location;
$C$: a finite set of clocks;
$I \subset (L \to c(C))$: a location invariant; and
$prob \subseteq L \times A \times c(C) \times Dist(2^C \times L)$: a finite set of probabilistic transition relations, where $c(C)$ represents a guard condition, and $Dist(2^C \times L)$ represents a finite set of probability distributions $p$. The Distribution $p(r, l) \in Dist(2^C \times L)$ represents the probability of resetting clock variables in $r$ and also moving to the location $l$;

Figure 5 shows an example of a PTA. In the figure, from the location $a$, it moves to the location $b$ with the probability 0.5 and also moves to the location $c$ letting the value of the clock $x$ reset to zero with the probability 0.5. Both of the arcs starting location $a$ are connected with a small arc at their source points, which represents that they belong to the same probability distribution.

**Definition 2.8** (Transitions of a Probabilistic Timed Automaton)**.** For $PTA = (A, L, l_0, C, I, prov)$, 6-tuple $(l, a, g, p, r, l')$ represents a transition generated by a probabilistic distribution $(l, a, g, p) \in prob$ such that $p(r, l') > 0$.

**Definition 2.9** (Semantics of a Probabilistic Timed Automaton)**.** Semantics of a probabilistic timed automaton $PTA = (A, L, l_0, C, I, prob)$ is given as a timed probabilistic system $TPS_{PTA} = (S, s_0, TSteps)$ where,

- $S \subseteq L \times \mathbb{R}^C$;
- $s_0 = (l_0, 0^C)$; and
- $TSteps \subseteq S \times A \cup \mathbb{R}_{\geq 0} \times Dist(S)$ is composed of action transitions and delay transitions.

  **a)** action transition
  if $a \in A$ and there exists $(l, a, g, p) \in prob$ such that $g(\nu)$ and $I(l')(r(\nu))$ for all $(r, l') \in support(p)$, $((l, \nu), a, \mu) \in TSteps$ where for all $(l', \nu') \in S$

$$\mu(l', \nu') = \sum_{r \subseteq C \wedge \nu' = r(\nu)} p(r, l').$$

  **b)** delay transition
  if $d \in \mathbb{R}_{\geq 0}$, and for all $d' \leq d$, $I(l)(\nu + d')$, $((l, \nu), d, \mu) \in TSteps$ where $\mu(l, \nu + d) = 1$.

The concrete delay in the delay transition can be decided non-deterministically on the semantics of a probabilistic timed automaton as well as those of a timed automaton.

In this paper, using a location $l$ and a zone $D$, we describe a set of semantic states as $(l, D) = \{(l, \nu) \mid \nu \in D\}$.

A probabilistic timed automaton is said to be well-formed if a probabilistic edge can be taken whenever it is enabled[2]. Formally, a probabilistic timed automaton $PTA = (A, L, l_0, C, I, prob)$ is well-formed if

$$\forall (l, g, p) \in prob.\, \forall \nu \in \mathbb{R}^C_{\geq 0}.\, (g(\nu)) \rightarrow \forall (r, l) \in support(p).\, I(l)(r(\nu)).$$

In this paper, we assume that a given PTA is well-formed.

**Definition 2.10** (Path on a Timed Probabilistic System)**.** A path $\omega$ with length of $n$ on a timed probabilistic system $TPS_{PTA} = (S, s_0, TSteps)$ is denoted as follows.

$$\omega = (l_0, \nu_0) \xrightarrow{d_0, \mu_0} (l_1, \nu_1) \xrightarrow{d_1, \mu_1} \cdots \xrightarrow{d_{n-1}, \mu_{n-1}} (l_n, \nu_n)$$

, where $(l_0, \nu_0) = s_0$, $(l_i, \nu_i) \in S$ for $0 \leq i \leq n$ and $((l_i, \nu_i), d_i, \mu) \in TSteps \wedge ((l_i, \nu_i + d_i), 0, \mu_i) \in TSteps \wedge (l_{i+1}, \nu_{i+1}) \in support(\mu_i)$ for $0 \leq i \leq n-1$.

For model checking of a probabilistic timed automaton, we extract a number of paths and calculate a summation of their occurrence probabilities in order to check the probability of satisfying a given property. The important point is that we have to choose a set of paths which are compatible with respect to time elapsing.

**Lemma 2.1** (Compatibility of two paths)**.** If two paths $\omega^\alpha = (l_0^\alpha, \nu_0^\alpha) \xrightarrow{d_0^\alpha, \mu_0^\alpha} (l_1^\alpha, \nu_1^\alpha) \xrightarrow{d_1^\alpha, \mu_1^\alpha} \cdots \xrightarrow{d_{n-1}^\alpha, \mu_{n-1}^\alpha} (l_n^\alpha, \nu_n^\alpha)$ and $\omega^\beta = (l_0^\beta, \nu_0^\beta) \xrightarrow{d_0^\beta, \mu_0^\beta} (l_1^\beta, \nu_1^\beta) \xrightarrow{d_1^\beta, \mu_1^\beta} \cdots \xrightarrow{d_{m-1}^\beta, \mu_{m-1}^\beta} (l_m^\beta, \nu_m^\beta)$ on a timed probabilistic system $TPS_{PTA}$ satisfy the following predicate $isCompatible$, then $\omega^\alpha$ and $\omega^\beta$ are said to be compatible.

$$isCompatible(\omega^\alpha, \omega^\beta) = \begin{cases} \text{true}, & \text{if } \forall i < min(n, m).\, l_i^\alpha = l_i^\beta \wedge d_i^\alpha = d_i^\beta \\ & \text{or there exists } i < min(n, m) \text{ such that} \\ & l_i^\alpha \neq l_i^\beta \wedge d_i^\alpha = d_i^\beta \wedge \\ & \forall j < i.\, l_j^\alpha = l_j^\beta \wedge d_j^\alpha = d_j^\beta \\ \text{false}, & \text{otherwise.} \end{cases}$$

Above predicate $isCompatible$ stands for that two paths are compatible if and only if one path is a prefix of the other, or same amount of delay is executed in both paths at the branching point of them.

**Lemma 2.2** (Compatibility of a set of paths)**.** If a set $\Omega$ of paths on a timed probabilistic system $TPS_{PTA}$ satisfies the following predicate $isCompatible$, then all of the paths over $\Omega$ are said to be compatible.

$$isCompatible(\Omega) = \begin{cases} \text{true}, & \text{if } \forall i \leq min(\Omega) \bigwedge_{\substack{\omega^\alpha, \omega^\beta \in \Omega \\ \wedge \omega^\alpha \neq \omega^\beta}} (l_i^\alpha = l_i^\beta \wedge d_i^\alpha = d_i^\beta) \\ & \text{or there exists } i \leq min(\Omega) \text{ such that} \\ & \bigwedge_{\substack{\omega^\alpha \omega^\beta \in \Omega \\ \wedge \omega^\alpha \neq \omega^\beta}} (l_i^\alpha \neq l_i^\beta \wedge d_i^\alpha = d_i^\beta \wedge \bigwedge_{j \leq i} (l_j^\alpha = l_j^\beta \wedge d_j^\alpha = d_j^\beta)), \\ & \text{and also } \bigwedge_{\substack{\Omega' \in 2^\Omega \wedge \\ \Omega' \neq \Omega \wedge |\Omega'| \leq 2}} isCompatible(\Omega') \\ \text{false}, & \text{otherwise.} \end{cases}$$

In Lemma2.2, we give the predicate $isCompatible$ for a set $\Omega$ of paths on a timed probabilistic system. In the lemma, we let paths in $\Omega$ be compatible if there is no contradiction with respect to time elapsing at the branching point of all the paths in $\Omega$, and also if the compatibility is kept for every subset of $\Omega$ which contains more than two paths.

Next, we give a simple example of a pair of paths which does not satisfy the compatibility. In the Fig. 5, paths from the location $a$ to $d$ are given as $\omega^\alpha = (a, x = 0 \wedge y = 0) \xrightarrow{0, 0.5} (b, x = 0 \wedge y = 0) \xrightarrow{0, 1.0} (d, x = 0 \wedge y = 0)$ which reaches to $d$ through $b$, and $\omega^\beta = (a, x = 0 \wedge y = 0) \xrightarrow{1, 0.5} (c, x = 0 \wedge y = 1) \xrightarrow{0, 1.0} (d, x = 0 \wedge y = 1)$ which reaches to $d$ through $c$. In the path $\omega^\alpha$, we are required to let delay at the location $a$ be less than one unit of time because of the guarded condition $x < 1$ of the transition between $b$ and $d$. On the other hand, in the path $\omega^\beta$, we are required to let delay at $a$ be grater than or equal one unit of time because of the condition $x == 0 \wedge y \geq 1$ of the transition between $c$ and $d$. Like the path $\omega^\alpha$ and $\omega^\beta$, if the required conditions of time elapsing at the branching point are contradict, we cannot use such paths simultaneously in the probability calculation.

## 2.6 CounterExample-Guided Abstraction Refinement

### 2.6.1 General CEGAR Technique

Since model abstraction sometimes over-approximates an original model, we may obtain spurious CEs which are infeasible

on the original model. Paper [5] gives an abstraction refinement framework called CEGAR (CounterExample-Guided Abstraction Refinement) (Fig. 1).

In the algorithm, at the first step (called Initial Abstraction), it generates an initial abstract model. Next, it performs model checking on the abstract model. In this step, if the model checker reports that the model satisfies a given specification, we can conclude that the original model also satisfies the specification, because the abstract model is an over-approximation of the original model. If the model checker reports that the model does not satisfy the specification, however, we have to check whether the CE detected is spurious or not in the next step (called Simulation). In the Simulation step, if we find that the CE is valid, we stop the loop. Otherwise, we have to refine the abstract model to eliminate the spurious CE, and repeat these steps until valid output is obtained.

#### 2.6.2 CEGAR Technique for a Timed Automaton

In Paper[6], we have proposed the abstraction refinement technique for a timed automaton based on the framework of CEGAR. In this approach, we remove all the clock attributes from a timed automaton. If a spurious CE is detected by model checking on an abstract model, we transform the transition relation on the abstract model so that the model behaves correctly even if we don't consider the clock constraints. Such transformation obviously represents the difference of behavior caused by the clock attributes. Therefore, the finite number of application of the refinement algorithm enables us to check the given property without the clock attributes. Since our approach does not restore the clock attributes at the refinement step, the abstract model is always a finite transition system without the clock attributes.

## 3 PROPOSED APPROACH

In this section, we will present our abstraction refinement technique for a probabilistic timed automaton. In the technique, we use the abstraction refinement technique for a timed automaton proposed in Paper[6]. Though the probability calculated on the abstract model may be spurious because the abstract model has no time attributes, the finite number of applications of the refinement algorithm enables us to obtain correct results on the abstract model. In addition, we resolve the compatibility problem shown in Sec.2.5 by performing a backward simulation technique and generating additional location to distinguish the required condition for every incompatible path. Figure 2 shows our abstraction refinement framework. As shown in the figure, we add another flow where we resolve the compatibility problem.

Our abstraction requires a probabilistic timed automaton $PTA$ and a property to be checked as its inputs. The property is limited by the PCTL formula $P_{<p}[\text{true } \mathbf{U}\ err]$. The formula represents a property that the probability of reaching to states where $err$ (which means an error condition in general) is satisfied, is less than $p$.

In model checking techniques, several properties presented in CTL[9], LTL[10], and others would be checked in general. The typical properties, however, are safety and progress. The reachability analysis is the primitive procedure for safety checking, thus model checking problems on several important properties represented in CTL could be reduced into the reachability analysis problem. Therefore, the reachability analysis is important problem. On the other hand, the limitation of the properties that we can check derives from the abstraction technique proposed in Paper[6]. Since the technique of Paper[6] focuses on properties of reachability, in this paper we also focus on reachability properties only.

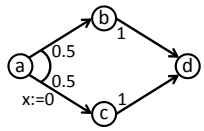



Figure 6: An Initial Abstract Model

### 3.1 Initial Abstraction

The initial abstraction removes all the clock attributes from a given probabilistic timed automaton as well as the technique in Paper[6]. The generated abstract model over-approximates the original probabilistic timed automaton. Also, the abstract model is just an MDP without time attributes.

**Definition 3.1** (Abstract Model)**.** For a given probabilistic timed automaton $PTA = (A, L, l_0, C, I, prob)$, a markov decision process $M\hat{D}P_{PTA} = (\hat{S}, \hat{s}_0, \hat{Steps})$ is produced as its abstract model, where

- $\hat{S} = L$
- $\hat{s}_0 = l_0$
- $\hat{Steps} = \{\ (s, a, p) \mid (s, a, g, p) \in prob\ \}$

Figure 6 shows an initial abstract model for the PTA shown in Fig. 5 As shown in the figure, the abstract model is just an MDP where all of the clock constraints are removed though we keep a set of clock reset as a label of transitions.

### 3.2 Model Checking

In model checking, we apply Value Iteration[8] into the markov decision process obtained by abstraction and calculate a maximum reachability probability. Also, it decides an action to be chosen at every state as an adversary. If the obtained probability is less than $p$, we can terminate the CEGAR loop and conclude that the property is satisfied.

Although Value Iteration can calculate a maximum reachability probability, it cannot produce concrete paths used for the probability calculation. To obtain the concrete paths, we use an approach proposed in Paper[11] which can produce CE paths for PCTL formulas. The approach translates a probabilistic automaton into a weighted digraph. And we can obtain at most $k$ paths by performing $k$-shortest paths search on the graph.

**Definition 3.2** (Path on the Abstract Model)**.** A path $\hat{\omega}$ on an abstract model $M\hat{D}P_{PTA} = (\hat{S}, \hat{s}_0, \hat{Steps})$ for $PTA = (A, L, l_0, C, I, prob)$ is given as follows,

$$\hat{\omega} = \hat{s}_0 \overset{a_0,p_0,r_0}{\longrightarrow} \hat{s}_1 \overset{a_1,p_1,r_1}{\longrightarrow} \cdots \overset{a_{n-1},p_{n-1},r_{n-1}}{\longrightarrow} \hat{s}_n$$

, where $\hat{s}_i \in \hat{S}$ for $0 \leq i \leq n$ and $(\hat{s}_i, a_i, p_i) \in \hat{Steps} \wedge (r_i, \hat{s}_{i+1}) \in support(p_i)$ for $0 \leq i \leq n-1$.

As defined in Def. 3.2, we associate a set $r$ of clock reset with a path on an abstract model in order to show the difference of $r$ over the probabilistic distribution $p$.

For the abstract model shown in Fig. 6, Value Iteration outputs 1.0 as the probability that it reaches to the location $d$ from the location $a$. On the other hand, $k$-shortest paths search ($k \geq 2$) detects two paths $\hat{\omega}^\alpha = a \overset{\tau,0.5,\{\}}{\longrightarrow} b \overset{\tau,1.0,\{\}}{\longrightarrow} d$ and $\hat{\omega}^\beta = a \overset{\tau,0.5,\{x:=0\}}{\longrightarrow} c \overset{\tau,1.0,\{\}}{\longrightarrow} d$, where $\tau$ represents a label for transitions with no label in the figure.

## 3.3 Simulation

Simulation checks whether all the paths obtained by $k$-shortest paths search are feasible or not on the original probabilistic timed automaton. We use the simulation algorithm proposed in Paper[6] where we use some operations of DBM (Difference Bound Matrix)[12] to obtain zones which are reachable from the initial state. If there is at least one path which is infeasible on the original PTA, we proceed to the abstraction refinement step.

Figure 7 shows the simulation results for two paths $\hat{\omega}^\alpha$ and $\hat{\omega}^\beta$. Simulation concludes that the two paths are feasible on the original PTA.

## 3.4 Abstraction Refinement

In this step, we refine the abstract model so that the given spurious CE also becomes infeasible on the refined abstract model. We can use the algorithm proposed in Paper[6]. Since the algorithm of Paper[6] performs some operations on transitions of a timed automaton, we replace such operations by those on probability distributions of a probabilistic timed automaton.

## 3.5 Compatibility Checking

When all the paths obtained by $k$-shortest paths search are feasible and a summation of occurrence probabilities of them is greater than $p$, we also have to check whether all the paths are compatible or not. In this compatibility checking step, at each location of the paths, we have to obtain a condition (zone) which is reachable from the initial state and also reachable to the last state along with the path. Next, we check the compatibility of such conditions among all paths. To obtain such conditions, we have to perform both forward simulation shown in Sec. 3.3 and backward simulation for each path, and merge the results. For the result of forward simulation, we can reuse the result obtained in the Simulation step. Then we check the compatibility based on Lemma 2.2.

**Algorithm 1** BackwardSimulation($PTA$, $\omega$)

```
1: /* PTA = (A, L, l_0, C, I, prob)
      ω̂ = ŝ_0 -a_0,p_0,r_0-> ŝ_1 -a_1,p_1,r_1-> ... -a_{n-1},p_{n-1},r_{n-1}-> ŝ_n */
2: D^ω̂_{b,n} := I(ŝ_n)
3: for i := n − 1 downto 0 do
4:     D^ω̂_{b,i} := D^ω̂_{b,i+1}
5:     D^ω̂_{b,i} := down(D^ω̂_{b,i})              /* reverse the time elapse */
6:     D^ω̂_{b,i} := and(D^ω̂_{b,i}, I(ŝ_{i+1}))
7:     D^ω̂_{b,i} := free(D^ω̂_{b,i}, r_i)         /* remove all constraints on r_i */
8:     D^ω̂_{b,i} := and(D^ω̂_{b,i}, g_i)          /* (ŝ_i, a_i, g_i, p_i) ∈ prob */
9:     D^ω̂_{b,i} := and(D^ω̂_{b,i}, I(ŝ_i))
10: end for
11: return D^ω̂_b
```

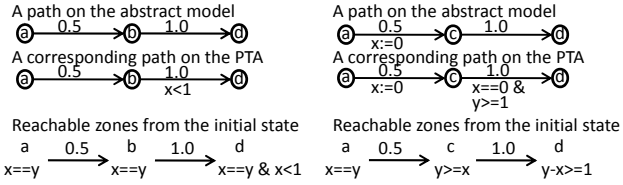



Figure 7: Simulation Results for a Set of Paths

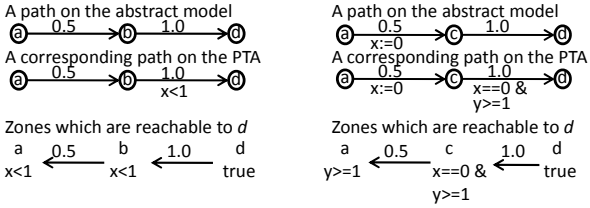



Figure 8: Results of Backward Simulation for a Set of Paths

### 3.5.1 Backward Simulation

Algorithm 1 implements the backward simulation. Functions $and$, $free$, $down$ used in the algorithm are operation functions on a zone, and are defined in Paper[12]. Formally, for a zone $D$, a constraint $c$, and a set $r$ of clock reset, those functions are defined as follows; $and(D, c) = \{u \mid u \in D \wedge u \in c\}$, $free(D, r) = \{u \mid r(u) \in D\}$, and $down(D) = \{u \mid u + d \in D \wedge d \in \mathbb{R}_{\geq 0}\}$

Figure 8 shows results of backward simulation for two paths $\hat{\omega}^\alpha$ and $\hat{\omega}^\beta$ detected in Sec. 3.2.

### 3.5.2 Determination of Compatibility

In this step, we check compatibility of the set $\hat{\Omega}$ of paths on the abstract model using the required conditions obtained by both of forward and backward simulation. Algorithm 2 checks the compatibility of $\hat{\Omega}$ using the Algorithm 3.

Algorithm 3 first checks whether the required conditions of the $i$-th locations for each path are compatible or not ($l2$-$l8$) using the results of forward and backward simulation. Next,

**Algorithm 2** IsCompatible($PTA$, $\hat{\Omega}$, $D_f$, $D_b$)

```
1: /* PTA = (A, L, l_0, C, I, prob), Ω̂ is a set of abstract paths,
      and D_f and D_b are sets of forward and backward simulation
      results for each path ω̂ ∈ Ω̂,respectively. */
2: return CompatibleCheck(PTA, Ω̂, D_f, D_b, 0)
```

**Algorithm 3** CompatibleCheck($PTA$, $\hat{\Omega}$, $D_f$, $D_b$, $i$)

1: $D' := true$
2: **foreach** $\hat{\omega} \in \hat{\Omega}$ such that $length(\hat{\omega}) \geq i$ **do**
3: $D^{\hat{\omega}}_{c,i} := D^{\hat{\omega}}_{f,i} \cap D^{\hat{\omega}}_{b,i}$
4: $D' := D' \cap D^{\hat{\omega}}_{c,i}$
5: **if** $D' = \emptyset$ **then**
6: **return** $false$
7: **end if**
8: **end for**
9: $S^{\hat{\Omega}}_{i+1} := SplitPathSet(\hat{\Omega}, i+1)$
10: /* split $\hat{\Omega}$ into a set of its subsets without overlap with respect to the $i$+1-th location and clock reset for each path in $\hat{\Omega}$ */
11: **foreach** $\hat{\Omega}' \in S^{\hat{\Omega}}_{i+1}$ such that $|\hat{\Omega}'| \geq 2$ **do**
12: **if** $CompatibleCheck(PTA, \hat{\Omega}', D, i+1) = false$ **then**
13: **return** $false$
14: **end if**
15: **end for**
16: **return** $true$

**Algorithm 4** SplitPathSet($\hat{\Omega}$, $i$)

1: $S := \emptyset$
2: **foreach** $\hat{\omega} \in \hat{\Omega}$ **do**
3: /* $\hat{\omega} = \hat{s}_0 \overset{a_0,p_0,r_0}{\rightarrow} \hat{s}_1 \overset{a_1,p_1,r_1}{\rightarrow} \cdots \overset{a_{n-1},p_{n-1},r_{n-1}}{\rightarrow} \hat{s}_n$ */
4: **if** $\hat{\Omega}_{r_{i-1},\hat{s}_i} \notin S$ **then**
5: $\hat{\Omega}_{r_{i-1},\hat{s}_i} := \{\hat{\omega}\}$
6: $S := S \cup \hat{\Omega}_{r_{i-1},\hat{s}_i}$
7: **else**
8: $\hat{\Omega}_{r_{i-1},\hat{s}_i} := \hat{\Omega}_{r_{i-1},\hat{s}_i} \cup \{\hat{\omega}\}$
9: **end if**
10: **end for**
11: **return** $S$

the algorithm divides $\hat{\Omega}$ into some subsets of it based on the ($i$+1)-th locations and the set of clock reset for each path ($l$9). Then, it checks the compatibility for the following sequences of paths by applying the algorithm into the divided subsets recursively ($l$11-$l$15). Although the predicate $isCompatible$ in the Lemma 2.2 checks the compatibility for each subset of $\Omega$, the algorithm omit redundant checks by dividing $\Omega$ based on the branches of the paths.

For the path $\hat{\omega}^{\alpha}$ in Sec. 3.2, zones at $a$ which is reachable from initial state and which can move to $d$ are given as $D^{\hat{\omega}^{\alpha}}_{f,0} = (x == y)$, and $D^{\hat{\omega}^{\alpha}}_{b,0} = (x < 1)$, respectively. Also, a zone of the product of them is given as $D^{\hat{\omega}^{\alpha}}_{c,0} = (x == y \wedge x < 1)$. Similarly, for the path $\hat{\omega}^{\beta}$, the product zone is given as $D^{\hat{\omega}^{\beta}}_{c,0} = (x == y \wedge y > 1)$. Since $D^{\hat{\omega}^{\alpha}}_{c,0}$ and $D^{\hat{\omega}^{\beta}}_{c,0}$ contradict each other, we can conclude that the paths $\hat{\omega}^{\alpha}$ and $\hat{\omega}^{\beta}$ are incompatible each other.

## 3.6 Model Transformation

When the compatibility check procedure decides a given set $\hat{\Omega}$ of paths is incompatible at $i$-th location, our proposed algorithm resolves the incompatibility by refining behaviors from the $i$-th location. Our algorithm uses $D^{\hat{\omega}}_{c}$ which is a product of results of forward and backward simulation for a path $\hat{\omega} \in \hat{\Omega}$. It duplicates locations which are reachable from the zone $D^{\hat{\omega}}_{c,i}$ by an action associated with the $i$-th distribution $p_i$. Also it constructs transition relations so that the transformation becomes equivalent transformation. For example, transition relations from a duplicated location are duplicated if the relations are executable from the invariant associated with the duplicated location.

**Algorithm 5** TransformPTA($PTA$, $D_c$, $\hat{\Omega}$, $i$)

1: $D_{complement}$ := true
2: **foreach** $\hat{\omega} \in \hat{\Omega}$ **do**
3: $L_{dup} := DuplicateLocation(PTA, \hat{\omega}, D^{\hat{\omega}}_{c,i}, i)$
4: $L := L \cup L_{dup}$
5: $prob_{dup} := DuplicateDistribution(PTA, \hat{\omega}, L_{dup}, i)$
6: $prob := prob \cup prob_{dup}$
7: $D_{complement} := D_{complement} \cap D^{\omega}_{c,i}$
8: **end for**
9: $L_{dup} := DuplicateLocation(PTA, \hat{\omega}, D_{complement}, i)$
10: $L := L \cup L_{dup}$
11: $prob_{dup} := DuplicateDistribution(PTA, \hat{\omega}, L_{dup}, i)$
12: $prob := prob \cup prob_{dup}$
13: $prob := RemoveDistribution(PTA, \hat{s}_i, p_i)$
14: /* for all path $\hat{\omega} \in \hat{\Omega}$, the $i$-th state $\hat{s}_i$ and $i$-th probability distribution is $p_i$ */
15: **return** $PTA$

**Algorithm 6** DuplicateLocation($PTA, \hat{\omega}, D, i$)

1: /* $PTA = (A, L, l_0, C, I, prob)$
$\hat{\omega} = \hat{s}_0 \overset{a_0,p_0,r_0}{\rightarrow} \hat{s}_1 \overset{a_1,p_1,r_1}{\rightarrow} \cdots \overset{a_{n-1},p_{n-1},r_{n-1}}{\rightarrow} \hat{s}_n$ */
2: $L_{dup} := \emptyset$
3: **foreach** $(l, r) \in L \times 2^C$ such that $p_i(l, r) > 0$ **do**
4: $(l, D) := Succ((\hat{s}_i, D), e)$
5: /* $succ$ returns a successor state set through a given edge $e$, and $e = (\hat{s}_i, a_i, g, p_i, r, l)$ */
6: $l_{dup} := newLocation()$
7: $I(l_{dup}) := D$
8: $L_{dup} = l_{dup}$
9: **end for**
10: **return** $L_{dup}$

Algorithm 5 transforms a given PTA with considering its compatibility. The algorithm calls $DuplicateLocation$ (Algorithm 6) which duplicates locations, $DuplicateDistribution$ (Algorithm 7) which duplicates probabilistic transitions, and $RemoveDistribution$ (Algorithm 9) which removes probabilistic transitions. The procedure $Succ$ in Algorithms 6 and 8 calculates a successor state set from a given state set $S$ through a given edge $e = (l, a, g, p, r, l')$, i.e. $Succ(S, e) = \{(l', r(\nu) + d) \mid (l, \nu) \in S \wedge g(\nu) \wedge I(l')(r(\nu)) \wedge \forall d' \leq d.I(l')(r(\nu) + d')\}$

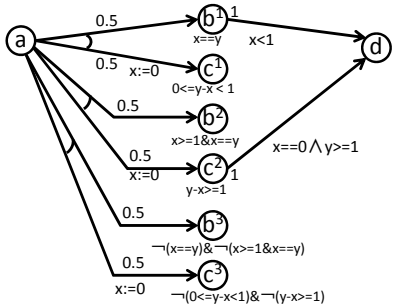



Figure 9: A Transformed PTA

**Algorithm 7** DuplicateDistribution($PTA, \hat{\omega}, L_{dup}, i$)

/* $PTA = (A, L, l_0, C, I, prob)$
$\hat{\omega} = \hat{s}_0 \overset{a_0,p_0,r_0}{\rightarrow} \hat{s}_1 \overset{a_1,p_1,r_1}{\rightarrow} \cdots \overset{a_{n-1},p_{n-1},r_{n-1}}{\rightarrow} \hat{s}_n$ */
$prob_{dup} := \emptyset$
$p_{dup} := newDistribution()$
/* generate a new distribution over $L \times 2^C$ */
**foreach** $(l, r) \in L \times 2^C$ **do**
  $p_{dup}(l_{dup}, r) := p_i(l, r)$
  /* $l_{dup}$ is a duplicate location of $l$ generated by DuplicateLocation algorithm */
**end for**
$prob_{dup} := Prob_{dup} \cup \{(\hat{s}_i, a_i, g, p_{dup})\}$
/* $(\hat{s}_i, a_i, g, p_i) \in prob$ */
**foreach** $l_{dup} \in L_{dup}$ **do**
  $prob_{dup} := Prob_{dup} \cup$
  $DuplicateDistFromDupLoc(PTA, l_{dup})$
**end for**
**return** $p_{dup}$

**Algorithm 8** DuplicateDistFromDupLoc($PTA, l_{dup}$)

/* $PTA = (A, L, l_0, C, I, prob)$, and let $l$ be an original location of $l_{dup}$ */
$prob_{dup} := \emptyset$
**foreach** $(l, a, g, p) \in Prob$ **do**
  $f_{dup} :=$ true, $p_{dup} := newDistribution()$
  **foreach** $(l', r) \in L \times 2^C$ **do**
    **if** $Succ((l, I(l_{dup})), e) \neq \emptyset$ **then**
      /* $e = (l, a, g, p, r, l')$ */
      $p_{dup}(l', r) = p(l, r)$
    **else**
      $f_{dup} :=$ false
      **break**
    **end if**
  **end for**
  **if** $f_{dup}$ **then**
    /* duplicate the distribution if it is executable from the duplicate location */
    $Prob_{dup} := Prob_{dup} \cup \{(l, a, g, p_{dup})\}$
  **end if**
**end for**

Figure 9 shows the transformed PTA by applying the model transformation procedure for the paths $\hat{\omega}^\alpha$ and $\hat{\omega}^\beta$. The locations $b^1$ and $c^1$ are duplicated locations based on the path $\hat{\omega}^\alpha$ and the zone $D^{\hat{\omega}^\beta}_{c,0} = (x == y \wedge x < 1)$ on the location $a$. We associate invariants to $b^1$ and $c^1$ based on zones which are reachable from $D^{\hat{\omega}^\beta}_{c,0}$ through transitions from $a$ to $b$, and from $a$ to $c$, respectively. Also, we duplicate a transition from $b$ to $d$ as the transition from $b^1$ to $d$ because the transition is feasible from the invariant of $b^1$. On the other hand, we do not duplicate a transition from $c$ to $d$ because the transition is not feasible from the invariant of $c^1$. Similarly, locations $b^2$ and $c^2$ are duplicated locations based on the path $\hat{\omega}^\beta$ and the zone $D^{\hat{\omega}^\beta}_{c,0}$. Locations $b^3$ and $c^3$ are generated as complements of the invariant associated with each duplicated location in order to preserve the equivalence.

By transforming the original PTA in such a way, if we remove all clock constraints from the model in Fig. 9, Value Iteration on its abstract model outputs 0.5 as the maximum probability.

**Algorithm 9** RemoveDistribution($PTA, l, p$)

/* $PTA = (A, L, l_0, C, I, prob)$, and let $l$ be an original location of $l_{dup}$ */
**foreach** $(l, a, g, p)$ **do**
  $prob := prob \setminus \{(l, a, g, p)\}$
**end for**
**return** $prob$

# 4 EXPERIMENTS

We have implemented a prototype of our proposed approach with Java, and performed some experiments. Though the prototype can check the compatibility of a given set of paths, currently it cannot deal with the model transformation.

The prototype performs $k$-shortest paths search and simulation concurrently in order to reduce execution time. By implementing the algorithms concurrently, we have not to wait until all of $k$ paths are detected, i.e. if a path is detected by the $k$-shortest paths search algorithm, we can immediately apply simulation and (if needed) abstraction refinement procedures.

Also, our prototype continues the $k$-shortest search algorithm when a spurious CE is detected and the refinement algorithm is applied. If other paths which do not overlap with the previous spurious CEs, are detected, we can apply simulation and refinement algorithms to it again. This helps us reduce the number of CEGAR loop.

## 4.1 Goals of the Experiments

In this experiment, we evaluated the performance of our proposed approach with regard to execution time, memory consumption, and qualities of obtained results. As a target for comparison, we chose the approach of Digital Clocks[3] where they approximate clock evaluations of a PTA by integer values.

## 4.2 Example

We used a case study of the FireWire Root Contention Protocol[13] as an example for this experiment. This case study concerns the Tree Identify Protocol of the IEEE 1394 High Performance Serial Bus (called "FireWire") which takes place when a node is added or removed from the network. In the experiment, we checked the probability that a leader is not selected within a given deadline. The probabilistic timed automaton for the example is composed of two clock variables, 11 locations, and 24 transitions.

## 4.3 Procedure of the Experiments

In this experiment, we checked the property that "the probability that a leader cannot be elected within a given $deadline$ is less than $p$." We considered three scenarios where the parameter $deadline$ is 5, 10, 20 $\mu$s, respectively. Also, for each scenario, we conducted two experiments where the value of $p$ is 1.5 times as an approximate value of the maximum probability obtained by the Digital Clocks approach[3] and a half of it, respectively. In the proposed approach, we searched at most 5000 paths by letting the parameter $k$ of the $k$-shortest

Table 1: Experimental Result

| | | Digital Clocks[3] | | | | Proposed Approach | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $D(\mu s)$ | $p$ | $Result$ | $Time(s)$ | $State$ | $MEM(MB)$ | $Result$ | $Time(s)$ | $Loop$ | $State$ | $Heap(MB)$ |
| 5 | $1.09\times10^{-1}$ | false | 20.90 | 297,232 | 10.2 | false | 4.19 | 10 | 37 | 8.0 |
| | $3.28\times10^{-1}$ | true | 20.89 | 297,232 | 10.2 | true | 3.60 | 9 | 36 | 8.0 |
| 10 | $1.26\times10^{-2}$ | false | 54.80 | 685,232 | 21.7 | false | 8.16 | 19 | 134 | 8.0 |
| | $3.79\times10^{-2}$ | true | 54.82 | 685,232 | 21.7 | true | 6.57 | 15 | 115 | 8.0 |
| 20 | $1.85\times10^{-4}$ | false | 176.93 | 1,461,232 | 41.0 | false | 1186.08 | 47 | 477 | 64.0 |
| | $5.56\times10^{-4}$ | true | 177.46 | 1,461,232 | 41.0 | true | 31.32 | 32 | 435 | 8.0 |

Table 2: Analysis of Counter Example Paths

| $D(\mu s)$ | $p$ | $Path$ | $Probability$ | $CC(ms)$ |
|---|---|---|---|---|
| 5 | $1.0938\times10^{-1}$ | 7 | $1.2500\times10^{-1}$ | 0.7 |
| 10 | $1.2635\times10^{-2}$ | 43 | $1.2695\times10^{-2}$ | 5.9 |
| 20 | $1.8500\times10^{-4}$ | 2534 | $1.8501\times10^{-4}$ | 296.9 |

paths search algorithm be 5000. For evaluation of existing approach, we used the probabilistic model checker PRISM[14].

The experiments were performed under Intel Core2 Duo 2.33 GHz, 2GB RAM, and Fedora 12 (64bit).

## 4.4 Results of the Experiments

The results are shown in Table 1. The column of $D$ means the value of $deadline$. For each approach, columns of $Results$, $Time$, and $States$ show the results of model checking, execution time of whole process, and the number of states constructed, respectively. The column $MEM$ in the columns of the Digital Clocks shows the memory consumption of PRISM. The columns $Loop$ and $Heap$ in the columns of the proposed approach show the number of CEGAR loops executed and the maximum heap size of the Java Virtual Machine (JVM) which executes our prototype, respectively.

Table 1 shows that for all cases we can dramatically reduce the number of states and obtain correct results. Moreover, we can reduce the execution time more than 80 percent except for the case when $deadline = 20\mu s$ and $p = 1.85 \times 10^{-4}$. In this case, however, the execution time drastically increases.

Table 2 shows the results of analysis of CE paths obtained when the results of model checking are false. The columns of $Path$, $Probability$ and $CC$ show the number of CE paths, the summation of occurrence probability of them, and execution time for compatibility checking, respectively. For this example, the obtained sets of CE paths are compatible in every case.

## 4.5 Discussion

From the results shown in Table 1, we can see that our proposed approach is efficient with regard to both execution time and the number of states. Especially, the number of states decrease dramatically. The execution time is also decreased even though we perform model checking several times shown in the column of $Loop$.

On the other hand, in the case when $deadline = 20\mu s$ and $p = 1.85 \times 10^{-4}$, the execution time increases drastically. We think that as shown in Table 2 we have to search 2534 paths and this causes the increase of execution time especially for $k$-shortest paths search. A more detailed analysis shows that the execution time for $k$-shortest paths search accounts for 1123 seconds of total execution time of 1186 seconds. Also, the results shows that the JVM needs 64MB as its heap size in this case. This is because compatibility checking for 2534 of paths needs a large amount of the memory. From the results, we have to resolve a problem of the scalability when the number of candidate paths for a CE becomes large.

# 5 CONCLUSION

This paper proposed the abstraction refinement technique for a probabilistic timed automaton by extending the existing abstraction refinement technique for a timed automaton.

Future work includes completion of implementation. General DBM does not support $not$ operator[15]; so we have to investigate efficient algorithms for the $not$ operator.

# ACKNOWLEDGMENTS

This work is being conducted as a part of Stage Project, the Development of Next Generation IT Infrastructure, supported by Ministry of Education, Culture, Sports, Science and Technology, as well as Grant-in-Aid for Scientific Research C(21500036), as well as grant from The Telecommunications Advancement Foundation.

# REFERENCES

[1] E. M. Clarke, O. Grumberg and D. A. Peled, editors, "Model Checking," MIT Press (1999).

[2] M. Kwiatkowska, G. Norman, J. Sproston and F. Wang, "Symbolic model checking for probabilistic timed automata," Information and Computation, Vol. 205, No. 7, pp. 1027–1077 (2007).

[3] M. Kwiatkowska, G. Norman and J. Sproston, "Performance Analysis of Probabilistic Timed Automata Using Digital Clocks," Formal Methods in System Design, Vol. 29, No. 1, pp. 33–78 (2006).

[4] M. Kwiatkowska, G. Norman and D. Parker, "Stochastic Games for Verification of Probabilistic Timed Automata," Proc. of the 7th Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS'09), Vol. 5813 of *LNCS*, pp. 212–227 (2009).

[5] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu and V. Helmut, "Counterexample-guided Abstraction Refinement

for Symbolic Model Checking," Journal of the ACM, Vol. 50, No. 5, pp. 752–794 (2003).
[6] T. Nagaoka, K. Okano and S. Kusumoto, "An Abstraction Refinement Technique for Timed Automata Based on Counterexample-Guided Abstraction Refinement Loop," IEICE Transactions on Information and Systems, Vol. E93-D, No. 5, pp. 994–1005 (2010).
[7] C. Derman, editor, "Finite-State Markovian Decision Processes," New York: Academic Press (1970).
[8] D. P. Bertsekas, "Dynamic Programming and Optimal Control," Athena Scientific (1995).
[9] E. Clarke, E. Emerson and A. Sistla, "Automatic verification of finite-state concurrent systems using temporal logics," ACM Transactions on Programming Languages and Systems, Vol. 8, No. 2, pp. 244–263 (1986).
[10] A. Pnueli, "The temporal logic of programs," Proc. of the 18th Int. Symp. on Foundation of Computer Science (FOCS), pp. 46–57 (1977).
[11] H. Aljazzar and S. Leue, "Directed Explicit State-Space Search in the Generation of Counterexamples for Stochastic Model Checking," IEEE Transactions on Software Engineering, Vol. 36, No. 1, pp. 37–60 (2010).
[12] J. Bengtsson and W. Yi, "Timed Automata: Semantics, Algorithms and Tools," Lecture Notes on Concurrency and Petri Nets, Vol. 3098, pp. 87–124 (2004).
[13] M. Kwiatkowska, G. Norman and J. Sproston, "Probabilistic Model Checking of Deadline Properties in the IEEE1394 Firewire Root Contention Protocol," Formal Aspects of Computing, Vol. 14, No. 3, pp. 295–318 (2003).
[14] A. Hinton, M. Kwiatkowska, G. Norman and D. Parker, "PRISM: A Tool for Automatic Verification of Probabilistic Systems," Proc. of the 12th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06), Vol. 3920 of *LNCS*, pp. 441–444 (2006).
[15] A. David, J. Hakansson, K G. Larsen and P. pettersson, "Model Checking Timed Automata with Priorities using DBM Subtraction," Proc. of the 4th Int. Conf. on Formal Modelling and Analysis of Timed Systems, pp. 128–142 (2006).
[16] A. Morimoto, R. Komagata and S. Yamane, "Probabilistic Timed CEGAR," Technical report of IEICE CST 109 (73), pp. 25–30 (2009).

(Received September 2, 2010)
(Revised April 24, 2011)

**Takeshi Nagaoka** received the MI, and Ph.D degrees in Computer Science from Osaka University in 2007, and 2011, respectively. His research interests include abstraction techniques in model checking, especially a timed automaton and a probabilistic timed automaton.

**Akihiko Ito** received the BE and MI degrees in Computers Sciences from Hiroshima University and Osaka University in 2008 and 2010, respectively. His research interests include model checking, especially timed automaton and probabilistic timed automaton.

**Toshiaki Tanaka** received the BE and MI degrees in Computer and Systems Engineering from Kobe University, and Osaak University in 2009, and 2011, rewspectively. His research interests include parallelization of model checking, especially a timed automaton.

**Kozo Okano** received the BE, ME, and Ph.D degrees in Information and Computer Sciences from Osaka University, in 1990, 1992, and 1995, respectively. Since 2002 he has been an associate professor in the Graduate School of Information Science and Technology, Osaka University. In 2002, he was a visiting researcher of the Department of Computer Science, University of Kent at Canterbury. In 2003, he was a visiting lecturer at the School of Computer Science, University of Birmingham. His current research interests include formal methods for software and information system design. He is a member of IEEE, IEICE of Japan and IPS of Japan.

**Shinji Kusumoto** received the BE, ME, and DE degrees in information and computer sciences from Osaka University in 1988, 1990, and 1993, respectively. He is currently a professor in the Graduate School of Information Science and Technology at Osaka University. His research interests include software metrics and software quality assurance technique. He is a member of the IEEE, the IEEE Computer Society, IPSJ, IEICE, and JFPUG.

# Simulated Collaboration to Understand Japanese Offshore Software Development in China

Takaya Yuizono* and Lihua Xuan*

*School of Knowledge Science, Japan Advanced Institute of Science and Technology, Japan
yuizono@jaist.ac.jp

*Abstract* – Two primary issues in Japanese offshore software development in China are how to manage specifications and intercommunication. To consider the software development using groupware technology in a laboratory setting, we proposed a simulated collaboration task. In the experiment, the Chinese, Japanese, and Japanese-Chinese groups corresponded with offshore development, domestic development, and cooperative development, which is a proposed future style, respectively. The laboratory experiments' results indicated that a well-collaborated team produced a good model chart; the Chinese participants experienced difficulty in using Japanese-language communication, making some of them self-assertive; and, in comparison to the Japanese participants, the Chinese participants tended to be satisfied with their results of the model chart, considering them neither good nor bad.

*Keywords*: offshore software development, software specification, laboratory simulation

## 1 INTRODUCTION

Nowadays, both Japan and China widely experience interactions in their trade and between their people. In trade, software development is representative of work that necessitates knowledge workers, making it an important collaborative undertaking. Japanese IT offshoring came into prominence in the 1980s, motivated by pressures to reduce labor costs and, to date, has been carried out primarily in China. Offshoring in recent years requires the management of the complete process of software development [1].

The largest amount of offshoring from Japan is sent priarily to China; hence, China receives the highest number of outsourcing jobs from Japan [2]. Offshoring from Japan to China is still prominent in the software development industry, although offshoring to India and Vietnam has also increased. As is well known, all offshoring projects were not always successful because of differences in language, culture, corporate climate, and work environment.

Groupware technologies, which support distributed software development, have the potential for IT offshoring because the major issues in offshore development involve software specification and human communication. In order to understand the software development process in different cultural settings, the time taken, expenses incurred, and detailed observations or logs maintained by an IT vendor are necessary [3]. In addition, evaluating a new type of offshoring may not be realistically acceptable to the vendor. Therefore, we design a collaboration task that simulates offshoring development and is availabile in a laboratory setting.

In the following section, we describe the collaboration model for offshore software development. In the third section, we explain the experiment and proposed collaboration task. In the fourth section, we present the results of the experiment and discuss them. In the last section, we summarize this paper and suggest a future direction.

## 2 SIMULATED COLLABORATION FOR OFFSHORE SOFTWARE DEVELOPMENT

### 2.1 Collaboration Task

We proposed a work model on the basis of the collaboration between the Japanese and Chinese in offshore software development, to investigate or explore the issues that exist in this collaboration.

The questionnaire on offshore software development was based on that of Nakahara and Fujino [4]. The results showed that, in the case of Japanese companies, about 20 percent indicated a communication problem, about 30 percent indicated problems with confirmation and modification of specifications, and about 20 percent indicated a communication problem in the case of offshore companies.

On the other hand, software development based on a specification is difficult to replicate in a laboratory experiment because the work is limited to only a few hours and the recruitment of programmers for the software development is unrealistic, since many students have yet to learn the essentials of programming. Therefore, we considered model charting as the collaborative work instead of program development. The model that simulates the collaboration in offshore software development is depicted in Figure 1.

In the collaboration task, the client and vendor are located at separate remote sites. The client sends a software specification to the vendor, and the vendor develops a model chart instead of developing a software program according to the specification.

The vendor consists of three persons—the team leader and two workers, who are not programmers—and they work in the same place. In the laboratory, the workers can pose any questions regarding the specification to the leader. When the leader is faced with a question that has not been resolved, he/she can question the client, who is in a different location.

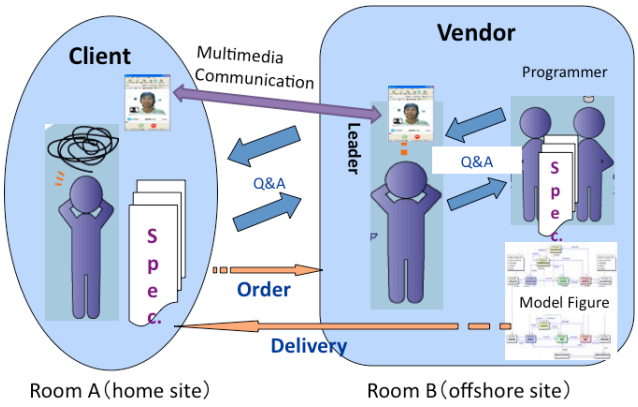



Figure 1: Collaboration model of offshore software development

The vendor prepares a system chart on the basis of the specification that is assumed to be ordered by the client. For this collaboration, workers can use a paper and pencil, but they must depict the chart as a final groupware product that supports a shared screen and chart-making function. The workers use a label and arrow representation for the charting. In the chart, a label object is used to represent an object in the specification and an arrow is used to represent the sending of an event; the arrow is often added to the string using a label object to explain the event. The leader can check the chart on the shared screen using the groupware.

In the task, two versions of specifications were prepared in order to mimic a problem caused by a specification change, which is considered a typical cause of confusion in software development [4]. The modified specification is an advanced version of the specification that was initially ordered by the client.

Three team patterns are considered because the characteristics of offshore software development are expected to be revealed in the comparison of these patterns. In the first pattern, all the people belonging to the vendor's side are Chinese. This simulates ordinary offshore software development and, hence, is labeled the "Chinese Group." In this case, the team leader is a Chinese who is fluent in Japanese, but the workers cannot speak Japanese though they can read Japanese sentences. In the second pattern, all the people are Japanese and are labeled the "Japanese Group," in order to simulate a domestic software development team for the comparison. The third pattern is considered the future style of offshore software development and is reflected by a collaboration of Japanese and Chinese participants and, hence, is labeled the "JC Group." In this pattern, the leader is Japanese and the workers are Chinese.

## 2.2 Software Specification for the Collaboration Task

The software specification was prepared before the collaboration task. The content of the specification was taken from a standard problem on stock management, for software specification research [5].

The theme for problem solving in the collaboration experiment was entitled "The store management system for a liquor company" and was developed to compare program design methods.

The structure of the specification was referred to as the guideline for the requirement specification [6], and the language description in Japanese follows Ooki et al's explanation [7]. The Japanese description reflects concurrent object processing and has no symbolic description that depends on a specific program design technique. Nowadays, a description with these characteristics suits the specification description for modem object-oriented development. A part of the specification is depicted in Figure 2.

In the specification, there are descriptions of the aim of the systems, glossaries, requirements, and specifications. In the first section, entitled "Introduction," the basic knowledge of the system is explained. In the next section, which is a subsection, the behaviors are explained according to the object. The objects named "Receptionist for stock," "Receptionist for delivery of goods," "Storage," and "Deficiency of stocks" are defined; accordingly, a behavior that sends an event to each object is defined.

1. Introduction
1.1 Goal: Model of Stock Management System
1.2 Scope: Management System of Liquor Shop
1.3 Glossary
Stored object: Commercial product in storage corresponding to each product.
Container: A container loaded with mixed products.
List of stored products: One item from the stored list and memorized correspondence between the stored product and container.
2. Requirements and Specification
2.1 Receptionist for the stock
2.1.1 When a container arrives
2.1.1.1 Making a new container
2.1.1.2 According to each branded product
a. Update the number of stocks to storage
b. Making an item list of the cargo.
2.2 Receptionist for the delivery of goods
2.2.1 When it receives a request for the delivery of goods
2.3 Storage
2.3.1 When it accepts the number of stocks from the receptionist.
2.3.1.1 The number of stocks become "the number of stocks + the number of entry stocks" and update the number to a deficiency of stocks.
2.3.1.2 When the delivery of goods necessarily occurrs because the deficiency of stocks is dissolved.
a. ~
b. ~
c. ~
2.4 Deficiency of stocks
~

Figure 2: Part of the specification with a stored system (underlined specification was added during the collaboration as a modification)

The parts of the sentence that were underlined were added when modifying the specification during the collaborative work to simulate a specification change, which often occurs

in software development and causes difficulties in the development.

# 3 EXPERIMENTS WITH THE COLLABORATION TASK

## 3.1 Experimental Procedure

There were 18 participants recruited for this research—eight Japanese and ten Chinese—and they were organized into six groups. All the groups assumed the role of vendors and each comprised one leader and two workers. A Japanese teacher assumed the role of the client in all the six experiments. In the case of the "Chinese group," all the members were Chinese; in the "Japanese group," all the members were Japanese; and in the "JC group," the leader was Japanese while the two workers were Chinese.

The procedure for the experiment was as follows. In the beginning, for about 15 minutes, the experimenter explained the collaboration task and how they should use the system. She described the specification and a chart of the system from a textbook on software engineering [8]. In the chart, a label object is used to represent an object in the specification and an arrow is used to represent the sending of an event; the arrow is often added to the string using a label object to explain the event. Then, the participants began the task. The total time for the task was fifty minutes. This time was set from the results of two experimenters, who completed the task in fifty minutes, using computers. At the beginning of the task, the participants were informed that the total time for the task was thirty minutes, and that it should be delivered with the specification of the storage system. The A3-sized paper and the pencil were provided for free usage at the same time. When it was twenty minutes into the task, the experimenter, who was the client, informed the participants of a modification to the specification and delivered the modified specification to them.

After completing the task, the participants answered a questionnaire that was based on a five-point scale and checked the relevant difficulty level with regard to their understanding of the specification. If they checked a low value, they were prompted to write a reason for this. Moreover, they were urged to underline the parts they did not understand and provide reasons or opinions for the same.

## 3.2 Environment

The experiments were carried out in two rooms—the faculty room and the research staff room—at the Graduate School of the Japan Advanced Institute of Science and Technology. Skype, a well-known software application that allows voice calls over the Internet, was utilized to communicate between the client and the leaders at the vendor site, primarily for questions and answers. The vendor used a groupware called KUSANAGI [9] to create a system chart. The groupware had a brainstorming tool, grouping tool, and arrow tool to support the grouping stage of the distributed and cooperative KJ Method [10]. A picture of the experimental setting at the vendor site is depicted in Figure 3. A screenshot of a modeling chart is shown in the next section.

In order to create the chart using KUSANAGI, the user labeled the objects or event explanations and depicted the event flow using arrows.

Figure 3: The experimental setting at the vendor site: On the left are the two workers and on the right is the team leader.

# 4 RESULTS AND DISCUSSION

## 4.1 Results of the Collaboration

The six charts of the storage system referred to the specification that was obtained from the experiment. The evalua-

Table 1: Evaluation of model charts and background of the group

| Group name | Chinese Group-A | Chinese Group-B | Japanese Group-A | Japanese Group-B | JC Group-A | JC Group-B |
|---|---|---|---|---|---|---|
| Specification score (correct percentage) | 12 (37.5%) | 23 (71.9%) | 17 (53.1%) | 21 (65.6%) | 4 (12.5%) | 20 (62.5%) |
| Number of labels (correct percentage) | 14 (78.6%) | 24 (95.8%) | 24 (83.3%) | 27 (92.6%) | 17 (35.3%) | 26 (80.8%) |
| Number of arrows (correct percentage) | 8 (100%) | 21 (100%) | 21 (66.7%) | 20 (95.0%) | 11 (36.4%) | 27 (77.8%) |
| Knowledge of IT | 1 | 3 (All) | 3 (All) | 3 (All) | 3 (All) | 2 |
| Learning experience of the specification | 1.7 | 4.0 | 3.7 | 2.7 | 1.7 | 1.7 |

tion points of the charts and the knowledge background along with which team prepared the charts are summarized in Table 1.

We evaluated the charts corresponding to the specification. The specification described in Section 2 has thirty-two lines. We checked the reflection of the contents on each line of the specification. We marked a circle when the line was adequately reflected in the chart, a triangle when the line was reflected to some degree in the chart, and a cross when the line was not reflected in the chart. Two persons performed this evaluation: one was a client and the other, an experimenter. We assigned 1 point to a circle-marked line, 0.5 points to a triangle-marked line, and 0 points to a cross-marked line. The total number of points from all lines implied the evaluation value of the chart. We named this value the "specification score." The marking between two evaluators indicated a high correlation coefficient: 0.75 from each line and 0.99 from each chart.

In addition, we presented the number of labels and arrows in each chart in Table 1. As described in Section 3, the labels represented an object or an event, and the arrows represented an event flow. We checked the correctness of the labels and the arrows by referring to the specification. The procedure for the check is similar to the procedure for the specification score. The score for the evaluation value of labels and that for the evaluation value of arrows are calculated as correct percentage on the Table 1.

The knowledge background of the group was self-reported, indicated by the number of persons who answered "yes" to knowledge on information technology, and the average score from the five-scale questionnaire on software specification. When the value is five, it implies that the person has learned well, and when the value is one, it implies that the person did not learn at all.

In the results, the Chinese Group-B scored about seventy percent and the Japanese Group-B and the JC Group-B got more then sixty percent. The Japanese Group-A continuously scored about fifty percent. The Chinese Group-A scored about forty percent, and the JC Group-A, only about ten percent. The charts of the groups that scored more than sixty percent had many arrows and many labels compared to the other charts. The charts by the Chinese Group-A, which had a very low score, had fewer labels and fewer arrows, and the correctness of representation was inferior. The model chart prepared by the Chinese Group-B, which had a high score, is depicted in Figure 4, and the model chart by the JC Group-A, which had a low score, is depicted in Figure 5.

Before the experiment, we assumed that a knowledge background affects chart making. However, there are no such characteristics with regard to the knowledge background in the results. The experience with software technology did not always lead to good results. This may imply that the proposed task did not require software technology skills such as programming.

## 4.2 Results of the Questionnaire

The results of the questionnaires taken after the collaboration task are described below, and the results with the five-scale evaluation are shown in Table 2. A 5-point evaluation implied very good and 1 point implied very bad. From the questions, Q.8 and Q.9 were given to only the workers. We added a star to the left of the table in the case that showed a significant difference in the ANOVA-analysis between the value of the Japanese and Chinese answers.

Table 2 illustrates the interest level of the participants in the work task, levels of cooperation, communication within their groups, and ability to clearly pose questions to the client.

From the perspective of differences between the Japanese and Chinese, the Chinese tended to be more satisfied with their system charts than the Japanese. The Chinese, who belonged to the Chinese Group-A and the JC Group-A, which had low scores, responded that they were satisfied with their results. These results could lead to confusions in software development, so it is necessary to periodically check if the progress of the development is sound.

According to the questions to the workers, the Japanese participants communicated well with the leaders of their groups, but the Chinese participants felt that their communication was neither good nor bad. The Chinese colleagues could speak their mother tongue among themselves, but they were required to communicate in the Japanese language in the case of the JC Group, which is assumed to be the future style of offshore development. In other cases such as questions to a client, the Japanese participants felt that their communication was good, but the Chinese were indifferent about theirs. It is assumed that the differences in language clearly affected the conscious effort to create cooperative communication.

Table 2: Results of the five-scale questionnaire

| Items | Value |
|---|---|
| Q1: Understandability of the collaboration task | 3.3 |
| Q2: Interest in the task | 3.8 |
| Q3: Pre-image of the system modeling | 3.1 |
| Q4: Understandability of specification | 2.6 |
| Q5: Satisfaction with the chart | 2.5* |
| Q6: Is the work collaborative? | 3.7 |
| Q7: Do you communicate within a group? | 3.6 |
| Q8: Can you question your leader? | 4.6* |
| Q9: Does your leader understand your question? | 4.4* |
| Q10: Do you communicate well with the client? | 3.3* |
| Q11: Can you pose a question to the client? | 4.1 |
| Q12: Does the client understand your question? | 3.7* |
| Q13 Do you feel the barriers imposed by different cultures? | 2.6 |

*Significant difference between the Japanese and Chinese with the ANOVA-test, $p < 0.05$.

Doubts on the specification were highlighted; after the analysis of the experiment, it was found that twenty out of the thirty-two lines were questioned by the participants owing to their lack of comprehension. The common doubts posed by both the Japanese and Chinese participants comprised five lines that included the repetition of words such as "container" or the "deficiency of stocks," phrases that were ambiguous such as "because the deficiency of stocks is dissolved" or "when a delivery of goods necessarily occurred,"

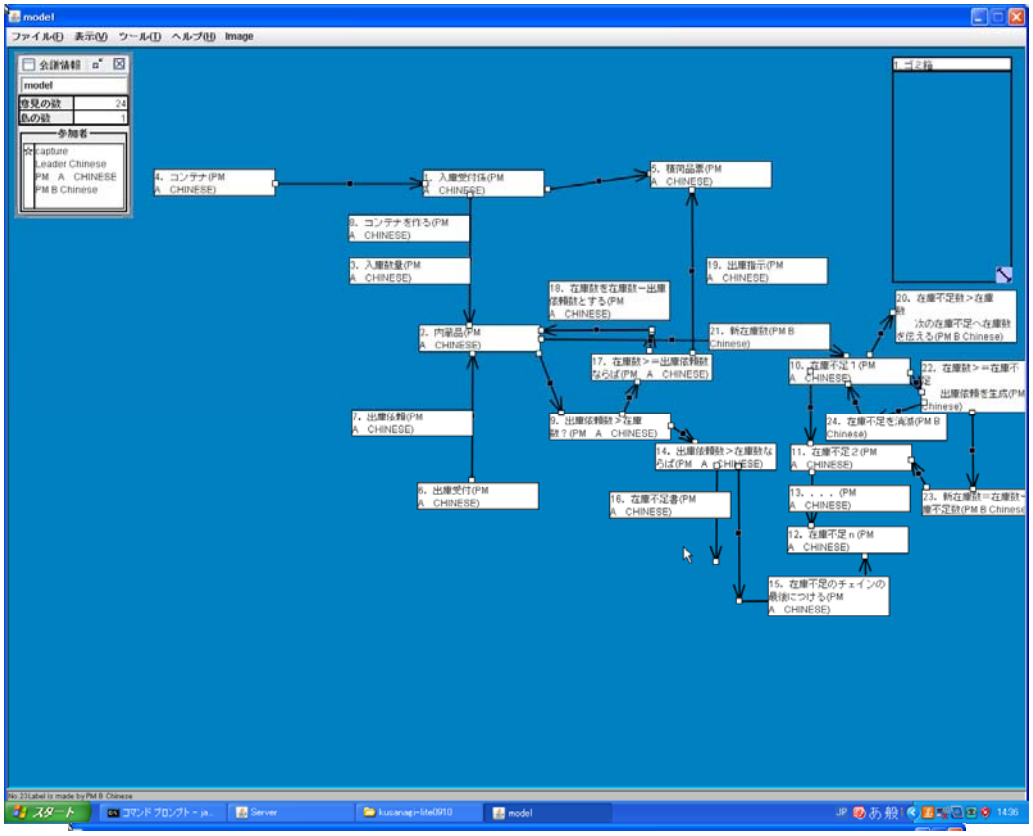


Figure 4: Screenshot of a Good Model Chart created by the Chinese Group-B

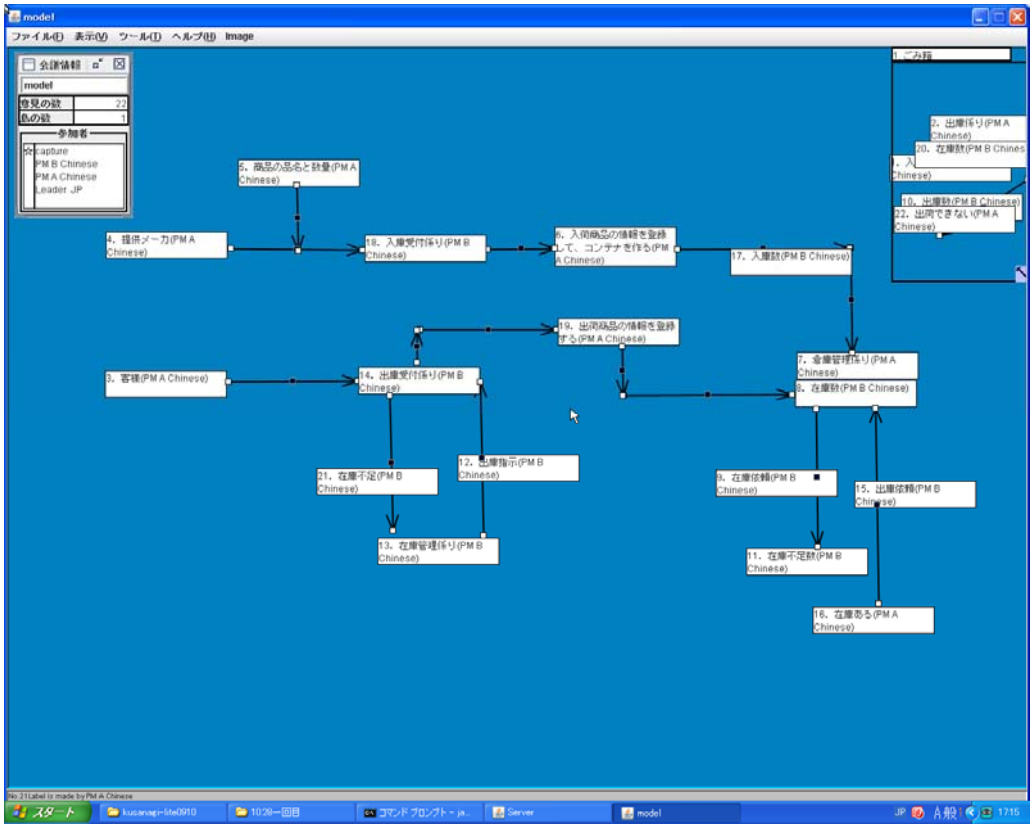


Figure 5: Screenshot of a Poor Model Chart created by the JC Group-A

Table 3: Video observation of the collaboration.

| Group | States of collaboration | States of communication |
|---|---|---|
| Chinese Group-A | The two workers participated individually in the work. One worked on the paper and then on the system. The other worked directly on the system. | Opinions were exchanged between the three persons. One worker was self-assertive. |
| Chinese Group-B | The two workers did not divide the work. One worked on the paper and the other on the system, on the basis of the results of the paper. | The three people collaboratively worked on each line of the specification. They spontaneously exchanged their opinions. |
| Japanese Group-A | Neither worker was given a share of the work. The leader controlled the work sharing, and the workers only worked on the system. | There were a few sets of good communication between the three participants. The leader identified the orders to the workers for each line of the specification. |
| Japanese Group-B | The two workers were each assigned a share of the work. One worked on the paper and the other on the system, on the basis of the results from the paper. | One worker was silent while the other actively and successfully communicated with the leader. |
| JC Group-A | Each of the two workers handled a share of the work. The two first worked on the paper and then on the system. | There were exchanges of opinion between the leader and workers, but only a few between the workers. One worker was self-assertive, while the other asserted negative words. |
| JC Group-B | Each of the two workers was assigned a share of the work. The two directly worked on the system. | There was communication between the three participants. One worker was self-assertive, and the other asserted less qualitative opinions. |

and technical terms such as “chain.” In addition, for the Chinese, their characteristic doubts stemmed from the definition of words, such as “container,” “stored object,” and “deficiency of stocks”; ambiguous representations, such as the “next deficiency of stocks” and “oneself”; and the obscurity of outputs, such as “output a document about deficiency of stocks” and “create a request for delivery of goods.”

The abovementioned Chinese participants had a disadvantage because of the use of Japanese, which is not their mother tongue. The Japanese participants should pay attention to clear communication in the Japanese language to ensure good collaboration.

## 4.3 Observation of Collaboration

We described a state of collaboration in Table 3 by using a video analysis of the participant’s cooperativeness and communication.

Drawing attention to the Chinese workers, the self-assertiveness of some of them stands out. This might reflect a cultural habit that the Chinese are more individualistic than the Japanese, who prefer homogeneity. Naturally, the cooperative Chinese group created the good chart.

The Chinese Group-A, JC Group-A, and JC Group-B had less learning experience with software specification. From these groups, the JC Group-B, who created a good chart that earned more than sixty percent, worked by having good communication between the three members and by sharing the work on the computer screen. On the other hand, the Chinese Group-A and JC Group-A had to balance the work by creating parts of the chart individually first and then combining them.

Using these observations, the group that scored the lowest points had smaller amounts of work sharing and some of them were self-assertive; that is to say, they did not collaborate in the task. This implies that good collaboration favors the success of a proposed task.

In this experiment, we allowed individual work on paper but urged the usage of the shared environment and groupware technology to encourage shared work. This can be explored by combining the consideration of effectiveness and shared work in software development.

# 5 CONCLUSION

In this paper, we design a laboratory experiment to simulate offshore software development between the Japanese and Chinese. We performed an experiment in which the participants prepared a system chart instead of program development, using the prepared specification. We considered three types of collaborations, such as an offshore type, domestic type, and a more collaborative type.

The results of these experiments were as follows:

(1) The most collaborative team produced a good model chart.

(2) The Chinese participants faced difficulty in using the Japanese language to communicate, and some of them were self-assertive.

(3) The Chinese participants, compared to the Japanese, tended to be satisfied with their results of the model chart, considering them neither good nor bad.

In the future, we will consider an interface that elicits a collaborative mind or considers the cultural habitat. And, it should be expected to execute the experiments with not natural language but also formal specification language like UML [4]. In addition, more investigations on collaborations in the software development process will be issued.

## REFERENCES

[1] H. Tuji, T. Moriyasu and T. Mori, "Evolution of Offshore Software Development and Tacit Knowledge of Engineering," IPSJ Magazine, Vol.49, No.5, pp. 551-557 (2008) (in Japanese).

[2] MIC in Japan, "Report of Progress of Offshoring and its Effect" (2007) (in Japanese).

[3] D. Perry, N. Staudenmayer, and L. Votta, Jr., "Understanding and Improving Time Usage in Software Development," In Trends in Software. Vol. 5, Software Process, A. Wolf and A. Fuggetta, Eds. John Wiley and Sons, New York (1995).

[4] T. Nakahara and H. Fujino, "The Application of UML in the Offshore Project," Journal of the Society of Project Management, Vol. 10, No. 6, pp. 9-14 (2008) (in Japanese).

[5] T. Yamazaki, "Explanation of the Programming Design Method with the Open Problem," IPSJ Magazine, Vol. 25, No. 9, p. 934 (1984) (in Japanese).

[6] Japan Users Association of Information Systems, "A Guideline to Requirement Specification" (UVC) (2007) (in Japanese).

[7] Y. Ooki et al., "Description of Stock Management System by Concurrent Prolog," IPSJ Magazine, Vol. 26, No. 5, pp. 470-476 (1985) (in Japanese).

[8] T. Nakadokoro, "Software Engineering 2nd Edit," pp. 42-46, Asakura-syoten (2004) (in Japanese).

[9] T. Yuizono and J. Munemori, "Groupware for a Knowledge Creative Process of a Research Group and its Application to Externalization and Combination Steps," Journal of IPSJ, Vol. 48, No. 1, pp. 30-42 (2007).

[10] J. Munemori and Y. Nagasawa, "GUNGEN: groupware for a new idea generation support system," Inf. and Soft. Technol., Vol. 38, No. 3, pp. 213-220 (1996).

## ACKNOWLEDGEMENTS

This research was partially supported by The Ministry of Education, Culture, Sports, Science and Technology (MEXT) and the Grant-in-Aid for Young Scientists (B) 21700133, 2010.

(Received August 25, 2010)
(Revised April 26, 2011)

**Takaya Yuizono** received the B.E., M.E., and Dr. of Engineering from Kagoshima University, in 1994, 1996, 1999, respectively. He was a research associate in Kagoshima University, a lecturer and an associate professor in Shimane University, respectively. He has been an associate professor at School of Knowledge Science, Japan Advanced Institute of Science and Technology since 2005. His research interests include in groupware, computer-supported cooperative work and knowledge medium. He received KES2005 Best paper award in 2005. He is a member of ACM, IEEE, IPSJ and IEICE.

**Lihua Xuan** received the bachelor of Japanese from Dalian Nationalities University in 2007 and master degree of School of Knowledge Science from Japan Advanced Institute of Science and Technology in 2010. Now work in Tagami EX Co.,Ltd. Her interests were cross-cultural collaboration in business.

# P2P-based WSN Data Sharing System using B+Tree

Nobuhiko Matsuura†, Hiroshi Mineno‡, Ken Ohta††, Norio Shiratori*, and Tadanori Mizuno**

†Graduate School of Informatics, Shizuoka University, Japan
‡Faculty of Informatics, Shizuoka University, Japan
††Research Laboratories, NTT DOCOMO, Japan
* Research Institute of Electrical Communication, Tohoku University, Japan
** Graduate School of Science and Technology, Shizuoka University, Japan
{matsuura, mineno, mizuno}@mizulab.net

***Abstract*** - A context-aware service that uses sensing data has attracted attention, along with the development of wireless technology and sensor technology. To provide these services, the sensing data sharing system in P2P networks needs to cope with a vast amount of data. However, existing algorithms do not respond to varying the number of sensing data types. In addition, most existing algorithms cannot execute reverse key resolutions because their search algorithms need to include specific data as the key in the query. To address these issues, we propose a multi-dimensional range search algorithm in P2P networks that uses a B+tree for an efficient search with an arbitrary number of sensing data types.

***Keywords***: P2P, B+tree, range search, multi-dimensional search, wireless sensor network

## 1 INTRODUCTION

Peer-to-Peer (P2P) networks are emerging as a new paradigm for structuring large-scale distributed systems. In these systems, resources are associated with keys, and each peer is responsible for a subset of the key to guarantee scalability performance, fault-tolerance, and robustness. P2P network have been developed to have a more suitable and practical design for applications, such as those in Building Monitoring [1] and Sewer Snort [2].

One of the systems that is suitable for using P2P is a sensing data sharing system, such as WSN with P2P [3]. A context-aware service has attracted attention, along with the development of wireless technology and sensor technology. This service can offer a remarkable transformation that considers user location and conditions using sensor data. To provide these services, the system needs to manage data from wireless sensor networks (WSNs). P2P networks are thought to be the answer to cope with the vast amount of data from WSNs because of their potential.

WSNs have some dynamic properties such as varying the data values, the total number of data, and the number of data property (such as Temperature and Humidity). In other words, P2P must deal with these properties to perform as general middle-ware for a WSN data sharing system because what it takes to provide service differs depending on the service. In particular, we are sure that considering the varying numbers of data property can improve the search performance because the sensor types will likely rapidly increase as the fundamental technology is developed. However, other works in P2P networks cannot deal with the varying number of data properties. These algorithms must include specific data, such as location information, as a key in the query. Therefore, they cannot execute a reverse resolution of keys and can only use a limited number from a vast number of sensor types in the query condition. To solve this problem, we need a multi-dimensional algorithm without a special property on the data store. "Multi-dimensional" denotes the tabular form in data storage, and the query condition number dynamically increases or decreases on demand. In this paper, we extend our previous work [4] and show simulation results to demonstrate the potential of our algorithm as a WSN data sharing system.

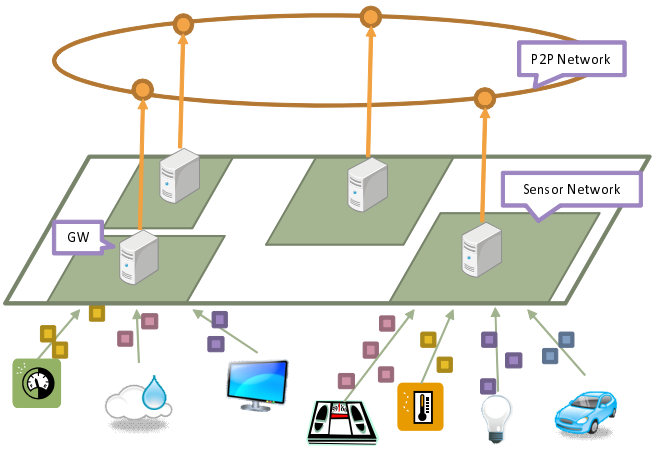



Figure 1: Sensing Data Sharing System

The rest of the paper is organized as follows: Section 2 discusses related work; Section 3 presents our previous work and the problem of bottleneck; Section 4 shows the experimental results and discussion; finally, Section 5 concludes the paper.

## 2 RELATED WORK

Many architectures have been developed on sensing data sharing systems using P2P. These architectures arrange Gateways (GWs) in WSNs to manage data transfers at each WSN and to automatically construct structuring P2P networks at each GW, an drown in Fig. 1. In general, structuring P2P networks are constructed using distributed hash tables (DHTs), but they have a major limitation in that they can only support an exact-match search. P2P networks need to have the exact key of a data item to store that item in the responsible node.

Table 1: NoSQL Categorization

| Name | Data Model | CAP Theorem | Distribute Model | Persistence Model |
|---|---|---|---|---|
| Cassandra | Column-oriented | AP | Consistent Hash | Memtable/SSTable |
| HBase | Column-oriented | CP | Sharding | Memtable/SSTable on HDFS |
| CouchDB | Document-oriented | AP | Consitent Hash | Append-only B-tree |
| Riak | Document-oriented | AP | Consistent Hash | ? |
| MongoDB | Document-oriented | CP | Sharding | B-tree |
| Tokyo Cabinet | Key-value | AP | Consistent Hash | Hash or B-tree |
| Voldemort | Key-value | AP | Consistent Hash | Pluggable |
| Redis | Key-value | CP | Consistent Hash | In-memory with background snapshots |
| Scalaris | Key-value | CP | Consistent Hash | In-memory only |

Because the exact key is given by a hash function, the key has no order relation, and the user cannot search flexibly such as when processing a range query or a multi-dimensional query.

In P2P networks, the additional idea of flexible searching must be applied and must often use location information. Znet [5] uses Z-ordering of space-filling curves [6] to partition the 2-dimensional ID space of the location and map space ID onto corresponding nodes, and it uses a Skipgraph [7] to manage the network topology. Mill [8] uses the same partitioning and mapping technique, but it uses Chord to manage the network topology. LL-Net [9] partitions a space into 4 blocks recursively, and the quad tree is used to manage the network topology. These architectures can process a range query by using location information and also process a multi-dimensional query by adding other properties to the location ID space as an additional dimension. These P2P architectures have a limitation that involves including the location properties as a key in the query. In other words, they cannot execute a reverse resolution of the location information and also cannot deal with the varying number of properties without a reboot of the entire system because it is assumed that the number of dimensions, which represents properties, is static.

On the other hand, Skipgraph can process a flexibly search without location information. Skipgraph constructs a doubly-linked list of inserted keys at each layer according to the membership vector. The membership vector is an identifier like NodeID, and is allocated to all nodes. A certain key of the doubly-linked list in the i-layer has links between the closest key, which is defined by not only numerical distance but also i-length prefix matching of the membership vector. Skipgraph can process a range query by the sorted doubly-linked list and also process a multi-dimensional query by constructing multiple P2P networks. However, the link of Skipgraph depends on the value of key. It is difficult to maintain robustness when we use real-time and high-density data, such as sensor data.

NoSQL, which means “Not Only SQL”, is another efficient way to store and search data. NoSQL behaves as database, and the major example is key-value store, column-oriented database, and document-oriented database. Key-value store is often implemented on P2P networks. They are great hopes becoming the solution about the problem of relational database (RDB), such as scalability and availability. Today, we can select one of what can serve our purpose from a lot of NoSQL; however, we often think “What should I use?”. Table 1 shows a part of NoSQL categorization list. In the table, CAP theorem says it is impossible for a distributed computer system to simultaneously provide all of the three guarantees (Consistency, Availability, and Partition Tolerance) and a distributed system can satisfy any two of these guarantees at the same time.

There are many work for categorization and comparing NoSQL, such as [10]. Cassandra [11] and HBase [12] are the most famous systems of NoSQL. Cassandra is the Apache [13] project and has a goal: develops a highly scalable second-generation distributed database, bringing together Dynamo [14] distributed design and Bigtable [15] column-oriented data model. We can insert the data which contained key space, column family, key, column, and value in Cassandra like RDB’s record. HBase is also Apache project and behaves as the database of Hadoop [16]. This goal is the hosting of very large tables, such as X billions of rows and Y millions of columns, atop clusters of commodity hardware. We can insert the text, such as log file, in the Hadoop Distributed File System which partition the file and distribute the part of file in multiple server. It is said that Cassandra and HBase have higher availability and scalability than RDB. However, they need to construct the distributed system on static network, such as data center. Thus, they are not good way to construct the distributed system includes the home GW.

# 3 THE DESIGN OF MULTI-DIMENSIONAL ALGORITHM

## 3.1 Overview

We are designed multi-dimensional algorithm in our previous works, and this algorithm uses a B+tree [17] for an efficient search with an arbitrary number of sensing data type. Our algorithm introduces the idea of building as many tree structures as there are properties to process a dynamic multi-dimensional range search, and these tree nodes (tree-nodes) are mapped to the node on P2P networks (P2P-nodes). The user can select an arbitrary number of trees to process a multi-dimensional search, and a conclusive result is obtained by merging all the results. Therefore, the reverse resolution of certain properties, such as location, can be done by using the

Table 2: Node Information

| Node (NodeID) | temp |
|---|---|
| $N_0$ (5) | 21 |
| $N_1$ (3) | 30 |
| $N_2$ (1) | 15 |
| $N_3$ (0) | 27 |
| $N_4$ (4) | NULL |
| $N_5$ (2) | 29 |
| $N_6$ (7) | 18 |
| $N_7$ (6) | 24 |

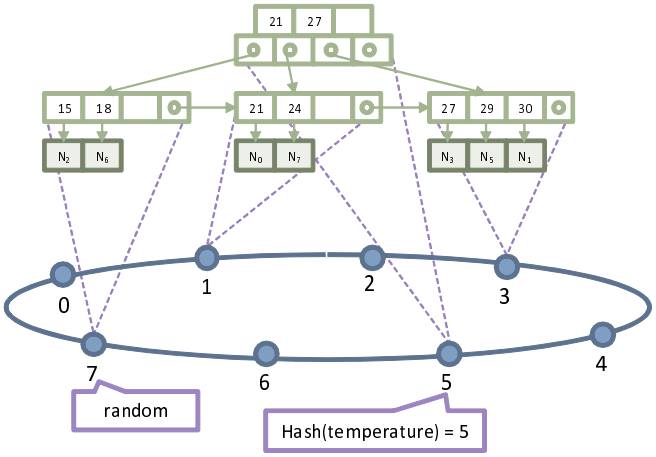



Figure 2: Mapping to P2P Networks

composition query, and the system can deal with increasing and decreasing properties while maintaining the running state.

To build a strong tree structure for frequently varying data values, such as sensor data, our algorithm uses a B+tree, which is a balanced tree, and the serch and insert order provides $O(log_t N)$ search cost, where N is the number of peers in the system and t is the constant number depending on list size of tree-nodes. The B+tree acts as a logic structure to make a comparison across property values. A network topology uses a P2P network to store the tree-nodes. Mapping from the tree-node to the P2P-node is necessary. In our algorithm, this mapping uses a hash function. Mapping is random in most tree-nodes, but the root of the tree should be uniquely known to all nodes because the search for the B+tree should obtain the root in the beginning. Therefore, the root has a mapping rule wherein the map from the tree-node to the P2P-node of $NodeID = Hash(PropertyName)$ acts as the manager node, and other node map to the P2P-node of $NodeID = Hash(Random)$. Based on this analysis, when the nodes listed in Table 2 showing a set of nodes and their properties join the P2P networks, mapping is done as in Fig. 2. Of course, this algorithm can deal with more properties by building a new tree structure.

## 3.2 Algorithm

If a new node join a P2P network, it should join the P2P network and insert its shared resource information, such as properties and its own NodeID, into a tree-node on the P2P-node. Join and leave algorithm is used according to the P2P algorithm. For example, the join algorithm of Chord is used when we use the Chord algorithm to construct a base P2P network. This put process shown as Algorithm 1. The node obtains a hash value based on property name and sends an insert request to the manager node that has the same hash value. The received node relays the request to the child tree-node that includes the request key between the tree ranges. On the other hand, the node creates a root for the corresponding tree if the tree-node does not exist. By repeating this recursive operation, the node in the target tree finally finds the destination leaf and inserts the data into the appropriate place.

The remove algorithm is shown as Algorithm 2, and a large part of it is constructed similarly to the put algorithm.

The get algorithm is shown as Algorithm 3. The algorithm also searches tree-nodes like the put and remove algorithms, but this algorithm uses a range key and an iterative search. There are two cases of supporting range key search in com-

**Algorithm 1** Put(targetHash, propertyValue, timeStamp)

**Require:** targetHash is closest to myNodeID
  $storedValue \leftarrow SearchLocal(targetHash)$
  **if** $storedValue = null$ **then**
    $PutLocal(targetHash, propertyValue, timeStamp)$
  **else**
    **if** $!storedValue.isLeaf()$ **then**
      **for all** $c$ such that $storedValue.getChild()$ **do**
        **if** $(c.min \leq propertyValue) \&\& (propertyValue < c.next.min)$ **then**
          $Put(c.getNodeID(), propertyValue, timeStamp)$
        **end if**
      **end for**
    **else**
      **if** $storedValue.isFull()$ **then**
        $Separate(storedValue)$
      **end if**
      $PutLocal(targetHash, propertyValue, timeStamp)$
    **end if**
  **end if**

**Algorithm 2** Remove(targetHash, propertyValue)

**Require:** targetHash is closest to myNodeID
  $storedValue \leftarrow SearchLocal(targetHash)$
  **if** $storedValue! = null$ **then**
    **if** $!storedValue.isLeaf()$ **then**
      **for all** $c$ such that $storedValue.getChild()$ **do**
        **if** $(c.min \leq propertyValue) \&\& (propertyValue < c.next.min)$ **then**
          $Remove(c.getNodeID(), propertyValue)$
        **end if**
      **end for**
    **else**
      **if** $storedValue.contain(propertyValue)$ **then**
        $RemoveLocal(targetHash, propertyValue)$
      **end if**
    **end if**
  **end if**

**Algorithm 3** Get(targetHash, min, max)

```
treeNode ← dht.get(targetHash)
result ← {}
for all e such that treeNode.getElements() do
  if treeNode.isLeaf() then
    if (min ≤ e.key)&&(e.key ≤ max) then
      result ← result ∪ e
    end if
  else
    if (min ≤ e.max)&&(e.min ≤ max) then
      r ← Get(e.hash, min, max)
      result ← result ∪ r
    end if
  end if
end for
return result
```

paring the search request with its own tree-node's information. Case 1 is a comparison of the leaf; this selects the data included between the request ranges. Case 2 is a comparison of the other place; this selects the node that includes the request range between its own tree ranges as the next candidate. The get algorithm repeats case 2 to find destination leaf-nodes, and it executes case 1 to obtain the result. We used an iterative search to repeat case 2 because a recursive search risks a fatal decrease in performance in the target environment, such as the general middle-ware for a WSN data sharing system. The tree-node needs to split the request into the same number of branches if two or more candidates are found, and then the tree-node must wait for the under-layer processing to finish or timeout in accordance with the recursive search. An iterative search can prevent this risk because it forces the node that sent the search request to wait.

## 3.3 The Problem of Previous Work

Actual sensor data occurs with great frequency because a sensor is generally used to obtain real-time and high-density data. The first thing to do in the these algorithms is to access the root of the tree structure, and the root tends to have a heavier workload than other nodes. Therefore, the root of the tree structure may become a bottleneck in the system. A bottleneck arises from insufficient disk space and memory, insufficient CPU capacity, and an exclusive control method.

- Insufficient disk space and memory
  When number of inserted data is over allowable number, the node frequently causes the Swapping and performance decreases. In addition, the node which too much data is concentrated greatly decreases performance in searching stored data. This is problem in a general relational database (RDB), but the Key-Value store (KVS) on P2P networks is not actually considered as problem. The KVS can easily deal with the increasing data by distributing the data to many nodes. This is called scale-out technology, therefore, it is not considered in this paper.

- Insufficient CPU capacity
  When the node receives requests, CPU use and processing time increases according to the request type and the current state, and it becomes a problem when the number of requests increases up to a certain number. A typical KVS can resolve this problem because it can distribute not only data but also the load of the processing request. However, the multi-dimensional algorithm on a KVS cannot resolve the problem well because it constructs a large tree structure. This structure can support the tabular form on KVS and is guaranteed to reach the target, but it limits the route to certain data and concentrates a local load. In particular, the nodes included in the route, such as the root, has a larger workload than other nodes.

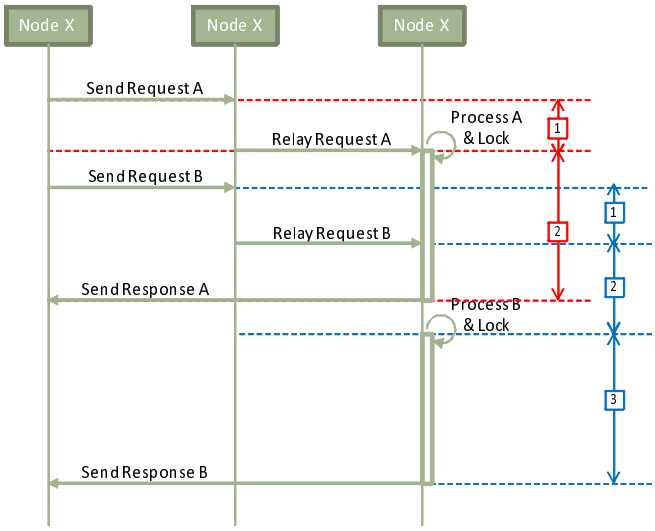


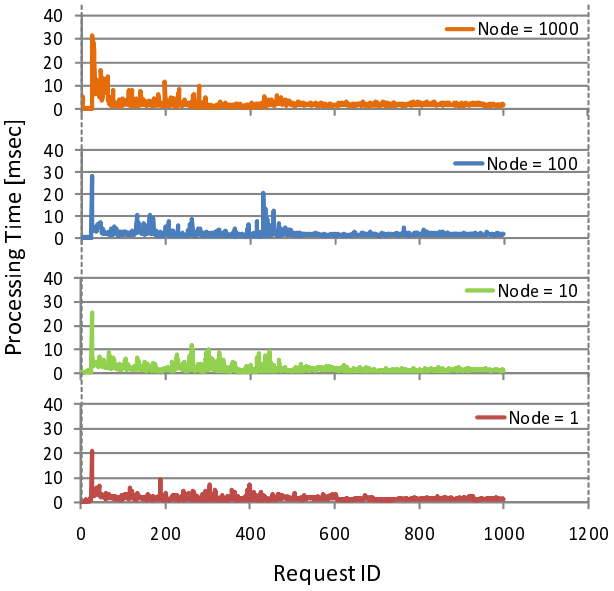

Figure 3: PutRequestSequence

- Exclusive control method
  This factor is an endemic problem of distributed systems and fatal problem of this architecture. The KVS is used by many users, and operation demand may occur around the same time. The structure breaks if operation is executed in parallel because it constructs a tree structure, as previously mentioned. To address this problem, an exclusive control method, such as lock, is needed. However, this method has a bad affects the processing time of the request. In other words, it is important to reduce the maximum number of concurrent connection to maintain throughput.

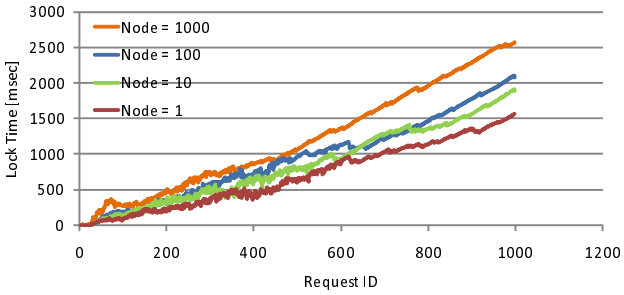
Based on the above analysis, it is necessary to evaluate the process time and the lock before being applied to the WSN data sharing system. In addition, we pay attention to the exclusive control method problem that is the biggest weakness.

# 4 PERFORMANCE EVALUATION

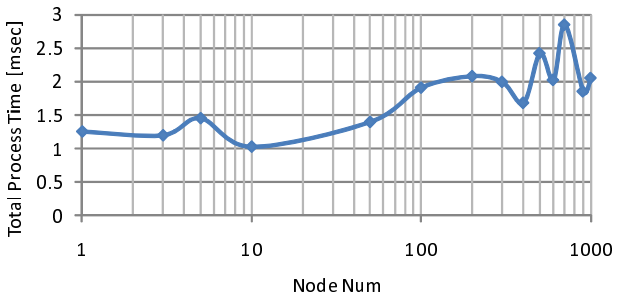
In this section, we describe the evaluation of our algorithm in delay time. There are three types of communication, lock, and processing delay time. Fig. 3 shows the sequence of processing the put request. The put request is sent from $Node_X$ to $Node_Z$. When $Node_X$ sends request A, $Node_Y$ receives



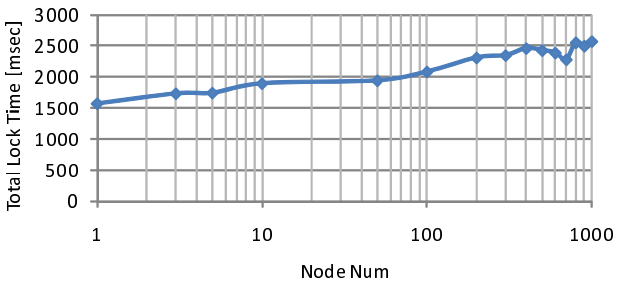

(a) Processing Time




(b) Lock Time




(c) Total Processing Time




(d) Total Lock Time

Figure 4: The Result Varying Number of Node

Table 3: Simulation Environment

| Varying | Node | Data |
|---|---|---|
| Num Nodes | 1 – 1000 | 100 |
| Num Request | 1000 | 1 – 10000 |
| Data Distribution | Normal | Normal |

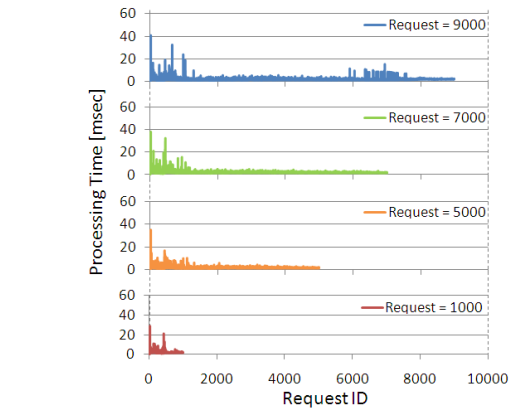
the request once and relays it to $Node_Z$. The communication and processing time occur in this sequence. Communication time is the time between sending and receiving of the request No.1 in the figure. Processing time is the time between processing of the request No.3. Lock time occurs in processing of request B. Request B is sent like request A, and lock time occurs by waiting for the processing of the request B. Lock time is the time between receiving and the start of processing No.2.

We evaluated the lock and processing times when a request is sent to one root node just around the same time. We did not evaluate the communication time because we wanted to consider only root performance. The evaluation was conducted with simulations using Overlay Weaver [18] in the conditions listed in Table 3. The list size of the tree-node in our algorithm was adjusted in the experiment to 25, and this algorithm used Chord to construct the base P2P networks.

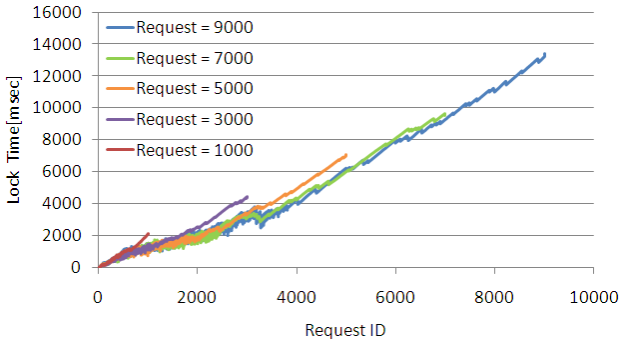
Fig. 4(a), 4(b) show the history of processing each request that has a sequential unique ID, where Node is the number of node in a P2P network. Fig. 4(c), 4(d) show the total time for processing the last request at each node. Fig. 5(a), 5(b) also show the history of processing each request that has a sequential unique ID, where Request is the number of simultaneously inserted requests. Fig. 5(c), 5(d) show the total time for processing the last request at each node.

Fig. 4(a) has place where processing time suddenly increases. The reason is that the B+tree structure must divide a full leaf, create new a leaf to store the leaf information, and increase the number of layers on the tree structure in addition to normal operation. These additional operations occur when the number of leaves becomes more than $25^i$ in an ideal environment, where 25 is adjusted as the leaf size. These additional operations also affect Fig. 4(b). The part near the origin point in Fig. 4(b) has a very low delay time, and the time suddenly increases when requests ID exceede 25 because the root can process the up to 25 request in only oneself. When the Fig. 4(c), 4(d) is considered, the effect of node number is small because the order is $O(logN)$, where N is the number of nodes. The reason of this order is that we constructed the tree structure on a P2P networks with Chord algorithm. The Chord algorithm search cost is $O(logN)$, and we used Chord search method to find the appropriate child node on the tree structure, and the effect of the Chord search order appeared in the result. We are certain that our idea, building a large structure on P2P networks to handle a multi-dimensional as our work, have enough scalability.

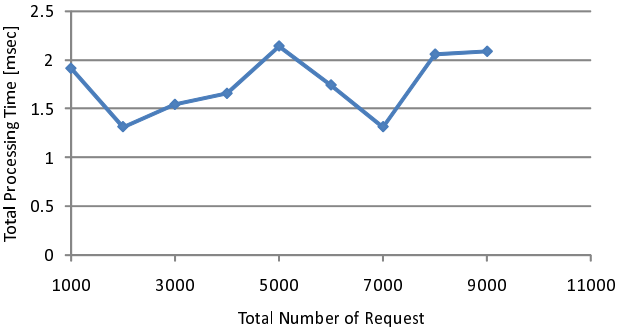
Fig. 5(a) shows a similar change as in Fig. 4(a), and the change in Fig. 5(c) is flat because rapid change occurs by dividing and creating leaves when the leaf is full. In Fig. 5(b), the total number of varying requests does not affect the lock time, and the result linearly increases, as shown in Fig. 5(d).



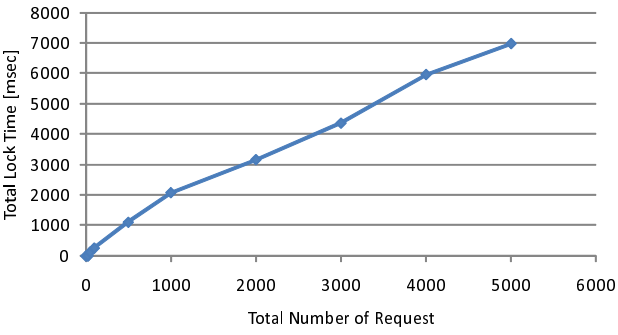

(a) Processing Time




(b) Lock Time




(c) Total Processing Time




(d) Total Lock Time

Figure 5: The Result Varying Number of Request

Because the request goes into a queue once, even if the system receives any number of request at any time. On the other hand, Fig. 5(d) shows throughput that can process $x$ request by $y$ msec from point of view of the system side. We can obtain the throughput that guarantees to process as soon as receiving request from the intersection which figure with expression $y = 1000$, and throughput is about 500 requests per second. This throughput is not too few because the traditional RDB's default number of concurrent connections is from 100 to 500 on demand and the some systems often remain at the default number.

## 5 CONCLUSION

We designed a multi-dimensional search algorithm for P2P networks using a B+tree. Our novel P2P index structure is well suited for applications, such as a sensing data sharing systems by supporting range and multi-dimensional queries. This structure is in accordance with the basic key idea that a tree structure, such as a temperature-tree, builds on P2P networks for each property.

We evaluated the performance of the proposed algorithm using simulations. From the simulation results, we found that the proposed algorithm has scalability because the processing and lock times are order $O(logN)$. In addition, the throughput that guarantees to immediately process the request is about 500 requests per second.

In the future, we will aim at the performance gain of the algorithm. We can improve the performance using the replication or cache algorithm. The replication may distribute the workload of the root and decrease the lock time. Cache algorithm may dramatically improve the performance because the number of request relays becomes 1 when the node has visited by search algorithm in the past. Additionally, both algorithm can become churn tolerant improvement techniques. Churn will cause a serious problem for the P2P put algorithm: The data of node is removed due to the continuous process of node arrival and departure. To use replication and cache algorithm, at least one node can hold inserted data and the system can recover from churn damage by using the data.

## REFERENCES

[1] M. Ceriotti, L. Mottola, G. Picco, A. Murphy, S. Guna, M. Corra, D. Zonta and P. Zanon, "Monitoring Heritage Buildings with Wireless Sensor Networks: The Torre Aquila Deployment," Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN'09) (2009).

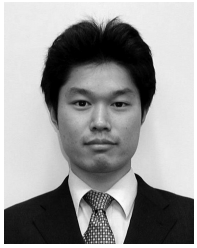
[2] J. Kim, J. Lim, J. Friedman, U. Lee, L. Vieira, D. Rosso, M. Gerla and M. Srivastava, "SewerSnort: A Drifting Sensor for In-situ Sewer Gas Monitoring," Proceedings of the 6th Annual IEEE communications society conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'09) (2009).

[3] G. Gutierrez, B. Mejias, P. Roy, D. Velasco and J. Torres, "WSN and P2P: A Self-Managing Marriage," Proceedings of the 2nd IEEE International Conference on

Self-Adaptive and Self-Organizing Systems Workshops (SASOW'08) (2008).
[4] N. Matsuura, H. Mineno, N. Ishikawa and T. Mizuno, "Evaluation of B+Tree-based Multi-dimensional Range Search Algorithm for P2P Networks," Proceedings of the 20th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT'10) (2010).
[5] Y. Shu, B. Ooi, K. Tan and A. Zhou, "Supporting Multi-dimensional Range Queries in Peer-to-Peer System," Proceedings of the 5th IEEE International Conference on Peer-to-Peer Computing (P2P'05) (2005).
[6] J. Wierum, "Logarithmic Path-Length in Space-Filling Curves," Proceedings of the 14th Canadian Conference on Computational Geometry (CCCG'02) (2002).
[7] J. Aspnes and G. Shah, "Skip Graphs," ACM Transactions on Algorithms (2007).
[8] S. Matsuura, K. Fujikawa and H. Sunahara, "Mill: Scalable Area Management for P2P Network based on Geographical Location," Proceedings of the 12t Annual Scientific Conference on Web technology, New Media, Communications and Telematics theory, Methods, Tools and Applications (Euromedia'06) (2006).
[9] Y. Kaneko, K. Harumoto, S. Fukumura, S. Shimojo and S. Nishio, "A location-based peer-to-peer network for context-aware services in a ubiquitous environment," Proceedings of the Symposium on Applications and the Internet Workshops (SAINT'05) (2005).
[10] B. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan and R. Sears, "Benchmarking cloud serving systems with YCSB," Proceedings of the 1st ACM symposium on Cloud computing (SoCC'10) (2010).
[11] A. Lakshman and P. Malik, "Cassandra: a decentralized structured storage system," Proceedings of the 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware (LADIS'09) (2009).
[12] HBase, http://hbase.apache.org/.
[13] The Apache Software Foundation, http://www.apache.org/.
[14] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall and W. Vogels, "Dynamo: Amazon's Highly Available Key-value Store," Proceedings of the 21st ACM SIGOPS symposium on Operating systems principles (SOSP'07) (2007).
[15] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes and R. Gruber, "Bigtable: A Distributed Storage System for Structured Data," Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI'06) (2006).
[16] Hadoop, http://hadoop.apache.org/.
[17] D. Comer, "The Ubiquitous B-Tree," ACM Computing Surveys (1979).
[18] Overlay Weaver, http://overlayweaver.sourceforge.net/.

(Received August 24, 2010)
(Revised April 26, 2011)

**Nobuhiko Matsuura** received the B.I. degree in Informatics from Shizuoka University, Japan in 2010. He is currently working towards the M.I. degree at Shizuoka University. In 2010, he received Best paper award at the International Workshop on Informatics. His current research interests include Internetworking, distributed systems, Peer-to-Peer networks, and databases.



**Hiroshi Mineno** received his B.E. and M.E. degrees from Shizuoka University, Japan in 1997 and 1999, respectively. In 2006, he received the Ph.D. degree in Information Science and Electrical Engineering from Kyushu University, Japan. Between 1999 and 2002 he was a researcher in the NTT Service Integration Laboratories. In 2002, he joined the Department of Computer Science of Shizuoka University as an Assistant Professor. His research interests include sensor networks as well as heterogeneous network convergence. He is a member of IEEE, ACM, IEICE, IPSJ and Informatics Society.

**Ken Ohta** received B.E., M.E. and Ph.D. degree from Shizuoka University, Shizuoka, Japan, in 1994, 1996 and 1998. In 1999, he joined NTT DoCoMo, Inc. His current research includes mobile terminal architecture, mobile application, and mobile security. He is a member of IEICE.

**Norio Shiratori** received his doctoral degree from Tohoku University in 1977. After that he served as an Associate Professor and Research Associate at Research Institute of Electrical Communication (RIEC), Tohoku University. He was also the Professor of Information Engineering at Tohoku University from 1990 to 1993. Now he is the President of IPSJ (Information Processing Society of Japan) and served as an IFIP representative of Japan. Now, he is an Emeritus and Research Professor at RIEC, Tohoku University, Japan. His research interests include Ubiquitous and Symbiosis computing. He is the fellow of IEE, IPSJ and IEICE.

**Tadanori Mizuno** received the B.E. degree in industrial engineering from the Nagoya Institute of Technology in 1968 and received the Ph.D. degree in computer science from Kyushu University, Japan, in 1987. In 1968, he joined Mitsubishi Electric Corp. Since 1993, he is a Professor of Shizuoka University, Japan. Now, he is a Professor of graduate school of Science and technology of Shizuoka University. His research interests include mobile computing, distributed computing, computer networks, broadcast communication and computing, and protocol engineering. He is a member of IEEE, ACM, IEICE, IPSJ and Informatics Society.

## 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:

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