# International Journal of Informatics Society

09/16 Vol. 8 No.2 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

Tomoya Kitani

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

We are delighted to have the twenty-third issue of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Ninth International Workshop on Informatics (IWIN2015), which was held at Amsterdam, Netherlands, Sep. 6-9, 2015.The workshop was the ninth 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 24 papers were presented in five technical sessions. The workshop was successfully finished with precious experiences provided to the participants. It highlighted the latest research results in the area of informatics and its applications that include networking, mobile ubiquitous systems, data analytics, business systems, education systems, design methodology, intelligent systems, groupware and social systems.

Each paper submitted IWIN2015 was reviewed in terms of technical content, scientific rigor, novelty, originality and quality of presentation by at least two reviewers. Through those reviews 14 papers were selected for publication candidates of IJIS Journal, and they were further reviewed as a Journal paper. This volume includes five papers among the accepted papers, which have been improved through the workshop discussion and the reviewers’ comments.

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

**Tomoya Kitani** is an associate professor at the College of Informatics, Academic Institute, Shizuoka University from 2013 after begin an assistant professor at the Graduate School of Information Science, Nara Institute of Science and Technology since 2005 to 2008, and being an assistant professor at the Division of Global Research Leaders, Shizuoka University from 2008 to 2013. He received his Ph.D. in Information Science and Technology from Osaka University in 2006. His research interests include computer algorithms, mobile computing, sensor networks, intelligent transport systems (ITS), and Bikeinformatics. He is a member of the Information Processing Society of Japan (IPSJ), the Institute of Electronics, Information and Communication Engineers (IEICE), the Institute of Electrical and Electronics Engineers (IEEE) and the Society of Automotive Engineers of Japan (JSAE). He is also a committee member of SIG-ITS of IPSJ, SIG-DPS of IPSJ and Division of Motorcycle Dynamics of JSAE.

# Is It Possible for the First Three-Month Time-Series Data of Views and Downloads to Predict the First Year Highly-Cited Academic Papers in Open Access Journals?

Hiroshi Ishikawa*, Masaki Endo*, Iori Sugiyama**, Masaharu Hirota***, and Shohei Yokoyama****

*Graduate School of System Design, Tokyo Metropolitan University, Hino, Japan

**Faculty of System Design, Tokyo Metropolitan University, Hino, Japan

***Department of Information Engineering, Oita National College of Technology, Oita, Japan

****Graduate School of Science and Technology, Shizuoka University, Hamamatsu, Japan

***Abstract***- Currently, academic papers and their authors can be mainly evaluated by the statistics in Bibliometrics such as number of citations, h-index, and impact factor. However, it usually takes at least half a year or more for Bibliometrics-based approaches to evaluate academic papers. Open access journals, which do not restrict browsers, are spreading especially in recent years. Further, the number of viewing academic papers published in open access journals and the number of posting articles about the papers to social media continue to increase year after year. Such data can be treated as time-series data with immediacy. Therefore it is thought that if the academic papers as time-series data can be analyzed by proper data mining techniques such as clustering, it will be possible to extract the characteristics of highly-cited scientific papers. Instead of conventional evaluation of scholarly papers based on Bibliometrics, this paper discusses a method for estimating scientific papers with the potential of being highly cited in future based on the associated time-series data.

***Keywords:*** open access journal; time-series data; Dynamic Time Warping; clustering; BIRCH

## 1 INTRODUCTION

Recently, open access journals (OAJ), which do not restrict viewing, are spreading in the world, especially in US and the ratio of such journal papers over all academic papers is increasing accordingly [1]. In general, since OAJ publish accepted papers worldwide in one week or so, the academic papers can be accessed and viewed without restrictions by any user. Compared to traditional journals, OAJ can ensure the immediacy of the scientific papers by proving shorter periods from submission to publication. On the other hand, traditional surveys about academic papers are based on so-called Bibliometric indices such as the total number of papers published by one author and the number of citations per paper. And such traditional surveys based on Bibliometrics tend to take long time just like submitted papers take long time to be published.

Very recently, as methods to evaluate the scientific papers in the immediate term, alternative Bibliometrics called Altmetrics, which use the posts to social media, such as Twitter (micro blogging), Facebook (blogging), and Mendeley (social bookmarking), and analyze the contents and numbers, are gathering attention. Typical services for evaluation of scientific papers by using Altmetrics include altmetric.com and ImpactStory. In Japan, Ceek.jp Altmetrics, a university-originated venture has begun to provide Altmetrics-based services [2]. In this paper, we mean by Altmetrics both methods for quantitative measurement of impacts of research products such as journal papers and data sets using the social media responses and activities as to measuring the future influences of emergent researches based on the results [3].

So we have thought that it is necessary to clarify the relationships between Altmetric indices such as traffic data (i.e., number of views) and social media posts and Bibliometric indices such as citation data. In the preliminary experiment, we have found weak or very weak correlations between the number of views or downloads as of the first month after publication and that of citations of the scientific papers. This detail will be explained in Section 3 of this paper. Since these correlations are negligible, we have clustered academic papers published in OAJ with immediacy by paying attention to the counts of views and downloads so as to predict Bibliometric indices such as the counts of citations. We have used time-series data consisting of the numbers of views and downloads of papers per month for three or six months in clustering. Our final objective, which we have not fully obtained yet, is to estimate papers with the possibility of being highly cited in the future by performing machine learning, that is, learning classifiers for such papers based on clustering results. If the objective is fully attained, it will be possible to know the technological trends at the very early stage. Figure 1 illustrates the big picture of our research as the flow of the associated processes. In this paper, we focus on the clustering of the academic papers based on the time- series data. Section 2 de-

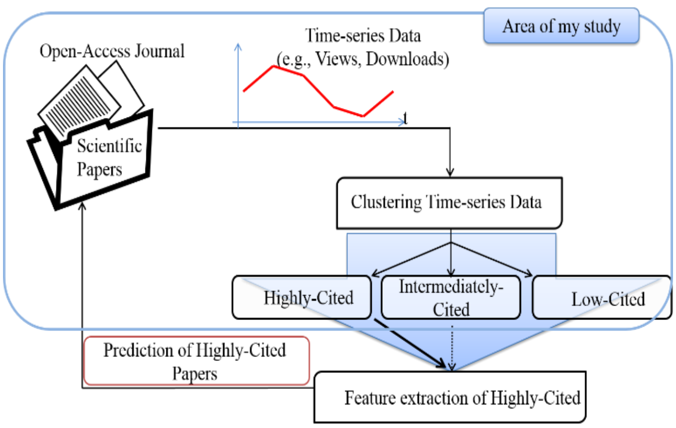



Figure 1: Big Picture of Our Research.

Table 1: Correlation between Numbers of Citations and Numbers of Views as of 1st Month.

| | Cited | Views |
|---|---|---|
| Cited | 1 | --- |
| Views | 0.0695 | 1 |

Table 2: Correlation between Numbers of Citations and Numbers of Downloads as of 1st Month.

| | Cited | Downloads |
|---|---|---|
| Cited | 1 | --- |
| Downloads | 0.2852 | 1 |

scribes the relevant works. Section 3 explains the normalization of the time series data. Section 4 and section 5 describe our clustering method and experiments using the method, respectively. Section 6 summarizes the contribution of our work and describes future challenges.

## 2 RELATED WORK

Most of works that cluster time-series data focus on the frequency. As clustering time-series data based on the frequencies as features, subsequence time-series clustering [4] is often used. So as to exactly handle the frequency of time-series data as the features, time-series data longer than a certain length are required. However, as our work focuses on the immediacy of the scientific papers published in OAJ, we use only data recorded every month for at most 12 months from publication, that is, 12 pieces of time-series data. Therefore, we use the shape of a wave as features of the time-series data. Then we can calculate dissimilarity (i.e., distance) of the time-series data using Dynamic Time Warping DTW [5].

Recently, studies on Altmetrics are becoming very active and are expected to complement Bibliometric evaluation of scientific papers. Posts to Twitter (i.e., Tweets) referring to academic papers published in OAJ are observed for several days just after publication, according to Gunther [6], thereby enabling the prediction of the number of citations. However, the number of collected Tweets mentioning scholarly papers is not so large, partially because it is rather difficult to exhaustively find correspondences between academic papers and tweets. Nakahashi et al [7] have done automatic mapping between academic papers and Tweets and have attained better performance than the previous works. However, their research have used only tweets referring to papers presented in the traditional academic meetings. There still remain a lot of works to do so as to automatically find correct correspondences between Tweets and scholarly papers on the web, such as OAJ.

## 3 NORMALIZATION OF TIME-SERIES DATA

We used the data of academic papers published in Public Library Of Science (PLOS) [8], one of US-based OAJ. We used the API provided by the PLOS and obtained the data of academic papers for two times, on 7/20/2013 and 12/22/2013. While the numbers of views are equal to those of accesses to web pages provided to individual academic papers by PLOS

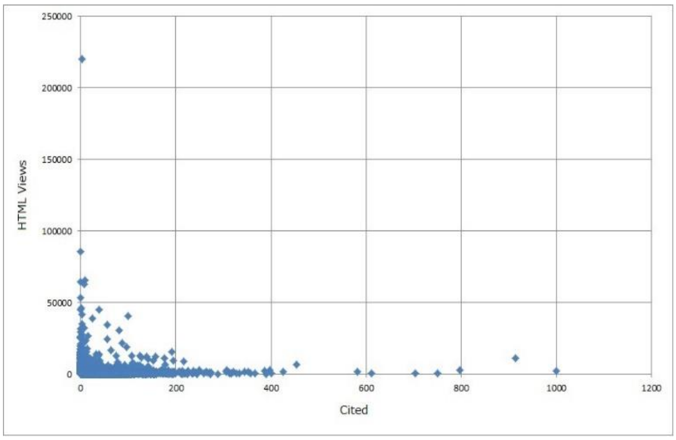



Figure 2: Number of Citations vs. Number of Views.

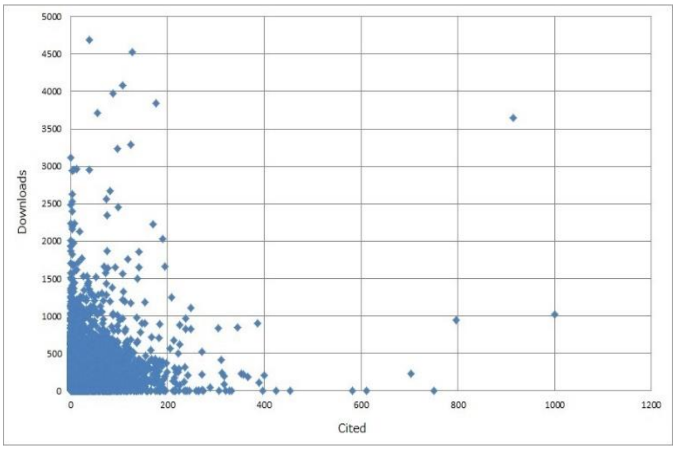



Figure 3: Number of Citations vs. Number of Downloads.

(html views), the numbers of downloads are equal to those of the saved PDF (pdf views). The numbers of views and downloads are monthly calculated from the published month. They constitute time-series data. The numbers of citations as to some papers are also calculated at fixed intervals from the published month. The numbers of citations as to others, however, are those of citations accumulated from the published date to the collected date. Then we used the accumulated numbers of citations in a uniform fashion.

We acquired academic papers on 7/20/2013 whose number n = 52,386 and calculated the correlation coefficient r between the numbers of citations and those of views as of the first month of papers since publication and we obtained a very weak correlation ($r = 0.0695$) (See Table I). As to the same set of papers, we also calculated a correlation coefficient $r$ between the numbers of citations and those of downloads as of the first month and obtained a weak correlation ($r = 0.2852$) (See Table II). The corresponding scatter charts are illustrated in Fig. 2 and Fig. 3. Then we judged these correlations negligible and decided to use time-series data of views and downloads instead. We normalized these time-series data by dividing all the values by those as of the first month. Figure 4 shows examples of normalized time-series data as to the numbers of downloads of 14 samples published in 2/2004 for the first 12 months.

# 4 CLUSTERING

We examined two different clustering techniques for time-series data. First, we made vectors out of the time-series data and conducted k-means clustering based on the cosine measure as to the vectors. Next, we calculated dissimilarities between two pieces of the time-series data by Dynamic Time Warping (DTW) and conducted clustering based on the dynamic time warping squashed trees, an extension of CF tree.

## 4.1 K-Means Clustering

### 4.1.1 Vectorizing Time-Series Data

First, let $a_i$ be the number of views or downloads (exactly, the ratio over the value for the first month) recorded for $i$-th month as to the scientific papers A. As a whole, the time series data for the paper A is represented as A = $a_1, a_2, \ldots, a_I$. Next we vectorize the time-series data as follows. So as to focus on the rate of change of each component of the time series data along the time line, we consider a sub-vector for the change of two consecutive elements. The $x$-component of the sub-vector is constantly 1 because the value is always one month. Each $y$-component of the sub-vector is represented as ($a_{k+1}$-$a_k$). Therefore, vectorized time-series data for the paper A is represented as a set of sub-vectors by the following formulas:

$$V_A = \{\overrightarrow{v_{A(1)}}, \overrightarrow{v_{A(2)}}, \ldots, \overrightarrow{v_{A(k)}}, \ldots, \overrightarrow{v_{A(kI-1)}}\} \quad (1)$$

$$\overrightarrow{v_{A(k)}} = \{1, a_{k+1} - a_k\} \quad (2)$$

Figure 5 shows two examples of vectorized time series data.

### 4.1.2 K-Means Method

Here we consider the similarity between the academic papers A and B. The similarity between corresponding sub-vectors is calculated by using the cosine measure. Let the angle between them be $\theta_k$. Then the cosine similarity between two vectors $\overrightarrow{v_{A(k)}}$ and $\overrightarrow{v_{B(k)}}$ is represented by the formula (3) (See Fig. 5). Note that as $0 \le \cos\theta_k \le 1$, two vectors is similar if the cosine similarity is close to 1 or dissimilar if close to 0. Further, by using the total sum $\cos_{sum}$ and the total product $\cos_{prod}$ of the cosine similarity between corresponding sub-vectors, which are defined in the formulas (4) and (5), respectively, the similarity of the academic papers A and B (i.e., the vector sets $\boldsymbol{V_A}$ and $\boldsymbol{V_B}$ ) $s$ is expressed as $s = (\cos_{\mathrm{sum}}, \cos_{\mathrm{prod}})$. Suppose that $I$ and $J$ are the lengths of the time-series data A and B, respectively. Every piece of time series data has the same length, that is, $I = J$.

$$\cos\theta_k = \frac{\overrightarrow{v_{A(k)}} \cdot \overrightarrow{v_{B(k)}}}{|\overrightarrow{v_{A(k)}}| \cdot |\overrightarrow{v_{B(k)}}|} \quad (3)$$

$$\cos_{sum} = \sum_{k=1}^{I-1} \cos\theta_k \quad (4)$$
$$(0 \le \cos_{sum} \le I - 1)$$

$$\cos_{prod} = \prod_{k=1}^{I-1} \cos\theta_k \quad (5)$$
$$(0 \le \cos_{prod} \le 1)$$

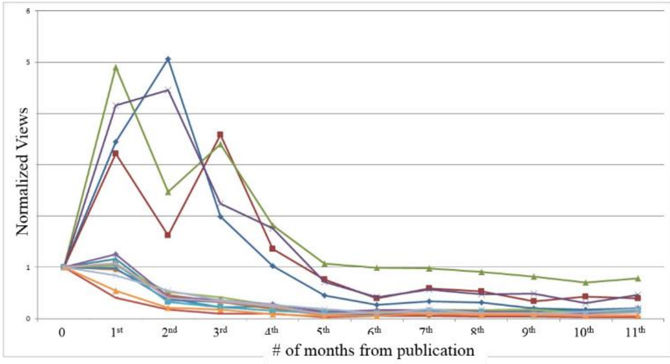



Figure 4: Examples of Normalized Time-Series Data.

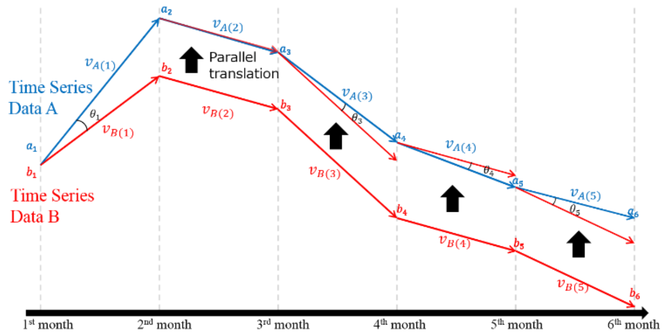



Figure 5: Vectorization of Time-Series Data.

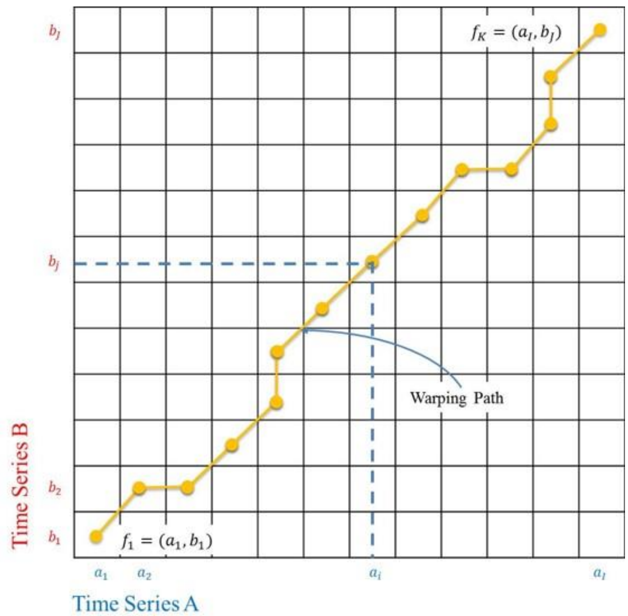



Figure 6: Example of Warping Path in DTW.

Here we cluster academic papers by using k-means clustering based on this similarity measure $s$. Beforehand, we sorted data for the whole academic papers according to the descending order of DOI (Digital Object Identifier) and divided into k groups as initial clusters. The centroid of the cluster is calculated as the vector set $\boldsymbol{Vm}$ expressed by the formula (6). Let

$N$ be the total number of the papers. The initial centroids are calculated by using the initial clusters.

$$V_m = \frac{\sum_{n=1}^{N} V_n}{N} = \{\frac{\sum_{n=1}^{N} \overrightarrow{v_{n(1)}}}{N}, \ldots, \frac{\sum_{n=1}^{N} \overrightarrow{v_{n(I-1)}}}{N}\} \quad (6)$$

In one repetition, the similarities between each paper and all the centroids are calculated. If there is another cluster to which the paper is more similar than the current cluster, the paper is moved to the former cluster. In case of any movement at the end of the repetition, the centroids of the clusters are updated. This process is repeated until the number of academic papers to move is less than a certain threshold, that is, stable.

## 4.2 Dynamic Time Warping Squashed Tree Clustering

### 4.2.1 Dynamic Time Warping

The Dynamic Time Warping (DTW) algorithm can calculate the dissimilarity between a pair of time-series data. The algorithm can map multiple points in one piece of time- series data to a single point in another piece of time series data, thereby allowing non-linear transformation as to the time axis.

Let us consider a pair of 12-month time-series data for two academic papers A and B. The time-series data are represented as $A = (a_1, a_2, \ldots, a_{11}, a_{12})$ and $B = (b_1, b_2, \ldots, b_{11}, b_{12})$. Then we construct a 12×12 matrix (i.e., table) by corresponding A and B to the row and column, respectively. In this table, grid points $f_k = (a_{ik}, b_{jk})$ represent correspondences between A and B. The series $F = (f_1, f_2, \ldots, f_k, \ldots, f_K)$ is called warping path. An example is shown in Fig. 6. The distance between $a_{ik}$ and $b_{jk}$ is denoted by $\delta(f_k)$ and is calculated by the formula (7). Using this distance, the evaluation function $\Delta(F)$ for the warping path $F$ is calculated by the formula (8). Here $w_k$ is a positive weight to $f_k$ and is calculated by the formula (9).

$$\delta(f_k) = |a_{ik} - b_{jk}| \quad (7)$$

$$\Delta(F) = \frac{1}{I+J} \sum_{k=1}^{K} w_k \cdot \delta(f_k) \quad (8)$$

$$w_k = (i_k - i_{k-1}) + (j_k - j_{k-1}) \quad (9)$$

$$i_0 = j_0 = 0$$

Here $i_k$ and $j_k$ which are integers between 1 and 12 denote subscripts of components $a$ and $b$ of a grid point $f_k$, respectively. $I$ and $J$ are the lengths of the time-series data A and B, respectively. The smaller $\Delta(F)$ is, the smaller the dissimilarity between $A$ and $B$ is. In other words, mapping is better in that case. The warping path $F$ with the minimum $\Delta(F)$ is the shortest warping path of $A$ and $B$. In that case $\Delta(F)$ is used as the dissimilarity between the two time series data. This is calculated by using the function mlpy.dtw_std provided by the machine learning library mlpy [9] in the programming language Python.

### 4.2.2 Dynamic Time Warping Squashed Trees

Dynamic time warping squashed tree (DTWS tree) [10] is a height-balanced binary tree, a variant of CF tree, when BIRCH, a typical hierarchical clustering method is adapted to time-series data. This method clusters time-series data based on DTW-dissimilarities by exhaustive searching, doing data compression.

First, let us consider the node vector $CF$ of the original CF tree. Generally using $N_0$ as the number of elements belonging to the node, the linear sum of vectors $\boldsymbol{LS}_0 = \Sigma_{k=1, N_0} \boldsymbol{X_k}$, and the squared sum $SS_0 = \Sigma_{k=1, N_0} (\boldsymbol{X_k})^2$, the node $CF$ of the CF tree is represented as $CF = (N_0, \boldsymbol{LS}_0, SS_0)$. On the other hand, the node vector $DTWS$ of the DTWS tree corresponds to the $CF$ vector but the squared sum $SS_0$ is omitted from the $CF$ and is represented as $DTWS = (N, \boldsymbol{ATW})$. Here $N$ is the number of time-series data belonging to the node $DTWS$ and $\boldsymbol{ATW}$ is the average vector of time-series data. $\boldsymbol{ATW}$ is calculated as follows: First the average of two time-series data $A$ and $B$ is calculated as $\boldsymbol{ATW}_\beta = (x_1, x_2, x_k, \ldots, x_K)$. Here $x_k$ correspond to $f_k$, calculated by the formula (10). As the average vector $\boldsymbol{ATW}_\beta$ is, in general, longer than the original time-series data A and B ($K \geq I = J$). Thus $\boldsymbol{ATW}_\beta$ is un-normalized with respect to the length. Then we compress it in order to obtain $\boldsymbol{ATW} = (y_1, y_2, \ldots, y_{k'}, \ldots, y_I)$ in accordance with the original time-series data. If the path for $x_k$ extends diagonally on the warping path,

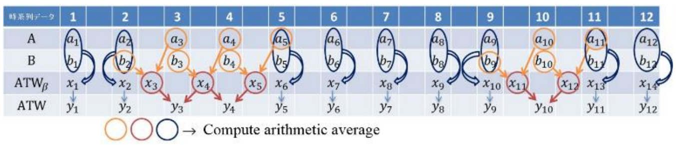



Figure 7: Computation of ATW.

Table 3: Data Used in the Experiments.

| Length of Time-series Data | Published | # of Papers | Data1 | | Data2 | |
|---|---|---|---|---|---|---|
| | | | Intermediately- and Higly-Cited | Highly-Cited | Intermediately- and Higly-Cited | Highly-Cited |
| 3 months | Before 2013/4 | 48261 | 10978 | 398 | 13890 | 579 |
| 6 months | Before 2013/1 | 43363 | 10978 | 398 | 13885 | 579 |
| 12months | Before 2012/7 | 33934 | 10978 | 398 | 13715 | 579 |

Table 4: Results of *K*-Means Clustering (*K*=10).

| # of Papers in a cluster | Data1 | | Data2 | | Difference | |
|---|---|---|---|---|---|---|
| | Cited≥10 | ratio(%) | Cited≥10 | ratio(%) | Cited≥10 | ratio(%) |
| 13,659 | 2,586 | 18.93 | 3,298 | 24.15 | 712 | 5.21 |
| 4,542 | 1,459 | 32.12 | 1,790 | 39.41 | 331 | 7.29 |
| 4,174 | 516 | 12.36 | 760 | 18.21 | 244 | 5.85 |
| 3,392 | 1,256 | 37.03 | 1,476 | 43.51 | 220 | 6.49 |
| 1,765 | 809 | 45.84 | 911 | 51.61 | 102 | 5.78 |
| 1,707 | 231 | 13.53 | 346 | 20.27 | 115 | 6.74 |
| 1,460 | 538 | 36.85 | 640 | 43.84 | 102 | 6.99 |
| 1,045 | 190 | 18.18 | 269 | 25.74 | 79 | 7.56 |
| 955 | 58 | 6.07 | 115 | 12.04 | 57 | 5.97 |
| 801 | 341 | 42.57 | 397 | 49.56 | 56 | 6.99 |
| 780 | 54 | 6.92 | 94 | 12.05 | 40 | 5.13 |
| 762 | 234 | 30.71 | 296 | 38.85 | 62 | 8.14 |
| 751 | 65 | 8.66 | 113 | 15.05 | 48 | 6.39 |
| 721 | 124 | 17.20 | 182 | 25.24 | 58 | 8.04 |
| 545 | 195 | 35.78 | 219 | 40.18 | 24 | 4.40 |
| 524 | 143 | 27.29 | 211 | 40.27 | 68 | 12.98 |
| 510 | 106 | 20.78 | 151 | 29.61 | 45 | 8.82 |
| 491 | 39 | 7.94 | 65 | 13.24 | 26 | 5.30 |
| 485 | 17 | 3.51 | 36 | 7.42 | 19 | 3.92 |
| 469 | 198 | 42.22 | 220 | 46.91 | 22 | 4.69 |
| 436 | 17 | 3.90 | 39 | 8.94 | 22 | 5.05 |
| 432 | 122 | 28.24 | 155 | 35.88 | 33 | 7.64 |
| 429 | 177 | 41.26 | 196 | 45.69 | 19 | 4.43 |

Table 5: Results of *K*-Means Clustering (*K*=25).

| | Data1 | | Data2 | | Difference | |
|---|---|---|---|---|---|---|
| # of Papers in a cluster | Cited>=10 | ratio(%) | Cited>=10 | ratio(%) | Cited>=10 | ratio(%) |
| 4,290 | 647 | 15.08 | 848 | 19.77 | 201 | 0.05 |
| 3,997 | 504 | 12.61 | 697 | 17.44 | 193 | 0.05 |
| 3,761 | 1,108 | 29.46 | 1,365 | 36.29 | 257 | 0.07 |
| 2,126 | 384 | 18.06 | 538 | 25.31 | 154 | 0.07 |
| 1,732 | 635 | 36.66 | 759 | 43.82 | 124 | 0.07 |
| 1,575 | 620 | 39.37 | 718 | 45.59 | 98 | 0.06 |
| 1,359 | 371 | 27.3 | 440 | 32.38 | 69 | 0.05 |
| 1,066 | 73 | 6.85 | 123 | 11.54 | 50 | 0.05 |
| 906 | 411 | 45.36 | 473 | 52.21 | 62 | 0.07 |
| 854 | 183 | 21.43 | 252 | 29.51 | 69 | 0.08 |
| 789 | 84 | 10.65 | 131 | 16.6 | 47 | 0.06 |
| 721 | 303 | 42.02 | 344 | 47.71 | 41 | 0.06 |
| 633 | 173 | 27.33 | 212 | 33.49 | 39 | 0.06 |
| 553 | 253 | 45.75 | 293 | 52.98 | 40 | 0.07 |
| 542 | 60 | 11.07 | 99 | 18.27 | 39 | 0.07 |
| 458 | 166 | 36.24 | 189 | 41.27 | 23 | 0.05 |
| 433 | 73 | 16.86 | 104 | 24.02 | 31 | 0.07 |
| 402 | 91 | 22.64 | 125 | 31.09 | 34 | 0.08 |

Table 6: Kruskal-Wallis Test of the Results by *K*-Means Clustering (*K*=10).

| | $\chi^2$ | # of degree of freedom | p-value |
|---|---|---|---|
| Data1 | 3283.75 | 96 | $p_1<2.2\times10^{-16}$ |
| Data2 | 2962.84 | 96 | $p_2<2.2\times10^{-16}$ |

its length will be 1; If the path for $x_k$ extends either horizontally or vertically, its length will be 0.5. Based on this calculation, $\boldsymbol{ATW}_\beta$ is compressed to $\boldsymbol{ATW}$. The value of $y_{k'}$ is calculated, that is, equal to that of $x_k$ by the formula (11) if the length to $x_k$ from the starting point of the time-series data is integer, otherwise it is calculated as linear interpolation between the two consecutive elements by using the formula (12). As to the time warping path in Fig. 6, the processes of the above calculation are shown in Fig. 7.

$$x_k = \frac{a_{ik}+b_{jk}}{2} \tag{10}$$

$$y_{k'} = \begin{cases} x_k & (11) \\ \dfrac{x_k + x_{k+1}}{2} & (12) \end{cases}$$

The procedures of the DTWS tree are similar to those of the CF tree. The dissimilarity between the time-series vector to be added and the average time-series vector of the node is calculated based on DTW. If there exists a node with the minimum dissimilarity, less than a prescribed threshold, the new vector is added to the node.

# 5 EXPERIMENTS

## 5.1 Outline of the Experiments

For the experiments, we used data that were collected from the scientific open access journal PLOS twice on 7/20/2013 and 12/22/2013, respectively. The data for the scientific papers include the numbers of views and those of downloads of each paper as of each month as time-series data starting just after the publication and the number of citations of each paper as of the time of collection. We collected data as to 52,555 scientific papers on 7/20/2013, among which 52,386 papers have information as to citations. Further, from the collection, we selected scientific papers which have 3-month, 6-month, and 12-month numbers of views and of downloads as of 7/20/2013. Data collected on 7/20/2013 and 12/22/2013 are called Data1 and Data2, respectively (See Table III). We used two clustering methods described in Sections 4.1 and 4.2 for comparison. In one run of experiments, we clustered the same collection of academic paper data separately based on the numbers of views and of downloads. As a result, we obtain two sets of clusters. By making intersections of two sets of clusters, we obtained one set of clusters as a final result. We evaluated the final set of clusters in terms of the number of citations. The average number of citations is 11.81 and the median is 5 in Data1. The low-cited papers are defined as those with less than 10 citations. The highly-cited papers are assumed to belong to the top 10 % of the whole collection with respect to citations. In the top 10% subset of Data1, the minimum number of citations is 87. For the simplification of the judgment when evaluating clustered results, the threshold for the highly-cited papers is set to 90 in our experiments. Therefore, the intermediately-cited papers are defined as those with 10≤citations<90. Table III shows some statistics about our scientific paper collections prepared for the experiments. The "intermediately- and highly-cited" papers and the "highly-cited" papers denote the items cited times ≥10 and those cited times≥90", respectively.

Table 7: Results of DTWS Tree Clustering of Three-Month Time-Series Data.

| Threshold | | # of clusters | | Data1 | | Data2 | |
|---|---|---|---|---|---|---|---|
| $X_{html}$ | $X_{pdf}$ | All | 10 papers or more | Intermediately- and Higly-Cited Ratio>=90% | Num | Intermediately- and Higly-Cited Ratio>=90% | Num |
| 32 | 47 | 123 | 43 | 3 | 8,851 | 3 | 8,874 |
| 42 | 22 | 161 | 49 | 3 | 8,818 | 3 | 8,819 |
| 42 | 42 | 96 | 33 | 2 | 8,818 | 2 | 8,820 |
| 42 | 40 | 114 | 38 | 4 | 8,809 | 5 | 8,848 |
| 42 | 25 | 150 | 51 | 3 | 8,806 | 3 | 8,801 |
| 42 | 32 | 118 | 37 | 2 | 8,803 | 2 | 8,797 |
| 42 | 15 | 254 | 90 | 5 | 8,779 | 7 | 10,020 |
| 42 | 37 | 112 | 40 | 2 | 8,778 | 3 | 8,822 |
| 42 | 12 | 295 | 92 | 4 | 8,774 | 4 | 8,767 |
| 42 | 10 | 301 | 80 | 4 | 8,773 | 5 | 8,802 |
| 42 | 50 | 101 | 36 | 2 | 8,761 | 2 | 8,755 |
| 42 | 47 | 100 | 36 | 2 | 8,760 | 2 | 8,754 |
| 42 | 20 | 185 | 65 | 3 | 8,757 | 4 | 8,785 |
| 42 | 35 | 118 | 40 | 2 | 8,757 | 3 | 8,799 |
| 42 | 7 | 395 | 101 | 7 | 8,754 | 7 | 8,747 |
| 42 | 27 | 143 | 52 | 3 | 8,753 | 4 | 8,827 |
| 42 | 30 | 148 | 57 | 2 | 8,751 | 3 | 8,760 |
| 42 | 45 | 101 | 38 | 2 | 8,750 | 3 | 8,791 |
| 42 | 5 | 634 | 139 | 9 | 8,733 | 11 | 8,778 |
| 42 | 17 | 194 | 57 | 3 | 8,721 | 4 | 8,817 |

## 5.2 Results

### 5.2.1 *K*-Means Clustering

We conducted k-means clustering on 3-month time-series data with respect to both views and citations as k = 10 and 25. Because each result consists of 10 or 25 clusters, the academic

Table 8: Recall, Precision, and $F$-Value of Intermediately- and Highly-Cited Papers.

| Length of Time-series Data | Threshold $X$ | | # of Papers in a Cluster | Data | # of Intermediately- and Highly-Cited Papers in a Cluster | R (%) | P (%) | F (%) |
|---|---|---|---|---|---|---|---|---|
| | Views | Downloads | | | | | | |
| 3 | 42 | 42 | 8,470 | Data1 | 8,460 | 77.06 | 99.88 | 78.42 |
| 3 | 42 | 32 | 8,353 | Data1 | 8,347 | 76.03 | 99.93 | 77.43 |
| 3 | 45 | 42 | 9,447 | Data1 | 8,385 | 76.38 | 88.76 | 77.19 |
| 3 | 47 | 42 | 12,766 | Data1 | 8,539 | 77.78 | 66.89 | 76.84 |
| 3 | 22 | 42 | 9,486 | Data1 | 8,338 | 75.95 | 87.90 | 76.74 |
| 3 | 42 | 40 | 8,276 | Data1 | 8,265 | 75.29 | 99.87 | 76.71 |
| 3 | 42 | 30 | 8,250 | Data1 | 8,245 | 75.10 | 99.94 | 76.54 |
| 6 | 50 | 42 | 14,464 | Data1 | 8,565 | 78.02 | 59.22 | 76.19 |
| 3 | 45 | 32 | 9,524 | Data1 | 8,275 | 75.38 | 86.89 | 76.14 |
| 3 | 45 | 30 | 8,311 | Data1 | 8,175 | 74.47 | 98.36 | 75.86 |

Table 9: Number of Clusters in Top Clusters Ranked by $F$-Value for Intermediately- and Highly-Cited Papers.

| Length of Time-series Data | # in Top 100 clusters | # in Top 200 clusters | # in Top 300 clusters |
|---|---|---|---|
| 3 months | 70 | 119 | 201 |
| 6 months | 30 | 79 | 93 |
| 12 months | 0 | 2 | 6 |

papers are divided into 10 × 10 or 25 × 25 clusters. However, there also exist clusters which contain no elements. Then clusters with more than 400 elements are picked up and described in Table IV and Table V for k = 10 and k = 25, respectively. In the tables, "# of papers", "Cited≥10", and "ratio(%)" denote the number of elements in the cluster, the number of elements cited for 10 or more times, and the ratio over the cluster (%), respectively. By k-means clustering, no clusters with only intermediately- and highly-cited academic papers could be found. However, the Kruskal-Wallis test, a nonparametric method, was conducted on 100 clusters constructed by k-means clustering (i.e., k=10) using Kruskal.test function of the R language [11]. Table VI shows the results of the Kruskal-Wallis test. As $p_1$, $p_2 \leq$ significance level $\alpha = 0.01$ from Table VI, it has been confirmed that there exist significant differences between the median numbers of citations of clusters. Note that, because there exist three clusters with less than two elements in the results and the tests were performed on the rest and thus the number of degrees of freedom was 96.

### 5.2.2 DTWS Tree Clustering

DTWS tree clustering was performed on 3-month, 6-month, and 12-month time-series data with respect to the number of views and of citations. DTWS tree is a clustering method using exhaustive searching based on thresholds. Then, the size of clusters and the number of clusters change, depending on the given thresholds. For both the numbers of views and of downloads, the threshold $X$ takes one of 26 different values:{0.3, 0.5, 0.7, 1, 1.5, 2, 3, 5, 7, 10, 13, 15, 17, 20, 22, 25, 27, 30, 32, 35, 37, 40, 42, 45, 47, 50}. And time-series data have 3 different lengths. Then we get 2,028 clusters as a final result. From the result, we excluded clusters whose size were less than 10. Thus, we evaluated the result, focusing on clusters in the result whose size is larger than or equal to 10. Table VII shows the results of 3-month time-series data. In Table VII, "$X_{html}$" and "$X_{pdf}$" denote the thresholds for the numbers of views and of downloads, respectively. "*All*" and "10 papers or more" denote the total number of clusters and the number of clusters which contain 10 or more elements, respectively. Further, "intermediately- and highly-cited" and "*Num*" denote the number of and the total paper cardinality of clusters, respectively, where 90 or more percent of the elements are intermediately- and highly-cited papers.

Table VII describes only results containing mostly intermediately- and highly-cited academic papers. In DTWS tree, no clusters with only highly-cited academic papers could be found, either. However, using 3-month time-series data, with 32 as the *view*-threshold and 47 as the *download*-threshold, approximately 80% of the intermediately- and highly-cited papers in Data1 could be successfully extracted. Similarly, approximately 72% of the intermediately- and highly-cited papers in Data2 could be extracted with 42 as the *view*-threshold and 15 as the *download*-threshold. As for 12-month time-series data, approximately 93% of the intermediately- and highly-cited papers in Data1 could be extracted with 37 as the *view*-threshold and 50 as the *download*-threshold. Approximately 79% of the intermediately- and highly-cited papers in Data2 could be extracted with 50 as the *view*-threshold and 40 as the *download*-threshold.

Further, evaluation of the clustering results was done focusing on one of the clusters using the $F$-value calculated from

Table 10: Recall, Precision, and $F$-Value of Highly-Cited Papers.

| Length of Time-series Data | Threshold | | # of Papers in a Cluster | Data | # of Highly-Cited Papers in a Cluster | R (%) | P (%) | F (%) |
|---|---|---|---|---|---|---|---|---|
| | Views | Downloads | | | | | | |
| 12 | 47 | 42 | 506 | Data1 | 396 | 99.50 | 78.26 | 97.50 |
| 12 | 47 | 50 | 516 | Data1 | 396 | 99.50 | 76.74 | 97.32 |
| 12 | 47 | 40 | 496 | Data1 | 393 | 98.74 | 79.23 | 96.94 |
| 12 | 47 | 30 | 494 | Data1 | 392 | 98.49 | 79.35 | 96.73 |
| 3 | 25 | 35 | 500 | Data1 | 389 | 97.74 | 77.80 | 95.88 |
| 12 | 47 | 22 | 459 | Data1 | 379 | 95.23 | 82.57 | 94.14 |
| 12 | 47 | 12 | 466 | Data1 | 375 | 94.22 | 80.47 | 93.02 |
| 6 | 25 | 40 | 473 | Data1 | 375 | 94.22 | 79.28 | 92.90 |
| 6 | 25 | 32 | 473 | Data1 | 375 | 94.22 | 79.28 | 92.90 |
| 6 | 25 | 47 | 479 | Data1 | 375 | 94.22 | 78.29 | 92.80 |

Table 11: Number of Clusters in Top Clusters Ranked by $F$-Value for Highly-Cited Papers.

| Length of Time-series Data | # in Top 100 clusters | # in Top 200 clusters | # in Top 300 clusters |
|---|---|---|---|
| 3 months | 48 | 87 | 110 |
| 6 months | 24 | 71 | 122 |
| 12 months | 28 | 42 | 68 |

the precision ratio $P$ and recall ratio $R$ of the results. $R$, $P$, and $F$ are calculated by the formulas (13), (14), and (15), respectively. The precision ratio is expected to increase as the numbers of citations increase in a course of time. The weight $\beta$ for the recall ratio is set to $\beta$ = 3.5 in the formula (15).

$$R(\%) = \frac{relevant\ papers\ in\ the\ cluster}{relevant\ papers} \times 100 \quad (13)$$

$$P(\%) = \frac{relevant\ papers\ in\ the\ cluster}{papers\ in\ the\ cluster} \times 100 \quad (14)$$

$$F(\%) = \frac{(\beta^2+1) \times P \times R}{\beta^2 \times P + R} \quad (15)$$

All the resultant clusters are sorted in the descending order of $F$-values with respect to the intermediately- and highly-cited scientific papers. Among them, the top 10 clusters are shown with precision ratios, recall ratios, and $F$-values in Table VIII. By inspecting Table VIII, it is known that there exist clusters with recall ratios over 75% and precision ratios over 90% for the intermediately- and highly-cited scientific papers. It is also known that 9 out of 10 clusters are based on 3-month time series data. The numbers as to the top 100 clusters, top 200 clusters, and top 300 clusters sorted by the $F$-values for the 3-month-, 6-month-, and 12-month time-series data are shown in Table IX. From Table IX, it is known that clusters from 3-month time-series data are relatively highly ranked with respect to $F$-values. Similarly, the top 10 clusters for the highly-cited papers sorted by $F$-values are shown in the descending order in Table X. From Table X, it is known that there exist clusters with recall ratios over 90% and precision ratios over 75%. The numbers as to the top 100 clusters, top 200 clusters, and top 300 clusters sorted by the $F$-values with respect to highly-cited papers for the 3-month-, 6-month-, and 12-month time-series data are shown in Table XI. From Table XI, it is known that clusters based on 3-month- and 6-month time-series data are relatively highly ranked.

### 5.2.3 Discussion

By k-means clustering, only highly-cited academic papers could not be extracted. In other words, at least under the k-means clustering, vectors made from time-series data of numbers of views and of downloads cannot effectively represent the characteristics of only highly-cited academic papers. However, by the results of the Kruskal-Wallis test, it is confirmed that there exist significant differences among the medians of numbers of citations as to clusters.

By merging multiple clusters with 90% or more precision ratios as to intermediately- and highly-cited papers based on DTWS tree clustering, 80% or more of the papers could be recalled. Further, it was found that there existed clusters containing intermediately- and highly-cited papers with the recall ratios of 75% or more and the precision ratios of 90% or more. It was also found that there existed clusters containing highly-cited papers with the recall ratios of 90% or more and the precision ratios of 75% or more. This indicates that vectors made from time-series data consisting of numbers of views and downloads can predict intermediately- and highly-cited papers under the DTWS tree clustering. The numbers of citations in the Data2 represent those of the scientific papers published 8 or more months ago. Therefore, by clustering 3-month time-series data by DTWS tree clustering, at best approximately 77% of the intermediately- and highly-cited papers published 8 or more months ago could be detected. Also, at best approximately 98% of the highly-cited papers academic papers could be detected. Further, clusters containing rather many low-cited scientific papers could be found. In summary, by clustering the 3-month time-series data by using DTWS tree clustering, each of low-, intermediately- and highly-, and highly-cited academic papers could be detected with high likelihood.

In this experiment, the number of days from publication was not sufficiently considered. Thus, the numbers of citations of papers with different publication dates were equally treated. It is expected that a clustering scheme considering exactly the number of days from publication can cluster the academic papers with higher accuracy.

## 6 CONCLUSION

We have clustered academic papers published in open access journals by using time series data of the numbers of views and downloads. We used k-means clustering and clustering DTWS tree clustering and compared the performances. As a result, clusters mostly containing intermediately- and highly-cited papers could be discovered. Indeed, the result described in this paper cannot directly confirm that highly-cited papers in the first year in open access journals can be predicted by at least the first three-month time series data of views and downloads based on the extracted features of these clusters. However, the obtained observation at least suggests that creation of classifiers based on the time-series data of downloading and browsing is promising. The discovery that clustering based on the *time-series data* can isolate highly-cited papers is significant for the relevant researchers and practitioners. Therefore, the following approaches to classification are among our future plan. To be more concrete, one possible approach is to directly use clusters created in the way proposed in this paper as a kind of classifier and classify papers into "will-be"-highly-cited papers if they are most similar to the clusters mostly consisting of highly-cited papers with respect to the first three-month time-series data. Another approach is to create a classifier based on the time-series data by using clusters (i.e., highly-cited, intermediately-cited, or low-cited) obtained by the proposed way as the training set. Of course, the latter approach will draw heavily upon further researches about how to represent features of time-series data and which methods to choose as a classifier.

DTWS tree clustering experiments in 3 different lengths of time-series data were compared. However, in k-means clustering, k was uniquely fixed and only 3-month time-series data were used. For this reason, there remain possibilities that the results were not properly compared among the two clustering methods. It is thought that other advanced methods such as x-means can remedy some of the above problems and improve the comparison results.

We used the numbers of views and downloads as of every month. This is because PLOS releases time-series data every month. However, if finer-grained time-series data, for example, of every day, are available, more accurate clustering of scholarly papers may be possible. Further, a wide range of academic papers have been posted on PLOS. Therefore, by clustering papers in restricted areas, more specialized characteristics may be detected.

Further, as using 12-month time-series data for clustering lacks the immediacy of estimated papers, clustering based on data provided by Altmetrics such as posts in social media and links in social bookmarks is expected to overcome the problem. From our experimental results, the clustering results are expected to provide the features required for machine learning, that is, classification of highly-cited papers.

## ACKNOWLEDGEMENTS

This research was supported by Grant-in-Aid for Research on Priority Areas, Tokyo Metropolitan University, "Research on social big data."

## REFERENCES

[1] M. Laakso, et al., "The Development of Open Access Journal Publishing from 1993 to 2009," PLoS ONE 6(6): e20961. doi:10.1371/journal.pone.0020961 (2011).
[2] Ceek.jp Altmetrics http://altmetrics.ceek.jp/ Accessed (2013).
[3] J. Priem, H. A. Piwowar, and B. M. Hemminger, "Altmetrics in the wild: Using social media to explore scholarly impact," arXiv preprint arXiv:1203.4745 (2012).
[4] T. Ide,"Why does Subsequence Time-Series Clustering Produce Sine Waves?" Knowledge Discovery in Databases: PKDD 2006, Lecture Notes in Computer Science, Vol. 4213, pp. 211-222 (2006).
[5] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," Readings in speech recognition, A. Waibel and K. -F. Lee (Eds.). Morgan Kaufmann Publishers Inc., pp.159-165 (1990).
[6] G. Eysenbach, "Can Tweets Predict Citations? Metrics of Social Impact Based on Twitter and Correlation with Traditional Metrics of Scientific Impact," Journal of Medical Internet Research, Vol.13, No.4-e123 (2011).
[7] H. Nakahashi, et al.,"Automatic Alignment of Tweet," IEICE Technical reports, Vol.113, No.105, .pp.65-70 (2013) (in Japanese).
[8] PLOS http://www.plos.org/ Accessed (2014).
[9] Mlpy http://mlpy.sourceforge.net/ Accessed (2014).
[10] K. Nakamoto, et al., "Fast Clustering for Time-series Data with Average-time-sequence-vector generation Based on Dynamic Warping," Trans. JSAI, Technical papers, vol.18, No.3, pp.144-152 (2003) (in Japanese).
[11] R http://www.r-project.org/ Accessed (2014).

(Received September,28,2015)
(Revised January 21,2016)

**Hiroshi Ishikawa** received the B.S. and Ph.D degrees in Information Science from the University of Tokyo. After working for Fujitsu Laboratories and being a full professor of Shizuoka University, he is now a full professor of Tokyo Metropolitan University from April, 2013. His research interests include database, data mining, and social big data. He has published actively in international, refereed journals and conferences, such as ACM TODS, IEEE TKDE, VLDB, IEEE ICDE, and ACM SIGSPATIAL. He has authored some books, *Social Big Data Mining* (CRC Press). He is fellows of IPSJ and IEICE (The Institute of Electronics, Information and Communication Engineers) and members of ACM and IEEE.

**Masaki Endo** received his B.E. degree from Polytechnic University, Tokyo and graduated from the course of Electrical Engineering & Computer Science, Graduate School of Engineering Polytechnic University and received M.E degree from NIAD-UE, Tokyo. He is currently an assistant professor of Polytechnic University, Tokyo. His research interests include Web service and Web mining. He is also members of DBSJ, IPSJ and IEICE.

**Iori Sugiyama** received his B.E. degree from Tokyo Metropolitan University, Japan in 2014. He entered Tokyo Institute of Technology, Japan in 2014 and now he is a Master course student. His research interests include educational technologies and educational data mining. He is a member of Japan Society of Educational Technology (JSET).

**Masaharu Hirota** received his Doctor of Informatics in 2014 from Shizuoka University. Since April 2015, he has been working as assistant professor in National Institute of Technology, Oita College, Department of Information Engineering. His main research interests include geo social data, multimedia, and visualization. He is a member of ACM, IPSJ and IEICE.

**Shohei Yokoyama** is a lecturer of Informatics at Shizuoka University, Japan, and he was previously with National Institute of Advanced Industrial Science and Technology (AIST) for two years. His research concerns data engineering, and its focus is on georeferenced user-generated contents on Social Networking Service

# Distributed Remote Input/Output Control Method in Real Time Processing for CNC


Akihiro Yamashita*, Hiroshi Mineno* , and Tadanori Mizuno**

*Graduate School of Science and Technology, Shizuoka University, Japan
**Faculty of Information Science, Aichi Institute of Technology, Japan
Yamashita.Akihiro@ma.mee.co.jp, mineno@inf.shizuoka.ac.jp, mizuno@mizulab.net



***Abstract*** - Systems for manufacturing using IoT (Internet of Things) solutions attract attention recently. In the IoT solutions, all the relevant elements are connected to internet and all the information are gathered from processes from manufacturing to logistics for data analysis, aiming at the construction of flexible manufacturing systems.

In relation to the network connection and the distributed input/output control method, which are key technologies for IoT, we report in this paper on a pioneering basic technology, which is the input/output control method in real time processing for CNC. We conceptualized and adopted in our products in the 1990s. The basic idea is to define the communication method which is high reliable and suitable for the real time control for the CNC and to achieve modularization and distribution of the components of the CNC. The cumulative number of products shipped with this I/O control method incorporated exceeded 5,000,000.



***Keywords***: network, Internet of Things, distributed control, CNC, flexible manufacturing systems


## 1 INTRODUCTION

Solutions using IoT (Internet of Things) technologies attract attention recently. In the IoT solutions, all the relevant elements are connected to the internet, and all the information of the processes from manufacturing to logistics are gathered, aiming at the construction of flexible manufacturing systems based on the analysis of such big data[1]. On the production sites, such IoT solutions are also expected to be incorporated into the manufacturing process. Efforts are made to incorporate all the relevant information into the manufacturing process to facilitate grasping of the current status or each step of the manufacturing process until completion[2]-[4]. The relevant information covers the following: order information provided by the sales department and incorporated into the production system, parts orders made from the purchasing system, manufacturing/production instructions to the production sites, control information for machining or assembly, and in-process product identification information based on sensor technologies. In the field of parts machining, NC machine tools play a central role in IoT, and the use of them makes it possible to establish a link between the parts machining information based on the manufacturing data (including part programs) and the part information obtained from the input/output (I/O) control.

In relation to the network connection and the distributed input/output control method, which are key technologies for IoT, we report in this paper on a pioneering basic technology, which is the input/output control method in real time processing for CNC we conceptualized and adopted in our products in the 1990s. The basic idea for this I/O control method is to define the communication method suitable for the real time control for the CNC[5]. With this I/O control method, a compact, low-cost and safety network can be achieved using serial communication. This method enables to modularize the components of the CNC, connect them through a network, and achieve distribution of the components to build a flexible system suitable for the purpose[6]. The cumulative number of communication modules shipped with this control method incorporated exceeded 5,000,000. It can be said that this control method contributed to the development of the precursor of today's CNC.

## 2 RELATED TECHNOLOGIES

### 2.1 Configuration and Functions of the CNC

Figure 1 shows the configuration of the CNC. The CNC consists of the display and operation unit, NC unit, drive units, servo motors, and spindle motor. The display and operation unit is used to create part programs, operate the machine, and display the part programs, machining status, and machine status.

The NC unit is used to analyze part programs, output the machine movement distance to the servo drive units as a movement command, and control the machine movement. The NC unit is also used for sequence control for machining. The servo drive units are used to control the servo motors for the tool nose path control. The spindle drive units are used to control the spindle motor for rotating the tool to achieve the cutting work.

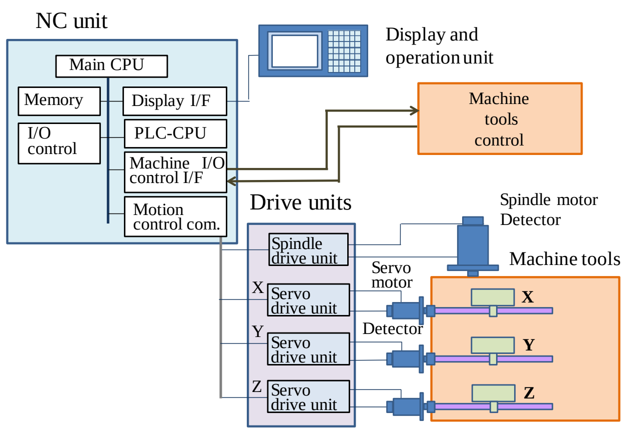



Figure 1: Configuration of the CNC.

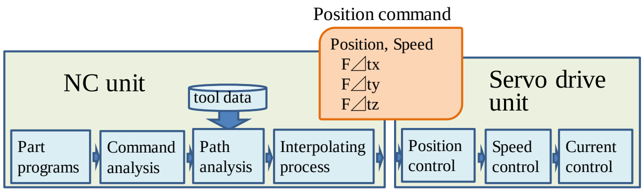



Figure 2: Position Control Processing in the CNC.

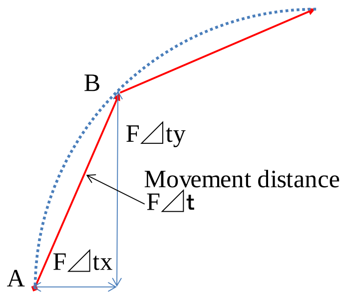



Figure 3: Interpolation Processing in the CNC.

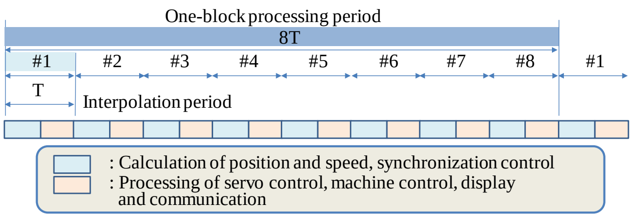



Figure 4: Real Time Control Processing in the CNC.

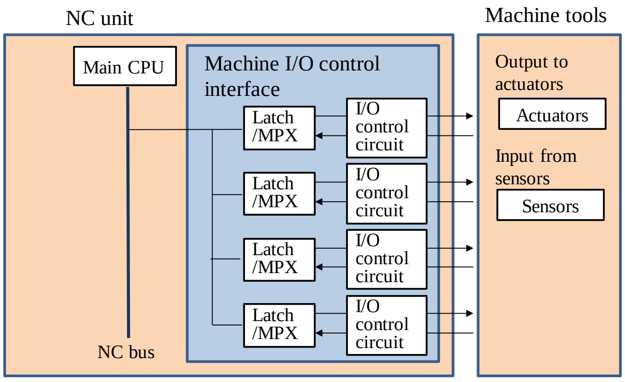



Figure 5: Schematic Block Diagram for Parallel I/O Control.

## 2.2 Tool Path Control in the CNC

The CNC executes part programs for driving the multi-axis machine tools (for example, X axis, Y axis, Z axis, etc. and the spindle) to move the tool nose position or move a part and cut the part

Figure 2 shows the details of the position control processing. The CNC analyzes the part programs, adds the tool information to the analysis result, and calculates the tool path data. Then, the NC unit calculates the movement distance per unit time for each control axis by an interpolating operation. Figure 3 shows the interpolation processing of the CNC, to control two axes (X axis and Y axis) simultaneously. In Fig. 3, the command for the movement from A to B is generated as follows: at first, the movement distance $F\Delta T$, which is the product of the speed F and the unit time $\Delta T$, is calculated, and then, the distance is decomposed into the x-axis component of the movement distance $F\Delta t_x$ and the y-axis component of the movement distance $F\Delta t_y$.

The position commands calculated in the NC unit are sent to the servo drive units. The servo drive units control the position/speed and the current for driving servo motors based on the position/speed command values sent by the NC unit and the feedback information sent by the detector attached to each servo motor.

## 2.3 Sequence Control in the CNC

In the sequence control (machine I/O control function), input signals are used to monitor the machine status or to obtain the sensor information, and output signals are used to control the actuators.

The CNC controls the machine tools operations by calculating the movement command values at a fixed time interval and sending them to the drive units, implementing the sequence control of the machining procedure, and indicating status information on the display. In the sequence control, the CNC executes sequence programs. In response to the sensor information input signals sent in synchronization with the control cycle in the CNC, the machine control signals are output. It is essential to ensure real time sequence control processing with an accuracy of 0.1 msec because the processing is synchronized with the positioning control cycle in the CNC.

Figure 4 shows the time division for real time control processing in the CNC. For one axis, CNC calculates a command value and speed, and performs miscellaneous synchronization control in the first half of the interpolation period T. In the second half of the interpolation period T, the system performs servo control, machine sequence control, and data processing for display and communication. For controlling eight axes (at the maximum), the interpolation periods for eight axes are regarded as one cycle for the control operation.

## 2.4 Conventional Parallel I/O Control Method and Issues

Figure 5 shows the schematic block diagram of the machine I/O control interface in the NC unit. The conventional machine I/O signal control is performed using a parallel I/O function of the NC unit main CPU. In this method, the machine status is checked using the parallel input signal from the NC unit main CPU (ON/OFF control, +24 V/0 V), and the machine operations are controlled using the parallel output signal (ON/OFF control, +24 V/0 V). For this type of I/O control, several tens to one hundred signal wires are connected inside the power panel and externally for the sensor information input and the actuator control.

In the parallel I/O control method, each signal requires a wire connection between the NC unit and the actuator. Thus, the method involves issues such as the degradation of workability caused by the increase in the number of signals and wire bundles, and the lowering of reliability due to noises, etc. caused by the longer wire length.

Also, in the parallel I/O control method, data is directly transmitted between the NC unit and the machines or actuators. Therefore, the NC unit has to have numerous circuits to support the I/O control points according to the machine specifications. The lineup of the NC unit ranges from the high-end model to the general market model depending on the number of control axes and the control performance. In other words, the maximum number of I/O control points is determined for each class of the model. Therefore, there are issues with the NC unit cabinet volume for using the high-end model according to the drive control specifications. In such cases, it is inevitable to choose a NC unit with a large cabinet which supports a lot of I/O control points even when the machine requires less I/O control points. More specifically, the NC unit cabinet volume cannot be selected flexibly without any restraints because the fact that the NC unit has to contain the I/O control circuits prevents downsizing of the NC unit cabinet.

## 2.5 Conventional I/O Control Method for Serial Communication and Issues

Figure 6 shows an example of the schematic block diagram of machine I/O control for a regular serial communication. The machine I/O control interface in the NC unit consists of the I/O buffer and the communication control part for serial communication. The distributed remote I/O unit consists of the communication control part and the I/O control circuit. By transmitting the I/O signals for controlling the machine tools through serial communication, it is possible to mitigate the issues such as the degradation of workability caused by the increase in the number of wire bundles for signals, and the lowering of reliability due to noises caused by the longer wire length.

In the 1990s, CPUs were in most cases mounted on both the input and output devices to establish serial communication. The master communication control part of the NC unit consists of CPU and memory dedicated to communication and communication control circuit. Likewise, the slave communication control part of the distributed remote I/O unit requires CPU and memory dedicated to communication, communication control circuit, and I/O control circuit. Therefore, there are issues that a large circuit area is required, and the method incurs higher costs as compared to the parallel I/O control method. In addition, the development of software for CPU dedicated to communication is required both in the NC unit and the distributed remote I/O unit.

## 2.6 Serial Communication Processing by the Main CPU and Issues

To reduce the hardware cost for serial communication, it may be possible for the main CPU of the NC unit to perform serial communication processing instead of the CPU in the communication control part.

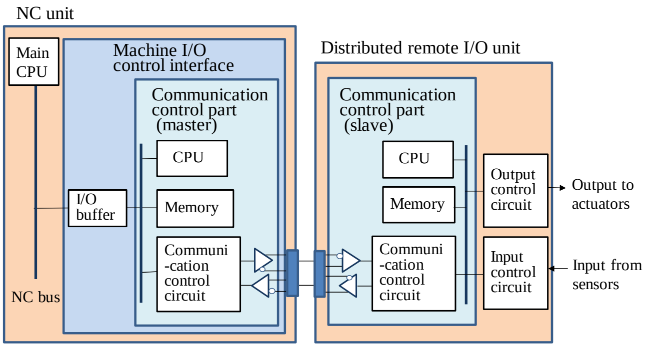



Figure 6: Schematic Block Diagram for Regular Serial Communication.

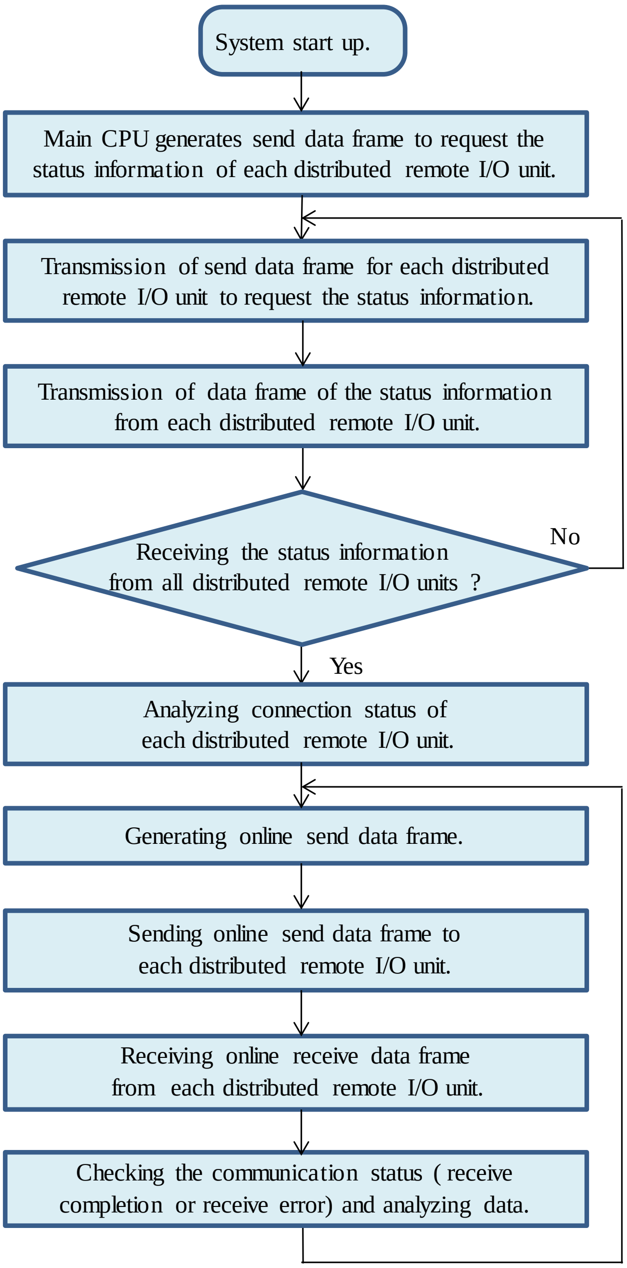



Figure 7: Processing Flow for Conventional Serial Communication.

Figure 7 shows the serial communication processing flow of the main CPU for the conventional serial communication. At power-on, the main CPU sends the data frame to request the status information of each distributed remote I/O unit. After receiving the status information from all units, connection status of each unit is checked. When all distributed remote I/O units are available for the communication, the online communication mode (normal I/O mode) is started. The main CPU generates output data frame and sends the frames to each distributed remote I/O unit. The main CPU then receives data frames sent from each unit. When no error is found with the data frames, the main CPU processes the input data. When an error is found with the data frames, the input data is discarded. These procedures are repeated in the online communication mode.

When software processing of the communication control part of the NC unit is performed by the main CPU, the load for the main CPU is increased. Furthermore, increasing of the distributed remote I/O units causes to increase the load. However, as mentioned in the description for Fig 4, real time processing for the CNC must be performed at a predetermined time period and the drive control command must be completed within a certain time limit. To ensuring the accuracy of real time control, there are issues that a further high-performance main CPU is required, which results in increased cost.

# 3 PROPOSED METHOD

## 3.1 Distributed Remote I/O Control Method

The feature of the distributed remote I/O control method introduced in this paper is as follows. In the first place, simplification of communication procedure and reduction of data enable to achieve the communication between the NC unit and the distributed remote I/O unit only by electronic circuits. In the second place, the definition of the supposed failure mode in advance achieves the fail-safe system without performing software processing. This method enables downsizing of the system, lowers costs, improves reliability, and enhances the safety.

Figure 8 shows a configuration of the CNC using the distributed remote I/O control method. In Fig. 8, the machine I/O control interface of the NC unit is used as the distributed remote I/O communication control part. The I/O control circuits are separated from the NC unit and contained in the distributed remote I/O units. The machine requiring numerous I/O control points can be supported by increasing the number of distributed remote I/O units. Consequently, the NC unit cabinet volume is no more dependent on the number of I/O control points.

## 3.2 Hardware for Distributed Remote I/O Communication

The increased cost is the major issue to perform machine I/O control for CNC using serial communication. Proposals for cost reduction are as follows:

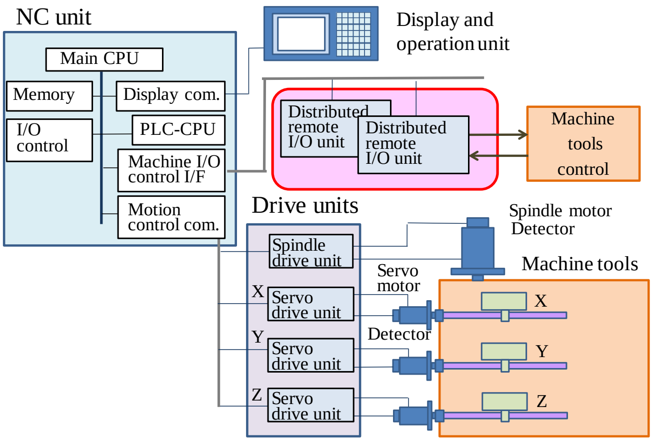



Figure 8: CNC Configuration Using the Distributed Remote I/O Control Method.

(1) Reduction of the number of signal communication wires

A half-duplex method is adopted for serial communication between the NC unit and the distributed remote I/O units. As compared to a full-duplex method, the number of communication wires can be reduced to one, and the wire rod, connector, and cable manufacturing costs can be reduced. In order to establish highly reliable data communication with the half-duplex method, a dedicated time-dividing communication procedure is defined to fit the characteristics of the CNC.

- Adoption of the EIA-485 differential system for the communication physical layer (data communication circuit)
- Adoption of the HDLC communication method
- In order to establish half-duplex communication, the communication cycle order is determined, and data transmission is enabled only during the transmission period.

(2) Elimination of CPU dedicated to communication

The conventional I/O control method for serial communication in Fig. 6 requires using CPU in the communication control part. For elimination of CPU dedicated to communication, simplification of communication procedure is needed. The frame size and data format suitable for the machine I/O control of the CNC functions are defined to minimize the communication processing. The communication pattern is defined to perform periodic communication in automatic synchronization with the CNC internal processing. Furthermore, processing at the time of power-on and processing at error are defined in the process of defining the data format and the communication procedure. These definitions achieve serial communication between the NC unit and the distributed remote I/O units without using CPU.

(3) Integration of the communication control part hardware into a one-chip LSI

With the predetermined frame size, data format, and procedure, the electronic circuits of the communication control part perform communication without using

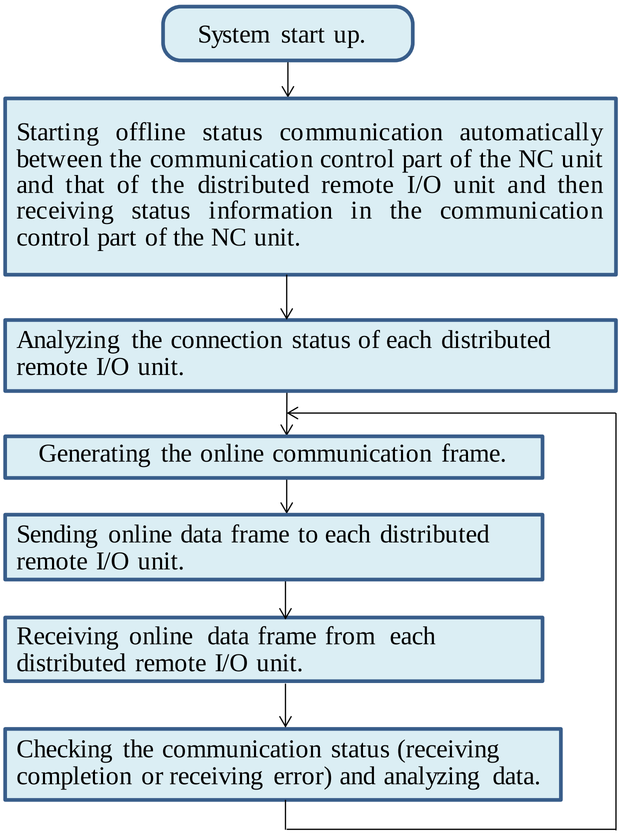



Figure 9: Processing Flow for Distributed Remote I/O Control Method.

CPU. It enables integrating the communication control part into a one-chip LSI both on the NC unit and the distributed remote I/O unit sides, downsizing and cost reduction of the communication control part.

(4) Simplification of the software processing by the main CPU

For I/O signal control processing of the NC unit, the main CPU reads the input data from the I/O buffer and writes the data to the I/O buffer in the communication control part in Fig. 6. These operations are performed either for the parallel I/O control or the I/O control for serial communication. Therefore, software processing by the main CPU can be simplified because the hardware automatically communicates with the distributed remote I/O units as mentioned in (2) and updates the data in the I/O buffers. Software processing to input or output the data can be performed without being aware of using serial communication.

(5) Simplification of processing on the distributed remote I/O unit side

The distributed remote I/O unit has the I/O control circuits which was built into the NC unit conventionally, and the one-chip LSI, mentioned in (3) in the communication processing part.

In remote I/O communication processing for the CNC, data frames are periodically generated from the output buffer at a predetermined time interval and sent to the distributed remote I/O units. The communication processing part of the distributed remote I/O unit receives data frames sent from the NC unit. When no error is found with the data frames, the distributed remote I/O unit outputs the received data as the machine control signal. Then, after the predetermined period of time, the distributed remote I/O unit generates data frames using the data taken as the machine input and sends them to the NC unit. This series of operations are performed without using CPU dedicated to distributed remote I/O unit communication or performing software processing.

(6) Ensuring reliability and safety of the system

To ensure reliability and safety of the system, processing at the time of power-on and at error are defined in the process of defining the data format and the communication procedure. Also, a fail-safe operation is defined for the machine I/O signal control at error.

## 3.3 Distributed Remote I/O Communication Procedure

In order to perform two-way serial communication between the NC unit and the multiple distributed remote I/O units, the NC unit performs a time dividing communication with each unit. The following two modes are defined for communication: the offline status communication mode, and the online communication mode (normal I/O mode). The two modes can be distinguished by the difference in the frame header pattern.

Based on the status information of the distributed remote I/O units stored in its communication control part, the NC unit generates output data frames in the online communication mode and sends the frames to the distributed remote I/O units. The NC unit then receives data frames sent from the distributed remote I/O units. When no error is found with the data frames, the NC unit processes the input data. When the count for errors found with frames sent from the distributed remote I/O units exceeds a predetermined value, the NC unit determines that a system error has occurred and stops the system.

## 3.4 Restraints on the Communication Procedure

As a CPU-less approach is used for communication processing in the distributed remote I/O control method, the following restraints apply to the communication.

(1) When the distributed remote I/O units receive the data sent from the NC unit, they send data to the NC unit after a predetermined time period because two-way communication is performed using one communication line.

(2) The NC unit sends data to the multiple distributed remote I/O units in a time dividing manner. Each unit send data to the NC unit only when the header pattern of the received data matches the predetermined station number of the own station.

(3) The two communication modes, the offline status communication mode and the online communication mode, are defined. The mode is switched between those two in accordance with the header pattern.

(4) It is necessary to determine the amount of data transmitted from the distributed remote I/O units to the NC

unit and the number of stations (the distributed remote I/O units) in advance.

# 4 IMPLEMENTATION SCHEME

## 4.1 Circuitry to Implement the Distributed Remote I/O Control Method

Figure 10 shows the configuration of the CNC for the distributed remote I/O control method. Eight distributed remote I/O units can be connected with communication network. (In Fig. 10, the distributed remote I/O unit stations from #2 to #6 are eliminated.) Half-duplex communication is used for serial communication. The communication control part (master function) and I/O buffer are integrated into a one-chip LSI in the NC unit. The communication control part (slave function) and control circuit such as multiplexer are integrated into a one-chip LSI in the distributed remote I/O unit. Also, half-duplex communication is established by controlling the communication control signal (RTS signal) to achieve reducing the number of communication signal wires. Input signals from sensors and a setting switch are selected by the switching signal (Mode signal) in the distributed remote I/O unit. In the offline status communication mode, setting switch information of the distributed remote I/O unit is send to the NC unit. In the online communication mode, sensor information obtained through input control circuit is send to the NC unit. Output data received from the NC unit is output to actuators through output control circuit.

The input and output control circuit are separated from the NC unit and contained in the distributed remote I/O unit. When it is required to increase the number of I/O control points, it is thus possible to increase the number of distributed remote I/O units and connect them to the serial communication line.

## 4.2 Implementation of the Distributed Remote I/O Control Method

In the distributed remote I/O control method, the NC unit state changes among the following: initial operation at power-on, offline status communication mode, and online communication mode. After initialization at power-on, the NC unit obtains information on the type and settings of the distributed remote I/O units in the offline status communication mode. After this, the NC unit mode transfers to the online communication mode. Figure 9 shows the processing flow in the NC unit. Figure 11 is the timing chart for shifting to the online communication mode.

The NC unit sends data frames in synchronization with the clock of the remote I/O communication cycle. After a predetermined time period, the distributed remote I/O units send the data frames to the NC unit. Communication with eight distributed remote I/O units can be performed in one cycle. When the main CPU sends the mode switching instruction signal (MPU-MODE) to change the mode to the online communication mode, the synchronous signal (SYNC-MODE) is output in synchronization with the clock to change the header pattern of the data frame. The communi-

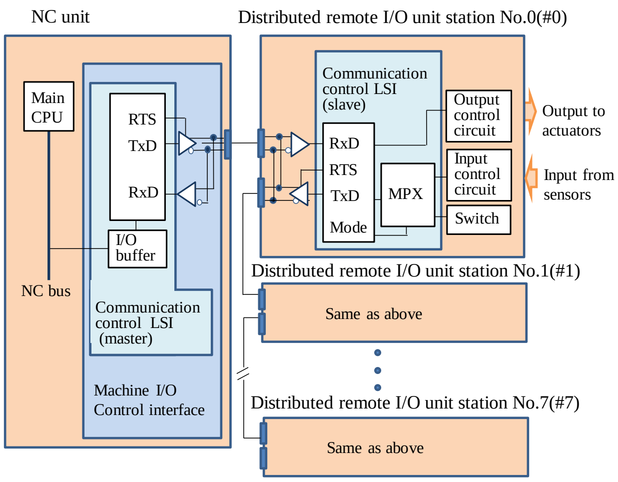



Figure 10: Configuration for Distributed Remote I/O Control Method.

cation control part of the NC unit sends the data frames repeatedly until it receives the response of normal reception completion from all distributed remote I/O units. After the NC unit receives the response of normal reception completion from all distributed remote units, the complete signal (STS-FIN) is output to change the mode to the online communication mode. (Figure 11 shows an example of the complete signal (STS-FIN). When no communication error occurs in the first cycle of mode switching, STS-FIN is output at the end of the first cycle. When a communication error occurs in the first cycle of mode switching and the NC unit receives a response of normal reception completion in the second cycle, STS-FIN is output at the end of the second cycle.)

Figure 12 is the timing chart for transferring of communication frames between the NC unit and eight distributed remote I/O units, and shows the data frame configuration. Each communication control signal (RTS) controls a sending data frame either in the NC unit or the distributed remote I/O unit. (In the Fig. 12, RTS signals of the distributed remote I/O unit stations from #4 to #7 are eliminated)
The NC unit sequentially sends data frames to multiple distributed remote I/O units in a time dividing manner, and each distributed remote I/O unit sends a data frame to the NC unit after a predetermined time period. The maximum number of stations is defined in advance for connecting distributed remote I/O units. The NC unit can sequentially send data to each distributed remote I/O unit, and receive the data sent from each distributed remote I/O unit. In this control method, the NC unit can complete the data communication with the distributed remote I/O units within a one-cycle interval. By repeating this procedure, the NC unit can perform the real time processing of the data I/O with a fixed cycle.

Every station is controlled by polling with using half-duplex serial communication. When the number of communication station is increasing, the delay time becomes grater. In the distributed remote I/O control method, the maximum number of station is defined eight on a serial communication line. The communication stations are controlled by polling. To perform the machine I/O control for the CNC, I/O data

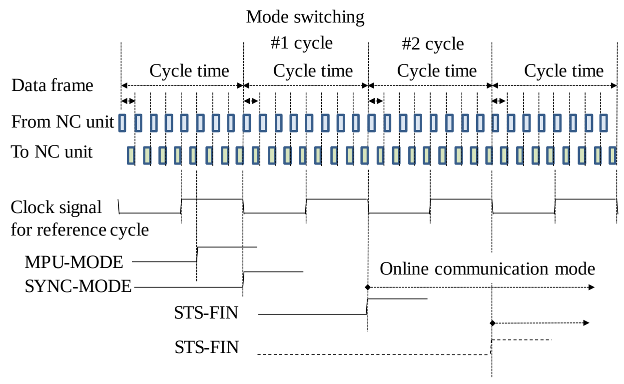



Figure 11: Timing Chart for Communication Mode Switching.

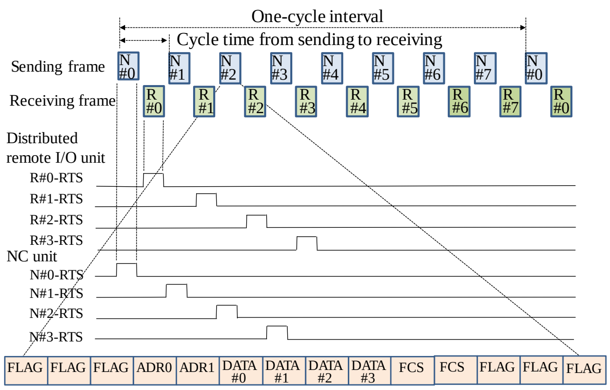



Figure 12: Timing Chart and Data Frame Configuration for the Distributed Remote I/O Control.

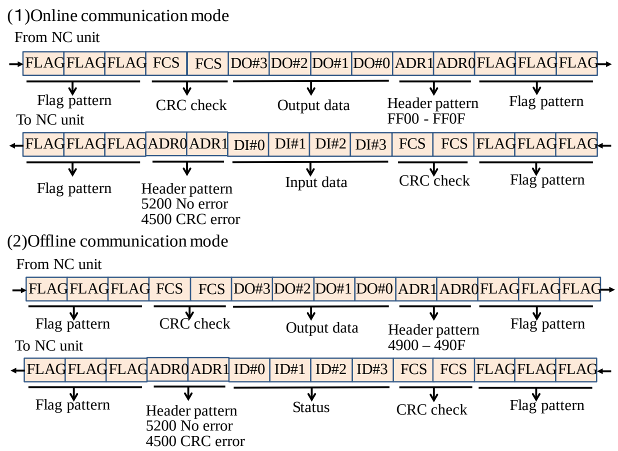



Figure 13: Data Format of Send and Receive Frames.

has to update within a one-block processing period of the CNC as shown in Fig. 4. In the implementation, a communication processing period (as shown a one-cycle interval in Fig. 12) is set to be fast enough to update I/O data more than 10 times within a one-block processing period.

Figure 13 shows the data format of send and receive data frames in each mode. The length of the data frame is fixed 14 bytes. The data frames consist of the flag pattern (3bytes), the header pattern (2 bytes), the data (4 bytes), the CRC check (2 bytes) and the flag pattern (3bytes). A flag sequence is assigned in the first part and the last part of the frame. The flag is a byte in a general data frame of HDLC. In this distributed remote I/O communication, the flag pattern is composed of 3 bytes to prevent malfunction when a flag pattern is misrecognized. The header pattern is used to control communication and to recognize the station number. The length of data is fixed 4 bytes. 32 bits data is small enough to control devices and get information from sensors of machine tools.

(1) These are the send frames from the NC unit to the distributed remote I/O unit and from the distributed remote I/O unit to the NC unit in the online communication mode (normal I/O mode).The header patterns from FF00 to FF0F which consist of 16-bit data are assigned for the online communication from the NC unit to the distributed remote I/O unit. The header pattern 5200 is assigned for the normal response header pattern from the distributed remote I/O unit to the NC unit, and 4500 is assigned for the error response header pattern from the distributed remote I/O unit to the NC unit.

(2) These are the send frames from the NC unit to the distributed remote I/O unit and from the distributed remote I/O unit to the NC unit in the offline status communication mode. The header patterns from 4900 to 490F are assigned for the offline status communication from the NC unit to the distributed remote I/O unit. The response header patterns from the distributed remote I/O unit to the NC unit are the same as the online communication. DO#0-3 are send data (output data) to the distributed remote I/O unit and DI#0-3 are send data (input data) to the NC unit. ID#0 is the identification code on the type of the distributed remote I/O unit and ID#1 is the information on settings of the distributed remote I/O unit. ID#2 and ID#3 are other status information.

Figure 14 shows I/O control circuits of a distributed remote I/O unit for the normal digital I/O control. Each bit of the I/O data has its own meaning. When the NC unit sends data in a data frame to a distributed remote I/O unit, the data is then output from the distributed remote I/O unit as its output signal. Then, when the distributed remote I/O unit takes in an input signal, the distributed remote I/O unit sends the data in a data frame to the NC unit, and the data is then used as the input data in the NC unit.

## 4.3 Implementation of the Distributed Remote I/O Units

By defining the data format for the distributed remote I/O control method, versatility was achieved for the communication function. Also, the I/O functions of the NC unit are integrated to use the control method through the distributed remote I/O units, and not dependent on the conventional individual interface circuits. Thus, it is possible to configure CNC using the distributed remote I/O units.

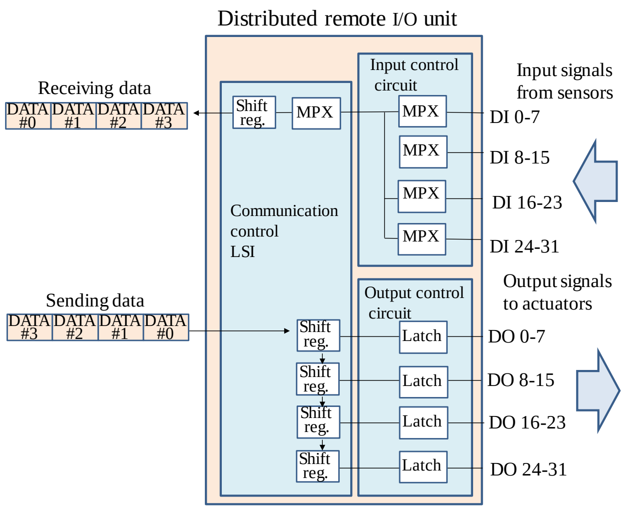



Figure 14: I/O Control Circuits of the Distributed Remote I/O Unit.

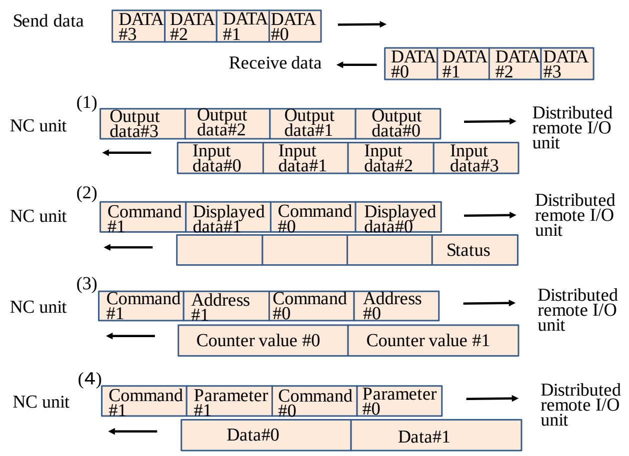



Figure 15: Examples of Communication Data Format for Distributed Remote I/O Control Method.

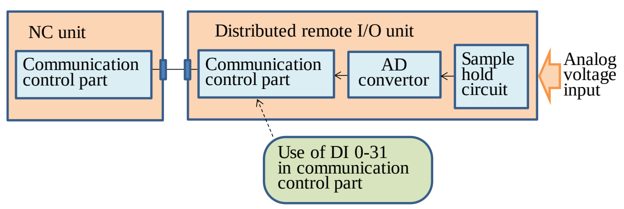



Figure 16: Configuration for Inputting Analog Voltage Information.

In the transmission data format, the data sent from the NC unit can be used as a command or output data, and the data received by the NC unit can be used as input data or status data. Each set of 8/16/32 bits of the data has its own meaning. Figure 15 shows the examples of various types of communication data format for distributed remote I/O. In Fig. 15, the example (1) shows the data format for the operation shown in Fig. 14. Each bit of the I/O data has its own meaning. The example (2) shows the following data formats: one which is sent from the NC unit to the display for combining the command to display data and the data itself, and one which is sent from the distributed remote I/O unit to the NC unit for reporting the display status data.

The example (3) shows the following data formats: one which is sent from the NC unit to the distributed remote I/O unit for sending the command to read the number of pulses generated from the manual pulse generator in a built-in circuit of the distributed remote I/O unit and the address to specify the manual pulse generator's station number, and one which is sent from the distributed remote I/O unit with 16-bit data of pulse number. The example (4) shows the following data formats: one which is sent from the NC unit to the distributed remote I/O unit for sending the command to read analog voltage information and the parameter to specify the station number of an analog voltage input circuit, and one which is sent from the distributed remote I/O unit with 16-bit data of analog voltage information.

In the conventional CNC, the necessary I/O interface circumethod, individual I/O interface circuits are provided in the distributed remote I/O units. Thus, the I/O functions required for the NC unit are achieved by using the distributed remote I/O units. Examples of the functions are as follows: outputting display data sent from the NC unit to the display, outputting the analog voltage command, reading the number of pulses of the pulse generator into the NC unit, or reading the analog voltage information.

Figure 16 shows a distributed remote I/O unit configuration for inputting the analog voltage information to the NC unit when the CNC contains distributed remote I/O units. When the NC unit sends a command to read analog voltage data to the distributed remote I/O unit which has an analog voltage input circuit, the distributed remote I/O unit detects the command in its communication control circuit, performs AD conversion of the analog voltage input through its sample hold circuit, and send the digitalized voltage data as its transmission data to the NC unit. The NC unit reads the analog voltage information at a predetermined time period and provides necessary controls on the real time basis.

## 4.4 Implementation of the Distributed Remote I/O Communication Procedure

In this section, we explain about the distributed remote I/O processing flow in the CNC unit and the distributed remote I/O units shown in Fig. 17.

### 4.4.1 Initial Operation at Power-on and Offline Status Communication Procedure

(1) After initialization at power-on, the distributed remote I/O unit reads its own station number from the hardware setting (setting switch). The offline status communication mode is the initial setting for the distributed remote I/O units.

(2) After initialization at power-on, the communication control part of the NC unit sequentially requests each distributed remote I/O unit in the offline status commu-

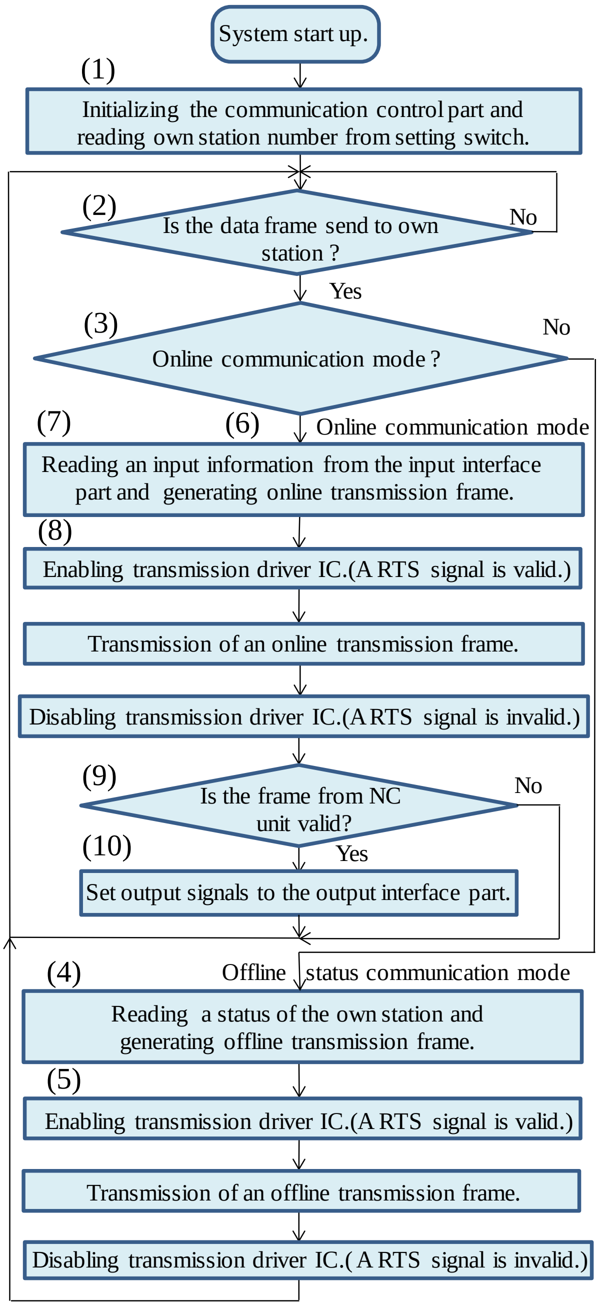



Figure 17: Processing Flow for Distributed Remote I/O Control Method.

nication mode in a time dividing manner to send the type and setting information of the unit. Each unit checks the header pattern (address part) of the data sent from the NC unit, and receives the data only when the header pattern corresponds to its own setting switch.

(3) The distributed remote I/O units distinguish the offline status communication mode from the online communication mode, which is the normal I/O mode, by the difference of the header pattern (address part) of the data frame sent from the NC unit.

(4) In the offline status communication mode, when the distributed remote I/O unit recognizes that the data is addressed to its own station and receives the data, the distributed remote I/O unit generates a data frame which contains the status information of its own station (unit type and setting information) in the data part.

(5) After a predetermined time period, the distributed remote I/O unit determines the communication control signal RTS as valid, and sends the data frame to the NC unit. The communication control part of the NC unit sequentially sends requests to obtain the type and setting information from the distributed remote I/O unit in a time dividing manner, and obtains the information on the number of stations and the unit type for all units connected with the one communication line. Then, the NC unit obtains the status information of all the distributed remote I/O units connected. After confirming the normal status for all of the distributed remote I/O units, the NC unit switches the mode from the offline status communication mode to the online communication mode.

### 4.4.2 Online Communication Procedure

(6) In the online communication mode, the NC unit periodically sends the data frame to each distributed remote I/O unit which contains the output data addressed to each unit. Each distributed remote I/O unit checks the header pattern of the data sent from the NC unit, and receives the data only when the header pattern corresponds to its own setting switch.

(7) The distributed remote I/O unit receives the data frame addressed to its own station, reads the input data from the input signal interface circuit, and generates a data frame which contains the input data.

(8) After a predetermined time period, the distributed remote I/O unit determines the communication control signal RTS as valid, and sends the data frame to the NC unit.

(9) The distributed remote I/O unit checks the data frame sent by the NC unit for any error.

(10) When no error is found in the data frame, the distributed remote I/O unit determines that the output data in the data frame is valid, and sends the output data to the interface circuit. Thus, the distributed remote I/O unit receives the data sent by the NC unit, and then outputs the data to the machine tools or actuators. When any error is found in the data frame, the previous data is retained as the output data.

(11) By repeating steps from (7) to (10) above, the NC unit performs sequence control for the machine tools.

## 4.5 Implementation of Safety and Reliability

In the distributed remote I/O control system, communication processing is performed with a fixed cycle in a CPU-less hardware configuration and the following operations are performed to ensure communication reliability and safety of machine tools.

(1) When the distributed remote I/O unit does not detect any data frame sent from the NC unit for a pre-

determined time period or longer, the distributed remote I/O unit automatically resets its own output.

(2) When the distributed remote I/O unit detects an error in the received data frame, the distributed remote I/O unit does not update the output signal. The distributed remote I/O unit changes the header pattern of the data frame and sends the data frame to the NC unit to let the NC unit recognize the data frame error.

(3) The NC unit stops the system when it does not receive the data frame sent from the distributed remote I/O unit.

(5) The NC unit stops the system when the count for errors found with frames sent from the remote distributed I/O units exceeds a predetermined value.

# 5 EVALUATION

## 5.1 Downsizing of the NC Unit and the Distributed Remote I/O Units

In the distributed remote I/O control method, a half-duplex communication method is adopted to simplify the communication wiring/circuit configuration. In addition, each electronic circuit is integrated into an LSI in the communication control part either on the NC unit or the distributed remote I/O unit side. Furthermore, the development of the device package technology and the implementation technology enabled electronic circuits to implement on a compact size board. The distributed remote I/O control method enabled to modularize the components of the CNC and connect them with serial communication network. This control method and the development of technologies mentioned above enabled the use of the more compact NC unit and distributed remote I/O units. Figure 18 shows a comparison of cabinet volume by CNC generation.

In the 1st generation CNC, many DIP (Dual Inline Package) devices were used for the electronic circuits. The I/O control circuits were provided in the NC unit. When the machine required numerous machine control signals, numerous circuit boards were required for machine I/O control, which resulted in increase in the NC unit cabinet volume. In the 2nd generation CNC, the device package was changed from DIP to SMT (Surface Mount Type).
Also, the I/O control circuits were separated from the NC unit to the distributed remote I/O control unit. Therefore, the NC unit could be significantly downsized owing to the high integration of the electronic circuits and downsizing of device. In the 3rd generation CNC, the NC unit volume was 20% smaller than the conventional one.

Instead, the I/O control circuits were contained in the distributed remote I/O units. Owing to the integration of the additional circuits into an LSI in the communication control part and downsizing of devices, the distributed remote I/O unit volume was 50% smaller than the conventional I/O control circuit in the 1st generation CNC.

By adopting the distributed remote I/O control method in the products, the system volume of the NC unit and the distributed remote I/O unit in the 3rd generation CNC was 25% smaller than the conventional one, which contributed to the downsizing of the cabinet.

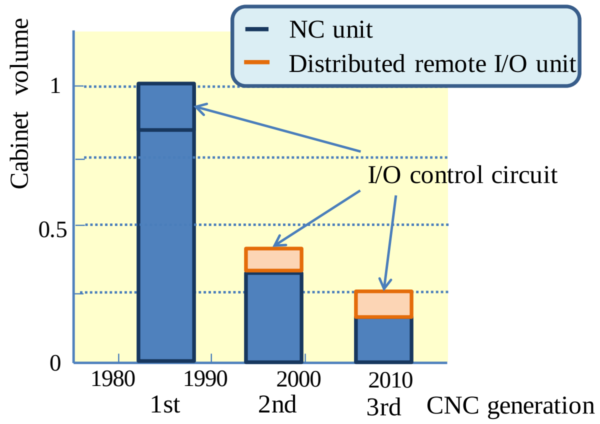



Figure 18: Comparison of Cabinet Volume by CNC Generation.

## 5.2 System Cost Reduction

### 5.2.1 HARDWARE COST REDUCTION

In the distributed remote I/O control system, a half-duplex method using a single signal wire was adopted for communication. Therefore, costs involved with serial communication could be reduced for wire rods, connectors, etc. In the 1990s, the development of ASIC which integrated CPU and user circuits was not practical or widely available yet. By simplification of communication procedure, the communication control part was configured using electronic circuits without using CPU and memory. Furthermore, we cloud to develop communication LSIs which integrated the electric circuits into a one-chip LSI, reducing the circuit cost by one-third or more as compared to the conventional communication method. The cost was almost equivalent to that of the conventional configuration for which serial communication was not used.

It was significantly beneficial for cost reduction to configure the communication control part without using CPU. Considering the fact that the LSIs are still used in today's products, it can be said that the use of LSI still achieves high enough cost performance.

### 5.2.2 Simplification of the Software Processing

After power-on, the hardware-configured communication control part of the NC unit automatically entered the offline status communication mode, and sequentially sent data frames to the distributed remote I/O units, requesting status information. When the distributed remote I/O units connected in the system received the data frame addressed to their own stations, they automatically sent their status information to the NC unit. As the status information was sent to the communication control part of the NC unit, the NC unit only had to check the status of each distributed remote I/O unit during software processing for initialization. Thus, it was possible to simplify the software processing. In the online communication mode, the NC unit writes the control command and parameters in the I/O buffer of the communication control part. After a predetermined time period, the NC unit reads the I/O buffer of the communication control part. It was possible to simplify the software processing because the information of the external devices could be easily read through the distributed remote I/O units without much concern for the serial data communication.

In the distributed remote I/O control method, the simplification was achieved without increasing the load for the main CPU to perform communication processing. Therefore, it was not necessary to replace the main CPU of the NC unit with a high-performance CPU, which enabled suppression of cost increase.

## 5.3 Flexible System Configuration

The number of I/O signal control points depends on the machine tools configuration. In the conventional NC unit, the number of I/O control points of each class of the NC unit is fixed for its hardware configuration. By adopting the compact-sized distributed remote I/O units in the products and configuring the system using multiple units as required, it was made possible to configure the machine I/O control part of the CNC flexibly to support the number of I/O control points required by the machine tools.

Also, versatility was achieved for the communication function by defining the command and the data as shown in the examples of communication data format in Fig. 15. Thus, it was possible to use distributed remote I/O units with a wider variety of functions as follows: other than for input/output of the normal I/O signals, it was possible to use distributed remote I/O units for outputting the data to the display, inputting the number of pulses and outputting or inputting the analog voltage signal to and from the peripheral devices. The wide selection of the distributed remote I/O units allowed the NC unit configuration to support the I/O type or the number of points of the machine to be controlled, and flexibility of system configuration was further improved. Table 1 shows the distributed remote I/O unit types.

The conventional CNC requires wire connections from one power panel to all machine components in the system. By using distributed remote I/O control method, the distributed arrangement of the NC unit, drive units, and distributed remote I/O units are made possible according to the configuration of the machines.

For example, when a machine has a building block structure and auxiliary devices may be added depending on specifications, it is possible to make a power panel for the machine and connect it to the machine, and make dedicated power panels for auxiliary devices and arrange the distributed remote I/O units to be connected to the power panels. Thus, it was made possible to configure a system by connecting the power panel of the machine and the power panel of the auxiliary devices using serial communication. This approach did not need the change on the machine power panel side. The auxiliary devices could be added only by connecting serial communication cables. Thus, it was made possible to reduce additional wiring works, add optional machine tools functions easily, and increase flexibility of the system.

Table 1: List of Distributed Remote I/O Units.

| Distributed remote I/O unit type | Digital input | Digital output | Pulse counter | Analog input | Analog output |
|---|---|---|---|---|---|
| A | 32 | 32 | | | |
| B | 64 | 48 | | | |
| C | 64 | 48 | | | 1ch |
| D | 32 | 32 | 2ch | | |
| E | 32 | 32 | | 4ch | 1ch |

Also, owing to the downsizing of the NC unit, the NC unit can be flexibly placed on various locations, not only inside the power panel as before, but also on the back side of the display part of the display and operation unit, for example.

## 5.4 Reliability and Safety of the System

### 5.4.1 Reliability at Power-on

In the distributed remote I/O control method, the CNC is configured by connecting the NC unit and the distributed remote I/O units through communication. We evaluated whether the NC unit recognizes distributed remote I/O units connected to the NC unit, and how the system operates when misrecognition occurs after initialization at power-on.

After initialization at power-on, the NC unit requested information on the type and settings of the distributed remote I/O units in the offline status communication mode. After this hardware information was obtained, the NC unit switched the mode to the online mode. In the above operation, when the actual machine information was not consistent with the NC unit setting information, inconsistency was regarded as an alarm and the CNC operation did not start. As a result, we confirmed that the actual system configuration is checked without fail when the system mode transfers to the normal online mode from the initial status after power-on, and the system is highly reliable in preventing malfunction when a connected device is misrecognized.

### 5.4.2 Reliability and Safety at Fault Conditions

In terms of communication functions, we evaluated how the reliability and safety of the system can be assured in case of communication failures between the NC unit and the distributed remote I/O units.

Under normal conditions, the NC unit periodically sends data to the distributed remote I/O units and receives data from the distributed remote I/O units after a predetermined time period, following a predetermined procedure. When a communication break or cable disconnection occurred, no data frame was sent from the distributed remote I/O units, the NC unit determined that a communication break or cable disconnection was occurring, and stopped the system.

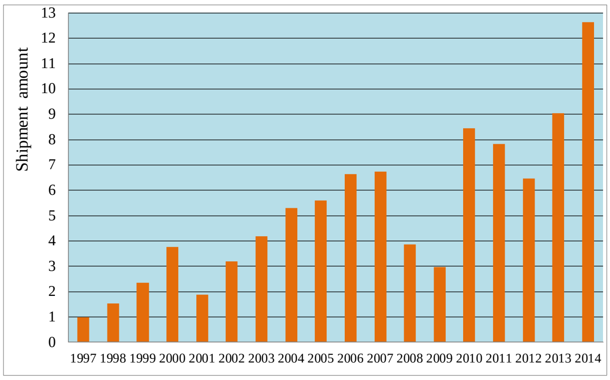



Figure 19: Annual Shipment Amount of the LSIs/IPs for Communication Control Part.

When noise was present and the count for errors in the data frame sent by the distributed remote I/O module exceeded a predetermined value, the NC unit also stopped the system.

We confirmed that the NC unit assures system safety by monitoring the data frame sent from the distributed remote I/O units.

On the other hand, the distributed remote I/O units check the communication status with the NC unit by monitoring the data frame sent from the NC unit.

When an incident such as a cable disconnection or break occurred, the distributed remote I/O unit reset the machine control signal output since it did not detect a data frame sent from the NC unit for a predetermined time period. When the NC unit stopped the system for some reasons, the NC unit detected the system stop with a watchdog using its internal timer, and the signal sent from the distributed remote I/O units was turned off. The distributed remote I/O unit detected that no data frame was sent from the NC unit for a predetermined time period, and reset the machine control signal output. Based on the above, we confirmed that the machine control signal can be reset even when the NC unit enters an error status and stops the system. Also, when the distributed remote I/O unit detected an error in the received data frame, the distributed remote I/O unit held and did not update the machine control signal output.

We confirmed that the above measures ensure the safety of the highly reliable system against abnormal stops of the NC unit or communication failures such as communication breaks.

## 5.5 Widespread Use of the Distributed Remote I/O

The CNC using the distributed remote I/O control method started to be shipped to the market in the late 1990s. The hardware circuits of the communication control part, a core of the distributed remote I/O control, were integrated into a one-chip LSI in the communication control part. Two types of LSIs were developed: one with a master function is equipped on the NC unit, and the other with a slave function is equipped on the distributed remote I/O units.

Owing to the subsequent development of semiconductor technologies, the LSI of the communication control part is also used as the intellectual property (IP) core of FPGA or ASIC. Figure 19 shows the annual shipment amount of the LSIs for the communication control part with the master or slave function and the IP cores. The annual shipment amount in 1997 when the shipment started is used as reference. The distributed remote I/O control method is now widely available in the market. The most recent annual shipment amount of products is more than 10 times of that of the first year of shipment, and the cumulative number exceeded 5,000,000.

# 6 CONCLUSION

We reported on the distributed remote I/O control method in the CNC. When serial communication started to be widely available in the 1990s, integrating CPU and user circuits into a one-chip LSI was not practical or widely available yet. Using communication CPUs caused increasing cost and decreasing power of competitive products. Simplification of communication procedure and reduction of data enabled to achieve the communication between the NC unit and the distributed remote I/O unit only by electronic circuits and integration into a one-chip LSI. With this communication control method, we established a compact and low-cost network for machine I/O control of CNC, without compromising the real time performance of the CNC. Thus we modularized the components of the conventional CNC according to their functions and achieved downsizing and optimization of the individual modules. Consequently, we achieved distribution of the component modules of the NC unit and improved flexibility of system configuration while maintaining high reliability and safety of the system. Modularization and distribution of the modules by the distributed remote I/O control method herein was a pioneering solution. The concept of the idea is the origin and the inheritance for the today's CNC.

The basic idea for this I/O control method is to define the communication control method suitable for the real time control for the CNC. Specifying the purpose and simplifying the related information are the key factors to solve issues. We presume that the concept of the idea will also be meaningful in the coming age of IoT. We anticipate further contributions to the development of the new systems for manufacturing in the future.

# REFERENCES

[1] J. Wan, H. Cai, and K. Zhou, “Industrie 4.0: Enabling Technologies,” IEEE/ICIT, 2014 International Conference on, pp. 135 - 140 (2015).

[2] F. Tao, Y. Zuo, L. D. Xu, and L.Zhang, “IoT-Based Intelligent Perception and Access of Manufacturing Resource Toward Cloud Manufacturing,” IEEE Tran. Industrial Informatics, Vol.10, No.2, pp.1547 – 1557(2014).

[3] F. Tao, Y. Cheng, L. Xu, L. Zhang, and B. H. Li, “CCIoT-CMfg: Cloud Computing and Internet of Things-Based Cloud Manufacturing Service System,” IEEE Tran. Industrial Informatics, Vol.10, No.2, pp.1435 – 1442(2014).

[4] O. Sauer, "Developments and Trends in Shopfloor-related ICT Systems," IEEE/IEEM, 2014 IEEE International Conference on, pp. 1352 - 1356 (2014).
[5] J.-D. Decotignie, J.-C. Gregoire, "Integrating the Numerical Controller and the FMS," IEEE/IECON, 15th Annual Conference of IEEE, Vol.3. , pp. 675 - 680 (1989).
[6] L. Du, C. Duan, S. Liu, and W. He, "Research on Service Bus for Distributed Real-time Control Systems," IEEE/ITAIC, 2011 6th IEEE Joint International,Vol1, pp.401-405 (2011).

(Received September 30, 2015)
(Revised January 28, 2016)

**Akihiro Yamashita** received the B.E and M.E. degree in Precision Engineering from the Kyoto University, Japan in 1977 and 1979, respectively. In 1979, he joined Mitsubishi Electric Corp. He received the M.E. degree in Mechanical Engineering from the Carnegie Mellon University, USA, in 1987. In 2012, he joined Mitsubishi Electric Engineering Corp. Since 2014, he has been a graduate school student at Shizuoka University, Japan. His research interests include control engineering and computer science. He is a member of Information Processing Society of Japan

**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. He is currently an Associate Professor. His research interests include intelligent sensor network system as well as IoT. He is a member of IEEE, ACM, IEICE, IPSJ and Informatics Society.

**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. From 1993 to 2011, he had been a Professor at Shizuoka University, Japan. Since 2011, he is a Professor at the Aichi Institute of Technology, Japan. His research interests include mobile computing, distributed computing, computer networks, broadcast communication and computing, and protocol engineering. He is a member of Information Processing Society of Japan, the Institute of Electronics, Information and Communication Engineers, the IEEE Computer Society and Informatics Society.

# Design of an Application Based IP Mobility Scheme on Linux Systems

Kohei Tanaka[†], Fumihito Sugihara[‡], Katsuhiro Naito[†], Hidekazu Suzuki*, and Akira Watanabe*

[†]Faculty of Information Science, Aichi Institute of Technology, Toyota, Aichi 470-0392, Japan
{kohei, naito}@pluslab.org
[‡]Department of Electrical and Electronic Engineering, Mie University, Tsu, Mie 514-8507, Japan
*Graduate School of Science and Technology, Meijo University, Nagoya, Aichi 468-8502, Japan

***Abstract*** - Internet of Thing (IoT) systems have been attracting attention as one of the solutions for new services in the Internet. They usually employ a client-server model due to a difficulty of end-to-end communication in practical networks. However, scalability will be a major issue in IoT systems because billions of IoT devices will be installed around the world in recent years. The authors have been proposed a new IP mobility mechanism called NTMobile (Network Traversal with Mobility) to realize end-to-end communication in IoT systems because end-to-end communication can improve system scalability by reducing traffic through servers. Conventional implementation employed a kernel module mechanism for NetFilter because the kernel module implementation is the best way of realizing high throughput performance. On the contrary, the kernel module should be maintained according to changes in NetFilter specifications. This paper designs an application based IP mobility scheme on Linux systems, where the developed IP mobility library can realize the IP mobility function in an application layer on Linux systems. As a result, developers can realize an end-to-end communication model by employing the enhanced IP mobility library. The proposed design ensures compatibility between the developed library and the conventional NTMobile. Therefore, developers can select the implementation scheme according to the required performance.

***Keywords***: IP Mobility, Accessibility, Application library, Linux, NTMobile.

## 1 INTRODUCTION

Recent microcomputer boards implement some network interfaces to cooperate with another microcomputer boards according to the increase of demand for IoT (Internet of Things) systems [1], [2]. Almost all microcomputer boards for IoT systems are usually installed in a private network due to a security policy and limitation of assignable global IP addresses. Typical private networks prohibit access from the global Internet to a node in their private networks. Therefore, interconnectivity between IoT devices is a big issue even if they should communicate with each other to realize their specific service. Additionally, operating systems select an interface to access to the Internet according to network condition of each interface and access policies when an IoT device is a mobile node [3]. Therefore, IoT systems also require a seamless connectivity scheme because their IP address typically changes when they switch the interface even if they should communicate with each other to realize their specific service.

IP mobility protocols are a solution to the requirement for inter-connectivity and seamless connectivity in IoT systems because they can realize continuous communication when an IP address for an interface changes due to switching of access networks [4]–[9]. They are classified into three types: IP mobility schemes for IPv4, IP mobility schemes for IPv6, and IP mobility schemes for both IPv4 and IPv6. Mainstream of IP mobility schemes is for IPv6 because IPv6 is suitable for mobility. On the contrary, the number of implementations for IPv4 is quite few though some mechanisms have been proposed [10]–[12]. DSMIPv6 (Dual Stack Mobile IPv6) [13] supports both IPv4 and IPv6 networks. However, it still does not support inter-connectivity between IPv4 and IPv6.

IPv4 continues to be the mainstream protocol under the present circumstances of the Internet. Additionally, some Internet service providers start the service with large scale network address translator (LSN) in order to meet the shortage of IPv4 global addresses [14]–[16]. As a result, IP mobility in a private network behind a NAT becomes an important issue. STUN (Simple Traversal of UDP through NATs) [17], TCP hole punching [18], TURN (Traversal Using Relay NAT) [19], and ICE (Interactive Connectivity Establishment) [20], [21] have been proposed for a NAT traversal. These conventional work requires implementation of the special mechanisms in an application. Additionally, some IoT devices may use IPv6 addresses because IPv6 networks have enough number of IP addresses and some devices are suitable for an IPv6 protocol stack. Therefore, inter-connectivity between IPv4 and IPv6 networks is also an essential function to realize inter-operation between IoT devices.

The authors have developed a new IP mobility technology called NTMobile (Network Traversal with Mobility) [22]–[24]. The features of NTMobile are an IP mobility and an accessibility in both IPv4 and IPv6 networks. Therefore, each client of NTMobile can communicate with each other even when they use a different IP protocol version because they can communicate with virtual IP addresses that are independent addresses from physical IP addresses. NTMobile systems have some servers: account server (AS), direction coordinator (DC), notification server (NS), relay server (RS), and cache server (CS). DC and RS serve an IP mobility and an accessibility functions for each client, AS serves an authentication service, and CS serves an offline data-exchange service. Our implementation assumes Linux systems. The implementation design is classified into a signaling daemon and a packet manipulation module for the Linux kernel. As a result, we have confirmed that our implementation scheme

ISSN1883-4566 

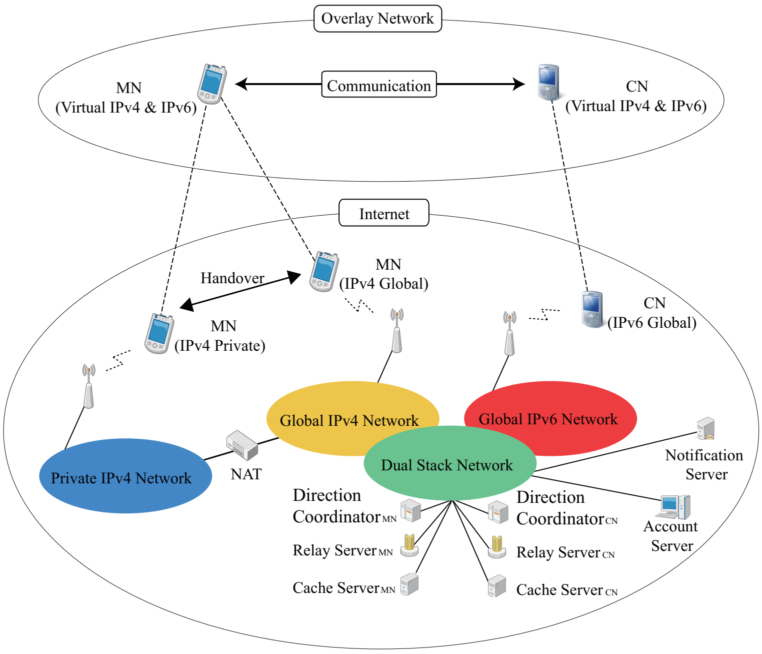



Figure 1: System model of NTMobile

can realize high throughput performance. On the contrary, we find a maintenance issue for the packet manipulation module because netfilter module in Linux systems sometimes changes their specification according to the upgrade of a Linux kernel.

This paper proposes a new design of an application based IP mobility for NTMobile nodes. The original design can provide NTMobile functions by an application library instead of the special kernel module. Therefore, application programmers can obtain IP mobility and accessibility functions by using of proposed application library.

# 2 NTMOBILE

NTMobile can realize IP mobility and accessibility for IPv4 and IPv6 networks. Figure 1 shows the overview of NTMobile system model. The NTMobile system consists of an account server (AS), some direction coordinators (DCs) with some relay servers (RSs), cache servers (CSs), Notification Servers (NSs) and NTMobile nodes. AS serves authentication service to all DCs, and each DC controls their RSs, CSs, NSs and NTMobile nodes. NTMobile node has IP mobility and accessibility functions by communicating with AS and its DC. Each DC has a virtual IP address pool for its NTMobile nodes, and assigns an address to each NTMobile node in a registration process. Each NTMobile node constructs a UDP tunnel between NTMobile nodes according to a signaling direction from its DC, and communicate with each other by using their virtual IP addresses. As a result, each NTMobile node can communicate continuously even if a real IP address at interfaces is changed because the virtual IP addresses are independent from physical IP addresses. The details of the system components are as followings.

## 2.1 Account Server (AS)

AS is an individual server that manages authentication information. Therefore, AS can distribute node information of each NTMobile node to initialize a setting for NTMobile nodes. Additionally, it bears responsibility for authentication by replying login response message when a NTMobile node makes inquiries about its authentication to AS. AS also distributes a shared key for encryption between the NTMobile node and DC.

## 2.2 Direction Coordinator (DC)

DC manages location information of each NTMobile node and indicates signaling processes for tunnel construction between NTMobile nodes. Each DC also owns the DNS (Domain Name Server) function. Therefore, DC can easily find another DC by searching a NS (Name Server) record in DNS. In addition, DC manages virtual IP addresses for NTMobile nodes to prevent a duplication of an address assignment. DC assigns them to each NTMobile node in a registration process. In the tunnel construction process, it indicates the tunnel construction processes to both NTMobile nodes.

## 2.3 Notification Server (NS)

NTMobile typically uses a UDP protocol to communicate between a NTMobile node and its DC. Therefore, the NTMobile node should send keep-alive messages to its DC when it uses a private address under a NAT router. NS can provide a notification service with a TCP connection between NS and the NTMobile node. Therefore, the employing NS can reduce the amount of messages for keep-alive. Additionally, NS also supports APNS (Apple Push Notification Service) for iOS and GCM (Google Cloud Messaging) for Android OS.

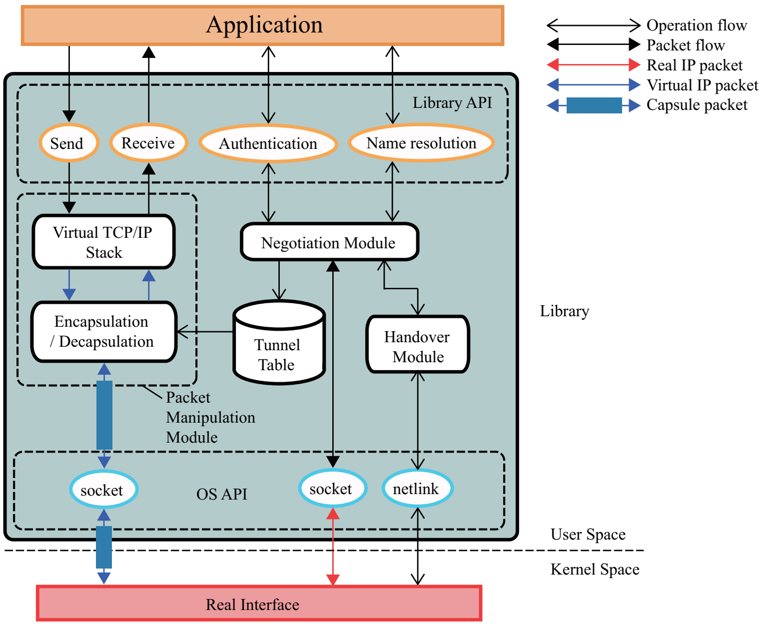



Figure 2: Overview of application library design

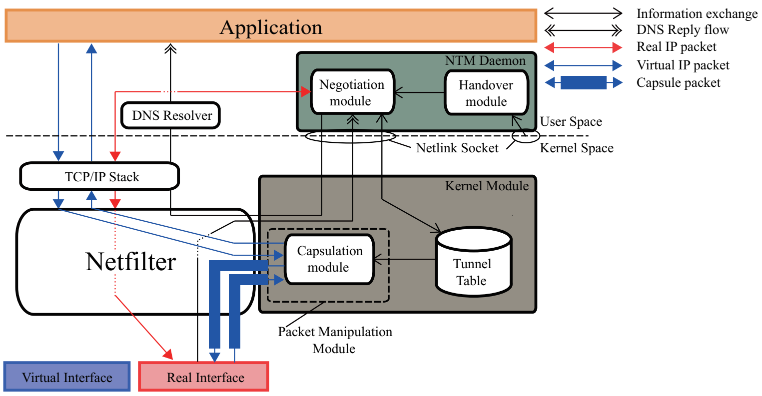



Figure 3: Overview of kernel module design

## 2.4 Relay Server (RS)

The relay server function is to relay tunnels between NTMobile nodes when both NTMobile nodes exist under different NAT routers or exist under different version networks such as IPv4 and IPv6 networks. DC manages some relay servers to realize load balancing and to avoid a single point of failure. It also chooses a relay server to activate the relay function for dedicated NTMobile nodes.

## 2.5 Cache Server (CS)

The function of cache servers is storing data temporary for non-realtime data exchange between NTMobile nodes. A correspondent NTMobile node can download the stored data since CS stores the data for a certain period.

## 2.6 NTMobile Node

Functions of NTMobile nodes are to realize IP mobility and accessibility in IPv4 and IPv6 networks. They obtain their own information from AS in the initialization phase when they connect to the NTMobile network at first. Then, they inform their own network information to their DC because DC should manage network information of each NTMobile node to realize IP mobility and accessibility. They also update the own network information when they switch access networks.

# 3 APPLICATION BASED IP MOBILITY

This paper proposes a new design of an application based IP mobility and accessibility for NTMobile nodes. The proposed design provides APIs (Application Programming Interface) for NTMobile functions: an initialization, an authentication, a tunnel construction request, and network sockets for NTMobile communication. It also provides a virtual IP address for the network sockets. Our API provides BSD compatible network sockets for application developers. Therefore, application developers can easily apply their network application with virtual IP addresses for NTMobile network.

### 3.1 System Model

Figure 2 shows the overview design of the proposed application based IP mobility and accessibility system for NTMobile nodes. Figure 3 shows the overview design of conventional kernel module. The conventional kernel module performs IP capsulation/decapsulation and encryption/decryption processes in Linux kernel. The design also provides a virtual network interface for applications. The conventional implementation consists of a shared library for common NTMobile functions, daemon programs for a NTMobile node, an account server, a direction coordinator, a relay server, and a notification server, and a kernel module for a NTMobile node. The source size of the conventional shared library, daemon program for a NTMobile node, and the kernel module is 23K lines, 10K lines and 12K lines respectively. Therefore the overhead of the NTMobile program is acceptable size for a small resource hardware.

On the contrary, the proposed application library is inserted between an original network socket in Linux OS and a developer's application. In the proposed design, we also employ the shared library to implement the common NTMobile functions. Additionally, we develop a new application library including functions for a transport layer, a daemon program of a NTMobile node and a kernel module. We employ lwIP [25] as a transport layer. The source size of lwIP and application library is 55K lines and 10K lines respectively. As a result, we think that our proposed design is still acceptable for a small resource hardware such as M2M and IoT devices. The application can request to the proposed application library for initialization of NTMobile node functions, an authentication for NTMobile network, and a tunnel construction request to a correspondent node. The actual data communication is performed with special network sockets for NTMobile functions. The application library can perform IP capsulation/decapsulation and encryption/decryption processes between the special network sockets and the original Linux network sockets.

### 3.2 Application Programming Interface

The proposed design provides APIs (Application Programming Interface) for NTMobile functions: an initialization, an authentication, a tunnel construction request, and network sockets for NTMobile communication. The following is the APIs for an application.

- Authentication API

  The proposed library assumes a specific authentication scheme with a user account and a password. Therefore, an application should request authentication with the user account information through the authentication API. The authentication API requests the negotiation module for authentication processes. The negotiation module should obtain the NTMobile node information when the application library has no configuration. Then, it also requests AS to authenticate with the account information. Finally, it registers the network information to the DC.

- Name Resolution API

  NTMobile network employs FQDN (Fully Qualified Domain Name) as an identifier for a NTMobile node. Therefore, an application should request a tunnel construction with a FQDN of a correspondent node through the name resolution API. The name resolution API requests the negotiation module to construct a UDP tunnel to the correspondent node. The negotiation module also requests to NTMobile node's DC to construct a UDP tunnel. Finally, it also reports to the application about the completion of the tunnel construction process.

- NTMobile Network Socket API

  NTMobile network socket API is connected to an original Linux socket when the tunnel construction process is completed. The application starts communication over the UDP tunnel when they receive this information about completion of the tunnel construction process from the negotiation module.

### 3.3 Library Modules

The proposed library consists of three main modules for negotiation, handover, and packet manipulation. The following is the functions of each module.

- Negotiation Module

  The negotiation module serves as a signaling function for NTMobile communication. Therefore, application developers do not care about the detail process of NTMobile communication because the negotiation module can handle IP mobility and accessibility functions in NTMobile. The negotiation module registers the specific information for packet manipulation into the tunnel table.

- Handover Module

  The handover module should check the network interface status to detect a change of the access network. It also handles reconstruction of the UDP tunnel to the correspondent node when IP address changes due to an access network change.

- Packet Manipulation Module

  The packet manipulation module consists of two main functions: a virtual TCP/IP stack and a capsulation function. The virtual TCP/IP stack provides a transport layer function such as TCP and UDP to an application. The capsulation function handles the encapsulation and decapsulation for a UDP tunnel according to the information in the tunnel table.

### 3.4 Signaling Process

Figures 4, 5, 6 and 7 show the signaling processes in NTMobile. These figures are assumed that NTMobile node $\mathrm{MN}$ in a global IPv4 network requests to communication with NTMobile node $\mathrm{CN}$ in a private IPv4 network. Then, NTMobile node $\mathrm{MN}$ changes the network from the global IPv4 networks to a private IPv4 network. Direction coordinators $\mathrm{DC_{MN}}$ and $\mathrm{DC_{CN}}$ are the management servers for NTMobile node MN and CN respectively. The detail signaling process is as follows.

- Authentication

  Figure 4 shows the signaling processes for login in NTMobile. MN should request authentication to join the NTMobile network. First, MN connects AS to obtain the configuration for the authentication of MN. Then, MN requests authentication to AS with MN's account information and the obtained information from AS. After authentication, MN registers the information about the access network to $\mathrm{DC_{MN}}$ for IP mobility. Finally, MN registers the information of MN to $\mathrm{NS_{MN}}$ to receive the notification of the tunnel request from CN.

  1. The application calls the authentication API with the account information.
  2. The negotiation module on MN connects AS to obtain the configuration of MN when the configuration of NTMobile has not completed.
  3. AS replies the configuration information when the account information is valid.
  4. The negotiation module registers the obtained configuration information.
  5. The negotiation module requests authentication to AS by transmitting the login request message.
  6. AS generates a shared key for encryption between MN and $\mathrm{DC_{MN}}$ and distributes to $\mathrm{DC_{MN}}$ the shared key.
  7. $\mathrm{DC_{MN}}$ replies the acknowledgement message to AS.
  8. AS informs the shared key for encryption between MN and $\mathrm{DC_{MN}}$, and a FQDN of $\mathrm{DC_{MN}}$ by replying the login response message.
  9. The negotiation module registers the information about the access network to $\mathrm{DC_{MN}}$ by transmitting the registration request message.
  10. $\mathrm{DC_{MN}}$ requests $\mathrm{NS_{MN}}$ to provide a notification service using TCP connection for MN.
  11. $\mathrm{NS_{MN}}$ replies the acknowledgement message to $\mathrm{DC_{MN}}$ when it prepares the notification service for MN.
  12. $\mathrm{DC_{MN}}$ updates the network information of MN, and replies the registration response message.
  13. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  14. MN registers the information of MN to $\mathrm{NS_{MN}}$.
  15. $\mathrm{NS_{MN}}$ replies the acknowledgement message to MN.
  16. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  17. The negotiation module on MN returns the result of the authentication to the application.

- Tunnel Construction

  Figure 5 shows the signaling processes of tunnel construction in NTMobile. MN requests the tunnel construction to $\mathrm{DC_{MN}}$ to communicate to CN by transmitting the direction request message containing the FQDN of CN. Then, $\mathrm{DC_{MN}}$ finds $\mathrm{DC_{CN}}$ with the NS record of CN's FQDN. $\mathrm{DC_{MN}}$ accesses $\mathrm{DC_{CN}}$ to obtain the information about the access network of CN. $\mathrm{DC_{MN}}$ selects a tunnel route from the obtained information of CN. In this situation, $\mathrm{DC_{MN}}$ selects the direct communication route. Each DC indicates a tunnel construction processes to each NTMobile nodes. Finally, CN under the NAT router constructs the tunnel to MN.

  1. The application calls the name resolution API to construct a UDP tunnel to NTMobile CN.
  2. The negotiation module requests the tunnel construction to $\mathrm{DC_{MN}}$ by transmitting the direction request message. The direction request message contains the FQDN of CN.
  3. $\mathrm{DC_{MN}}$ makes inquiries about NS record for the FQDN of CN since $\mathrm{DC_{CN}}$ is the domain name server and manages the domain for the FQDN of CN.
  4. $\mathrm{DC_{MN}}$ generates a shared key for encryption between $\mathrm{DC_{MN}}$ and $\mathrm{DC_{CN}}$ and distributes to $\mathrm{DC_{CN}}$ the shared key.
  5. $\mathrm{DC_{CN}}$ replies the acknowledgement message to $\mathrm{DC_{MN}}$.
  6. $\mathrm{DC_{MN}}$ requests the information for CN to $\mathrm{DC_{CN}}$ by transmitting the NTM information request message.
  7. $\mathrm{DC_{CN}}$ replies the information about CN by replying the NTM information response message.
  8. $\mathrm{DC_{MN}}$ requests the tunnel construction to CN to $\mathrm{DC_{CN}}$ by transmitting the route direction message.
  9. $\mathrm{DC_{CN}}$ requests the notification to CN to $\mathrm{NS_{CN}}$.
  10. $\mathrm{NS_{CN}}$ sends the notification to CN using a type compatible with CN's device.
  11. CN replies the acknowledgement message to $\mathrm{DC_{CN}}$.
  12. $\mathrm{DC_{CN}}$ forwards the route direction message to CN.
  13. CN replies the acknowledgement message to $\mathrm{DC_{CN}}$.
  14. $\mathrm{DC_{CN}}$ also replies the acknowledgement message to $\mathrm{DC_{MN}}$.
  15. $\mathrm{DC_{MN}}$ indicates the tunnel construction to CN to MN by transmitting the route direction message.
  16. The negotiation module on MN replies the acknowledgement message to $\mathrm{DC_{MN}}$.

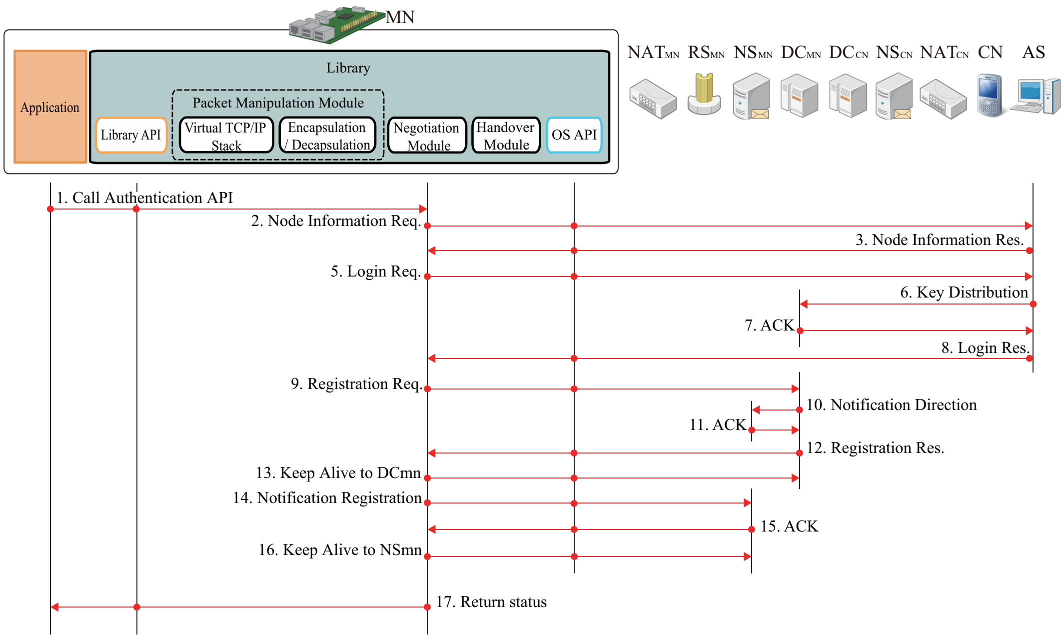



Figure 4: Signaling processes of the authentication

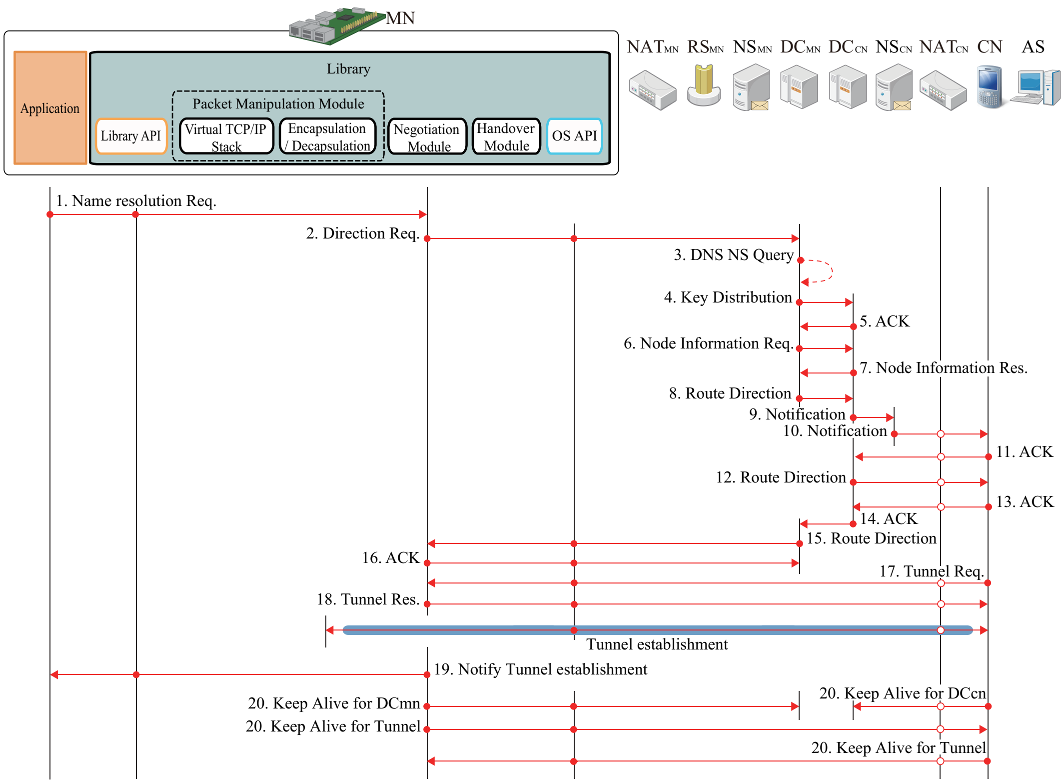



Figure 5: Signaling processes of the tunnel construction

17. CN transmits the tunnel request message to MN according to the direction from $DC_{MN}$ because MN has a global IP address in this case.

18. MN replies the tunnel response message to CN to complete the tunnel construction process.

19. The negotiation module notifies the completion of the tunnel construction process to the application.

20. The negotiation module transmits a keep alive message to keep the NAT table on the route.

21. The application starts the communication through the constructed UDP tunnel. Then, it transmits packets by using the special NTMobile socket.

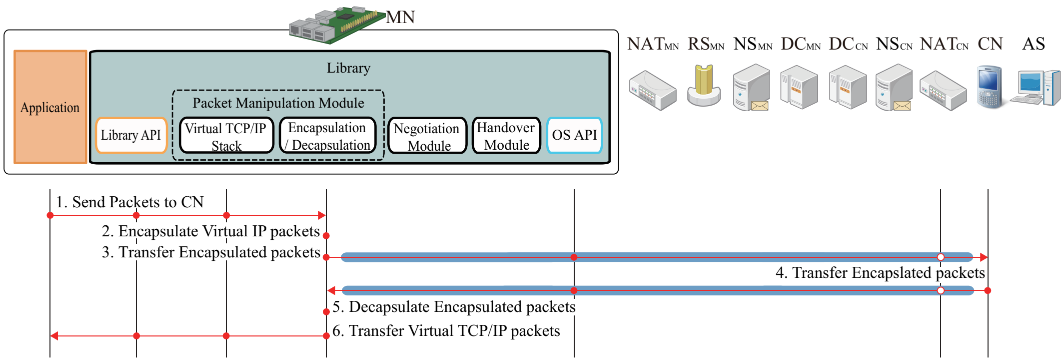



Figure 6: Signaling processes of the data communication

- Data Communication

  Figure 6 shows the signaling processes of data communication in NTMobile. The data communication between MN and CN is performed as encapsulated packet through the constructed tunnel. Therefore, the capsulation module performs the capsulation and decapsulation processes to conceal the change of the real IP address. The application uses the special socket API to perform the data communication.

  1. The virtual TCP/IP stack in the packet manipulation module serves the transport layer function to the application.
  2. The encapsulation and decapsulation module performs the encapsulation of virtual IP packets to CN.
  3. The encapsulated packet is transmitted to CN.
  4. The encapsulated packet is transmitted from CN.
  5. The encapsulation and decapsulation module performs the decapsulation the encapsulated packet from CN.
  6. The virtual TCP/IP stack in the packet manipulation module sends the decapsulated packet to the application.

- Change The MN Network

  Figure 7 shows the signaling processes when MN changes the access network. The handover module of NTMobile always observes the IP address of own network interface to detect the change of an access network. MN registers the information about the access network to $DC_{MN}$ when it detects the change of the access network. Then, the same tunnel construction procedures will be performed to reconstruct the tunnel. In this situation, $DC_{MN}$ selects the relay communication through RS. Each DC indicates the tunnel construction process to each NTMobile nodes. Finally, MN constructs the tunnel to CN through RS.

  1. Linux kernel can notify the change of network configuration through netlink socket.
  2. The handover module notifies the negotiation module that the access network of the device has changed.
  3. The negotiation module registers the new information about the access network to $DC_{MN}$ by transmitting the registration request message.
  4. $DC_{MN}$ requests $NS_{MN}$ to provide a notification service using TCP connection for MN.
  5. $NS_{MN}$ replies the acknowledgement message to MN.
  6. $DC_{MN}$ updates the network information of MN, and replies the registration response message.
  7. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  8. MN registers the information of MN to $NS_{MN}$.
  9. $NS_{MN}$ replies the acknowledgement message to MN.
  10. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  11. The negotiation module requests the tunnel construction to $DC_{MN}$ by transmitting the direction request message. The direction request message contains the FQDN of CN.
  12. $DC_{MN}$ makes inquiries about NS record for the FQDN of CN.
  13. $DC_{MN}$ generates a shared key for encryption between $DC_{MN}$ and $DC_{CN}$ and distributes to $DC_{CN}$ the shared key.
  14. $DC_{CN}$ replies the acknowledgement message to $DC_{MN}$.
  15. $DC_{MN}$ requests the information for CN to $DC_{CN}$ by transmitting the NTM information request message.
  16. $DC_{CN}$ replies the information about CN by replying the NTM information response message.
  17. $DC_{MN}$ requests RS to relay the communication between both NTMobile nodes.
  18. RS prepares the tunnel forwarding between both NTMobile nodes, and replies the acknowledgement message to MN.

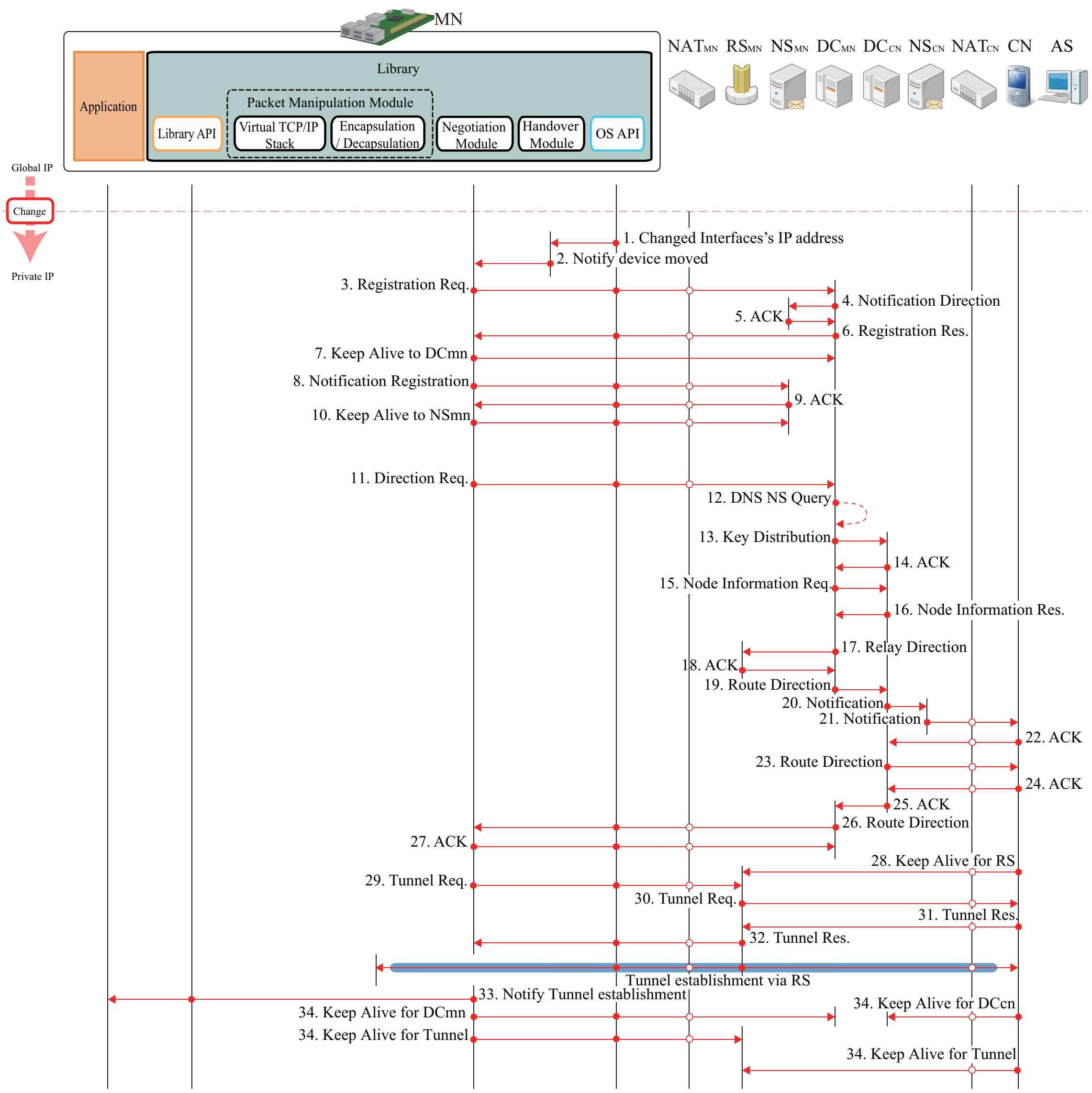



Figure 7: Signaling processes of the change the MN network

19. $DC_{MN}$ requests the tunnel construction toward CN to $DC_{CN}$ by transmitting the route direction message.
20. $DC_{CN}$ requests $NS_{CN}$ to transmit a notification to CN.
21. $NS_{CN}$ sends the notification to CN using a type compatible with CN's device in response to a request from $DC_{CN}$.
22. CN replies the acknowledgement message to $DC_{CN}$.
23. $DC_{CN}$ forwards the route direction message to CN.
24. CN replies the acknowledgement message to $DC_{CN}$.
25. $DC_{CN}$ also replies the acknowledgement message to $DC_{MN}$.
26. $DC_{MN}$ indicates the tunnel construction to CN to MN by transmitting the route direction message.
27. The negotiation module on MN replies the acknowledgement message to $DC_{MN}$.
28. CN transmits a keep alive message to keep the NAT table on the route.
29. MN transmits the tunnel request message to RS according to the direction from $DC_{MN}$ because both NTMobile nodes do not have a global IP address in this case.
30. RS transmits the tunnel request message to CN according to the direction from $DC_{MN}$.
31. CN replies the tunnel response message to RS to complete the tunnel construction process.
32. RS replies the tunnel response message to MN to complete the tunnel construction process.
33. The negotiation module notifies the completion of the tunnel construction process to the application.
34. The negotiation module transmits a keep alive message to keep the NAT table on the route.

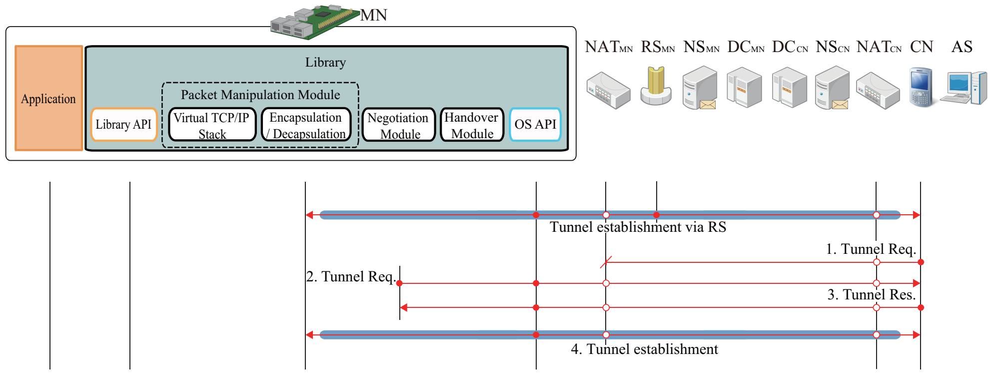



Figure 8: Signaling route optimization of library

- Route Optimization

  Figure 8 shows the signaling processes of route optimization in NTMobile. MN may be able to optimize the communication route by changing the route directly when it uses RS for the relay communication. Each NTMobile node transmits the tunnel request message to its own corresponding node to check the direct accessibility. They can change the route from the relay communication to the direct communication when either of NTMobile node can receive the tunnel request message from its own corresponding node. When neither NTMobile nodes can receive the tunnel request message, they use RS for the relay communication.

  1. CN transmits the tunnel request message to MN. In this case, the message cannot reach MN.
  2. MN transmits the tunnel request message to CN. In this case, the message can reach CN.
  3. CN replies the tunnel response message to MN to complete the tunnel construction process for the direct communication.
  4. Both NTMobile nodes change the tunnel route from the route through RS to the direct route.

- Tunnel communication in a same real private address

  Figure 9 shows the tunnel communication where a same real private IP address RIP is assigned to both MN and CN. In this situation, MN and CN use virtual IP addresses $\mathrm{VIP}_{\mathrm{MN}}$ and $\mathrm{VIP}_{\mathrm{CN}}$ respectively. Therefore, the application also applies these virtual IP addresses for communication. In the tunnel communication, the IP packet including these virtual IP addresses is encapsulated in the packet manipulation module. As a result, the packet manipulation module changes the IP address field from these virtual IP addresses to the real IP addresses RIP and $\mathrm{RIP}_{\mathrm{RS}}$ The source address changes from RIP to $\mathrm{RIP}_{\mathrm{NATMN}}$ through $\mathrm{NAT}_{\mathrm{MN}}$. RS transfer the received packet from MN. Hence, the addresses changes from the pair of $\mathrm{RIP}_{\mathrm{NATMN}}$ and $\mathrm{RIP}_{\mathrm{RS}}$ to the pair of $\mathrm{RIP}_{\mathrm{RS}}$ and $\mathrm{RIP}_{\mathrm{NATCN}}$. The destination address changes from $\mathrm{RIP}_{\mathrm{NATCN}}$ to RIP through $\mathrm{NAT}_{\mathrm{CN}}$. Finally, CN receives this packet from MN through RS.

- Tunnel communication with cascading NATs

  Figure 10 shows the tunnel communication where a same real private IP address RIP is assigned to both MN and CN and CN exists under the cascading NAT routers. In the tunnel communication, the IP packet including virtual IP addresses is encapsulated in the packet manipulation module. As a result, the packet manipulation module changes the IP address field from these virtual IP addresses to the real IP addresses in the similar manner in Fig. 9. Finally, CN receives this packet from MN through RS.

# 4 EVALUATION

## 4.1 Processing Overhead

We have measured the processing overhead in a NTMobile network. We have prepared the evaluation environment in Fig. 11. Table 1 shows the detail information about each component.

In the evaluation, we have measured each period for the NTMobile signaling. The results represent an average of 100 trials. In the measurement, we used ping tool to measure round trip time (RTT) and NSLOOKUP tool to measure a name resolution period for an AAAA record. Tables 2 and 3 show the RTT for each link. We can find that the RTT between servers is around 0.1[ms] and RTT between a server and a NTMobile node is around 1[ms]. The difference comes from the hardware performance.

Table 4 shows the processing period for each NTMobile signaling. The results show that the developed implementation requires about 200[ms] for authentication and a registration. The authentication is required when a NTMobile node joins a NTMobile network. Therefore, 200[ms] are acceptable time for the login process. The registration is required when a NTMobile node changes an access network. It is also acceptable for updating an own network information. The tunnel construction period is an important for a usability. From the results, we can find that the tunnel construction period is less that 200[ms]. It is a known fact that TCP connection setup requires a three-way handshake. Being compared to the period for the three-way handshake, the tunnel construction period is also acceptable.

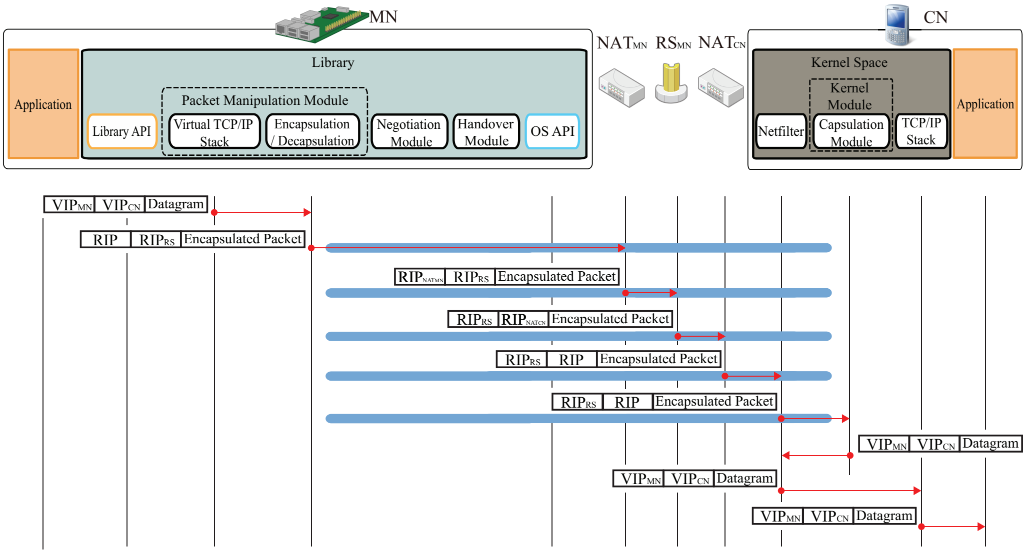



Figure 9: Tunnel communication in a same real private address

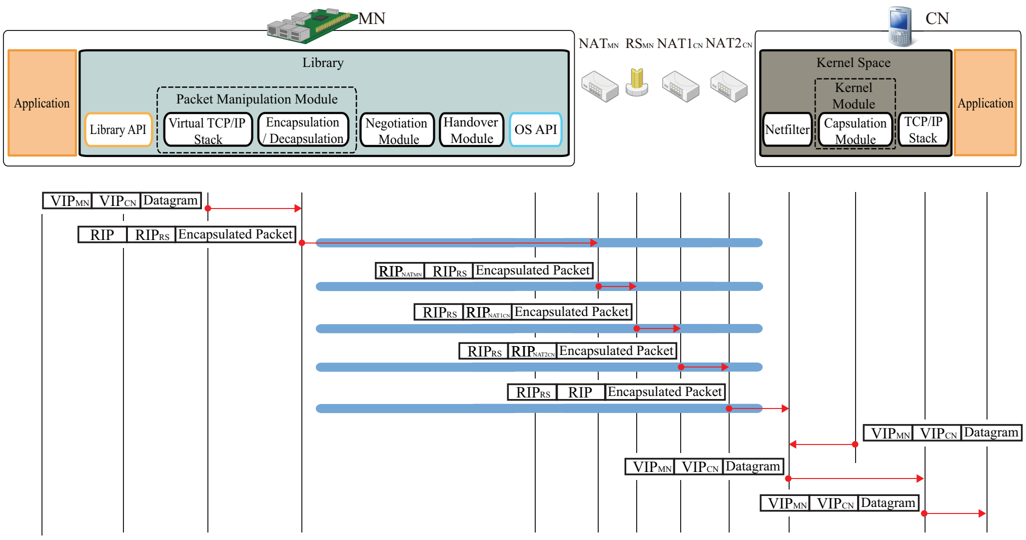



Figure 10: Tunnel communication with cascading NATs

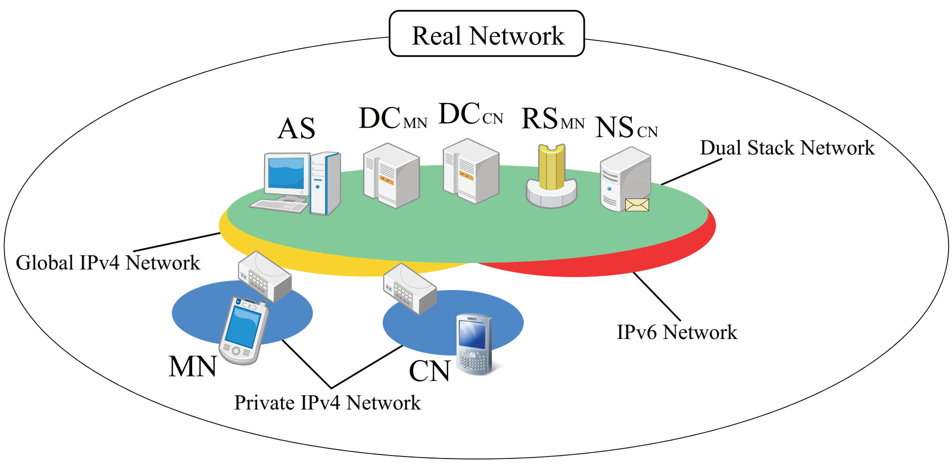



Figure 11: Evaluation environment

Table 1: Machines specification

| | AS, $DC_{MN}$, $DC_{CN}$, $RS_{MN}$, $NS_{CN}$ | MN, CN |
|---|---|---|
| OS | CentOS 6.7 | Ubuntu 12.04 |
| Kernel | Linux 2.6 | Linux 3.2 |
| CPU | Intel Xeon 2.8GHz | Virtual 1core 1.8GHz |
| Memory | 512MB | 512MB |

Table 4: Processes period

| | Delay time [ms] | | |
|---|---|---|---|
| Process | min | avg | max |
| Authentication | 152.0 | 179.1 | 226.6 |
| Registration & Notification Registration (without NS) | 57.4 | 64.7 | 71.3 |
| Registration & Notification Registration (with NS) | 172.3 | 194.6 | 233.5 |
| Notification Registration - ACK (TCP) | 48.9 | 55.1 | 63.4 |
| Tunnel Construction (without NS) | 137 | 169 | 191 |
| Direction Request - Route Direction | 112.0 | 134.6 | 151.8 |
| Tunnel Request - Tunnel Response | 13.8 | 29.3 | 45.5 |
| Tunnel Construction (with NS) | 170 | 197 | 226 |
| Direction Request - Route Direction | 133.5 | 154.3 | 174.2 |
| Tunnel Request - Tunnel Response | 13.8 | 29.3 | 45.5 |

Table 2: RTT between servers

| | RTT [ms] | | |
|---|---|---|---|
| Connection pair | min | avg | max |
| AS - $DC_{MN}$ | 0.117 | 0.140 | 0.170 |
| AS - $DC_{CN}$ | 0.110 | 0.121 | 0.174 |
| $DC_{MN}$ - $RS_{MN}$ | 0.128 | 0.148 | 0.173 |
| $DC_{CN}$ - $NS_{CN}$ | 0.133 | 0.155 | 0.177 |

Table 3: RTT between servers and nodes

| | RTT [ms] | | |
|---|---|---|---|
| Connection pair | min | avg | max |
| AS - MN | 0.99 | 1.45 | 1.87 |
| AS - CN | 0.91 | 1.39 | 2.20 |
| $DC_{MN}$ - MN | 0.91 | 1.38 | 1.97 |
| $DC_{CN}$ - CN | 0.97 | 1.42 | 2.26 |
| $RS_{MN}$ - MN | 0.91 | 1.39 | 1.81 |
| $RS_{MN}$ - CN | 1.00 | 1.38 | 2.22 |
| $NS_{CN}$ - CN | 0.92 | 1.35 | 3.04 |

## 4.2 Processing Transmission Delay

We have measured transmission delays in real IP networks to evaluate the overhead due to transmission delay in practical networks. Figure 12 shows the definition of each transmission delay in NTMobile network.

Table 5 shows the number of each transmission delay to construct a communication tunnel. Therefore, the total overhead due to the transmission delay is the summation of each delay. Tables 6 and 7 show the measurement results of transmission delay on ping tool in real wired networks and in LTE networks. The results are the average of 1,000 trials and 100 trials respectively. Table 8 shows the query period for the DNS mechanism with NSLOOKUP tool. From the evaluations in Tabs. 5, 6 and 7, the estimated tunnel construction period is 491 [ms] for direct communication and 689 [ms] for relay communication. The main factor of the overhead comes from the transmission delay in LTE networks. We think that the overhead can be acceptable in practical situations because it occurs only when users try to start a communication.

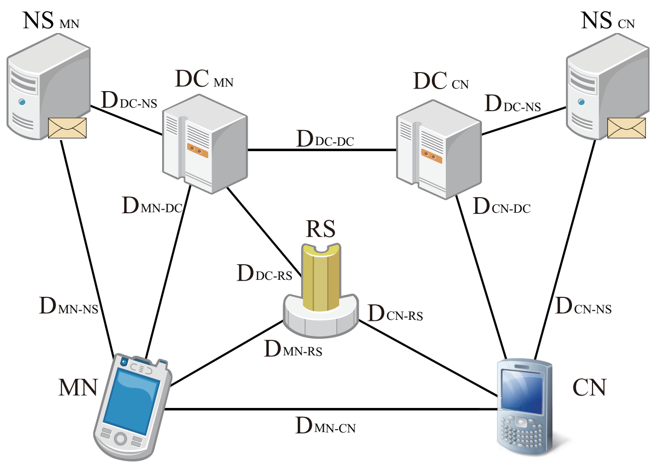



Figure 12: Transmission delay in NTMobile network

## 5 CONCLUSION

This paper has extend the IP mobility mechanism called NTMobile (Network Traversal with Mobility) to designed an application based IP mobility scheme on Linux systems, where the developed IP mobility library can realize the IP mobility function in an application layer on Linux systems. As a result, developers can realize an end-to-end communication model by employing the enhanced IP mobility library.

Table 5: Number of delay path for tunnel construction process

| Connection path | Route name | Times |
|---|---|---|
| Direct | $D_{MN-DC}$ | 2 |
| | DNS | 1 |
| | $D_{DC-DC}$ | 6 |
| | $D_{DC-NS}$ | 1 |
| | $D_{CN-NS}$ | 1 |
| | $D_{CN-DC}$ | 3 |
| | $D_{MN-CN}$ | 1 |
| Relay | $D_{MN-DC}$ | 2 |
| | DNS | 1 |
| | $D_{DC-DC}$ | 6 |
| | $D_{DC-RS}$ | 2 |
| | $D_{DC-NS}$ | 1 |
| | $D_{CN-NS}$ | 1 |
| | $D_{CN-DC}$ | 3 |
| | $D_{MN-RS}$ | 2 |
| | $D_{CN-RS}$ | 3 |

Table 6: Transmission delay in wired networks

| Target host | Trial | Average time [msec] |
|---|---|---|
| google.com | 1,000 | 6.222 |
| yahoo.co.jp | 1,000 | 7.3215 |

Table 7: Transmission delay in LTE networks

| Target host | Trial | Average time [msec] |
|---|---|---|
| google.com | 100 | 59.365 |
| yahoo.co.jp | 100 | 63.79 |

Table 8: Query period of DNS lookup

| Target domain | Trial | Average time [msec] |
|---|---|---|
| aitech.ac.jp | 100 | 12.54 |
| yahoo.co.jp | 100 | 12.9 |

## ACKNOWLEDGEMENT

This work is supported in part by the Grant-in-Aid for Scientific Research (26330103, 15H02697), Japan Society for the Promotion of Science (JSPS) and the Integration research for agriculture and interdisciplinary fields, Ministry of Agriculture, Forestry and Fisheries, Japan, and the Science Research Promotion Fund from The Promotion and Mutual Aid Corporation for Private Schools of Japan.

## REFERENCES

[1] M. Buddhikot, G. Chandranmenon, S. Han, Y. W. Lee, S. Miller, and L. Salgarelli, "Integration of 802.11 and third-generation wireless data networks," Proceedings of the IEEE INFOCOM 2003, Volume 1, pp. 503 - 512, March/April (2003).

[2] Q. Zhang, C. Guo, Z. Guo, and W. Zhu, "Efficient mobility management for vertical handoff between WWAN and WLAN," IEEE Communications Magazine, Volume 41, No. 11, pp. 102 - 108, November (2003).

[3] L. A. Magagula and H. A. Chan, "IEEE802.21-Assisted Cross-Layer Design and PMIPv6 Mobility Management Framework for Next Generation Wireless Networks," Proc. IEEE WIMOB '08, pp. 159 - 164, October (2008).

[4] D. Le, X. Fu and D. Hogrere, "A Review of Mobility Support Paradigms for the Internet," IEEE Communications surveys, 1st quarter 2006, Volume 8, No. 1, pp. 38 - 51, First Quarter (2006).

[5] A. C. Snoeren and H. Balakrishnan, "An End-to-End Approach to Host Mobility," Proceedings of the ACM Mobicom, pp. 155 - 166, August (2000).

[6] C. Perkins, "IP Mobility Support for IPv4, Revised," IETF RFC 5944, November (2010).

[7] M. Ishiyama, M. Kunishi, K. Uehara, H. Esaki, and F. Teraoka, "LINA: A New Approach to Mobility Support in Wide Area Networks," IEICE transactions on communications, Volume E84-B, No.8, pp. 2076 - 2086, August (2001).

[8] http://www.mip4.org, retrieved: February (2012).

[9] C. Perkins and D. Johnson, "Route Optimization in Mobile IP," draft-ietf-mobileip-optim-11.txt, Work in progress, September (2001).

[10] S. Salsano, C. Mingardi, S. Niccolini, A. Polidoro, and L. Veltri, "SIP-based Mobility Management in Next Generation Networks," IEEE Wireless Communication, Volume 15, Issue 2, pp. 92 - 99, April (2008).

[11] M. Bonola, S. Salsano, and A. Polidoro, "UPMT: universal per-application mobility management using tunnels," In Proc. of the 28th IEEE conference on Global telecommunications (GLOBECOM'09), pp. 1 - 8, November/December (2009).

[12] M. Bonola and S. Salsano, "S-UPMT: a secure Vertical Handover solution based on IP in UDP tunneling and IPsec," GTTI Riunione Annuale 2010, (online), http://www.gtti.it/GTTI10/papers/gtti10_submission_29.pdf, June (2010).

[13] H. Soliman, "Mobile IPv6 Support for Dual Stack Hosts and Routers," IETF RFC 5555, June (2009).

[14] E. Fogelstroem, A. Jonsson, and C. Perkins, "Mobile IPv4 Regional Registration," IETF RFC 4857, June (2007).

[15] H. Levkowetz and S. Vaarala, "Mobile IP Traversal of Network Address Translation (NAT) Devices," IETF RFC 3519, April (2003).

[16] G. Montenegro, "Reverse Tunneling for Mobile IP, revised," IETF RFC 3024, January (2001).

[17] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session Traversal Utilities for NAT (STUN)," IETF RFC 5389, October (2008).

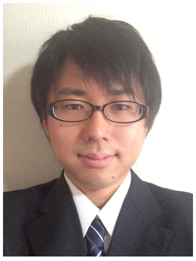
[18] P. Srisuresh, B. Ford, and D. Kegel, "State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)," IETF RFC 5128, March (2008).

[19] R. Mahy, P. Matthews, and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," IETF RFC 5766, April (2010).

[20] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols," IETF

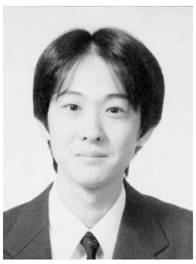
RFC 5245, April (2010).
[21] M. Westerlund and C. Perkins, "IANA Registry for Interactive Connectivity Establishment (ICE) Options," IETF RFC 6336, July (2011).
[22] K. Naito, K. Kamienoo, T. Nishio, H. Suzuki, A. Watanabe, K. Mori, and H. Kobayashi, "Proposal of Seamless IP Mobility Schemes: Network Traversal with Mobility (NTMobile)," IEEE GLOBECOM 2012, pp. 2572 - 2577, December (2012).
[23] K. Naito, K. Kamienoo, H. Suzuki, A. Watanabe, K. Mori, and H. Kobayashi, "End-to-end IP mobility platform in application layer for iOS and Android OS," IEEE Consumer Communications & Networking Conference (CCNC 2014), pp. 92 - 97, January (2014).
[24] http://www.ntmobile.net
[25] lwip - A Lightweight TCP/IP stack, http://www.nongnu.org/lwip/

(Received October 30, 2015)
(Revised February 7, 2016)



**Kohei Tanaka** is currently working towards the B.S. degree in the Faculty of Information Science at Aichi Institute of Technology, Japan. His research interests include network mobility systems and Internet of Things (IoT).

**Fumihito Sugihara** received the B.S. degree in Electrical and Electronic Engineering from Mie University, Japan in 2014. He is currently working towards the M.S. degree in the Department of Electrical and Electronic Engineering at Mie University, Japan. His research interests include network mobility systems and Machine to Machine (M2M).



**Katsuhiro Naito** received the B.S. degree in Electronics Engineering from Keio University, Japan in 1999, and received the M.S. and Ph.D. degrees in Information Engineering from Nagoya University, Japan in 2001 and 2004, respectively. From 2004 to 2014, Dr. Naito was an assistant professor in the electrical and electronic engineering department of Mie university. He was a visiting scholar in the computer science department of University of California, Los Angeles (UCLA) in 2011. Since 2014, he has been an associate professor in the information science department of Aichi Institute of Technology. His research interests include 5G technologies, vehicular communication systems, Internet of Things (IoT) and Machine to Machine (M2M) systems, overlay networks, and network protocols and architectures.

**Hidekazu Suzuki** received his B.S., M.E., and Ph.D. degrees in Information Engineering from Meijo University, Japan in 2004, 2006 and 2009, respectively. From 2008 to 2010, he was a Postdoctoral Research Fellow of the Japan Society for the Promotion of Science. He was an Assistant Professor in the Department of Information Engineering in Meijo University from 2010 to 2015. He has been an Associate Professor in the same department from 2015. His research interests are in the fields of network security, mobile networks and home networks, and so on. He is also a member of IEEE, ACM, IEICE and IPSJ.

**Akira Watanabe** received his B.E. and M.E. degrees from Keio University in 1974 and 1976, respectively. He joined Mitsubishi Electric Corporation in 1976, and engaged in R&D of network devices. He received the Ph.D. degree in communication systems from Keio University in 2001. Since joining Meijo University in 2002, he has been researching on network connectivity and mobility.

# A Road Information Sharing Scheme with a Still-Picture Internet Broadcasting System

Yoshia Saito* and Yuko Murayama**

*Faculty of Software and Information Science, Iwate Prefectural University, Japan
y-saito@iwate-pu.ac.jp
**Department of Computer Science, Tsuda College, Japan
murayama@tsuda.ca.jp

***Abstract*** – With an increase in navigation systems by portable terminals and car navigation devices, ITS which supports comfortable and effective driving has been evolved. However, in existing systems, it is difficult to offer real-time information to users since it needs to collect information from sensors and to analyze the collected information. Road information has to be provided timely so that drivers can pass through safe and comfortable roads. In this research, we use a still-picture Internet broadcasting system as a technique to share road information. It enables users to share the road information timely and to choose a road which is easy to pass. In addition, we implement an automatic photography function and conduct an experiment of the photography timing as a broadcaster. In this paper, we describe the design and implementation of our proposed system and evaluation experiments about the right photography timing of the automatic photography function. From the result of the experiments, we found road information was required more in the bad road situation such as rain and snow, and the road information could be grasped easily by introducing audience requests to the photography timing.

***Keywords***: Road Information Sharing, Still-Picture Internet Broadcasting, Photography Timing

## 1 INTRODUCTION

ITS (Intelligent Transport System) technology which addresses traffic information has been developed in recent years with the introduction of recent information technology to deal with increased volume of traffic. The road situation changes from moment to moment due to change of road surface conditions, weather conditions, traffic volume and various factors. The road information is an important factor and has high demand for drivers.

VICS (Vehicle Information and Communication System) [1] is one of systems to get road information in Japan. VICS provides road information which is collected by an information center via communication and broadcasting media such as FM multiplex broadcasting. Drivers can receive the road information by their car navigation systems and utilize it to select an appropriate route to the destination. However, VICS takes time to provide the road information to the drivers because the center needs to collect and analyze information. In addition, the ways of providing road information is limited to text, audio and map display. It is difficult for the drivers to understand the detailed road situation. The road information should be timely and provided in an easy-to-understand way.

Meanwhile we have studied a still-picture Internet broadcasting system [2]-[4] which uses still-pictures and audio streaming instead of video streaming to realize practical broadcasting via row-speed or limited high-speed cellular network by reducing data traffic. This system can broadcast anywhere using smartphones even if only connected to row-speed network. To provide timely and easy-to-understand road information, we propose a road information sharing scheme based on the still-picture Internet broadcasting system. The proposed system provide road information to users who want to know the road situation for route selection using still-pictures and audio streaming in real-time. To provide still-pictures and audio streaming, cooperative drivers set their smartphones on their cars. The broadcasting system works on the smartphones and automatically takes still-pictures and sends the still-pictures to the broadcasting server at the right time not to disrupt their driving operation. Users can select a car and view the broadcasting to get the road information in a certain area. The users can also communicate with the driver and get more detailed road information by the drivers through the communication.

In this paper, at first, we explain features and issues of existing traffic information systems. Secondly, we describe detail of the road information sharing scheme with a still-picture Internet broadcasting system. Then, preliminary experiment is conducted using the implemented prototype system to study appropriate timing to take still-pictures. At last, we introduce several photography timing algorithms to the prototype system and evaluate the algorithms.

## 2 ISSUES OF EXISTING SYSTEMS AND OUR APPROACH

There are probe vehicle systems [5] to get road information. The probe vehicle system collects wide-area road information from probe vehicles which have various sensors and reduces cost for sensor installation. One of the services of the probe vehicle system, there is a vehicle tracking map which is provided by Honda [6]. This service shows whether the road is travelable or not on the map based on collected information from probe vehicles and aims to support driving in disaster area. It was used in 2007's the Niigataken Chuetsu-oki Earthquake and 2011's the Great East Japan Earthquake. This fact means road information is in great demand.

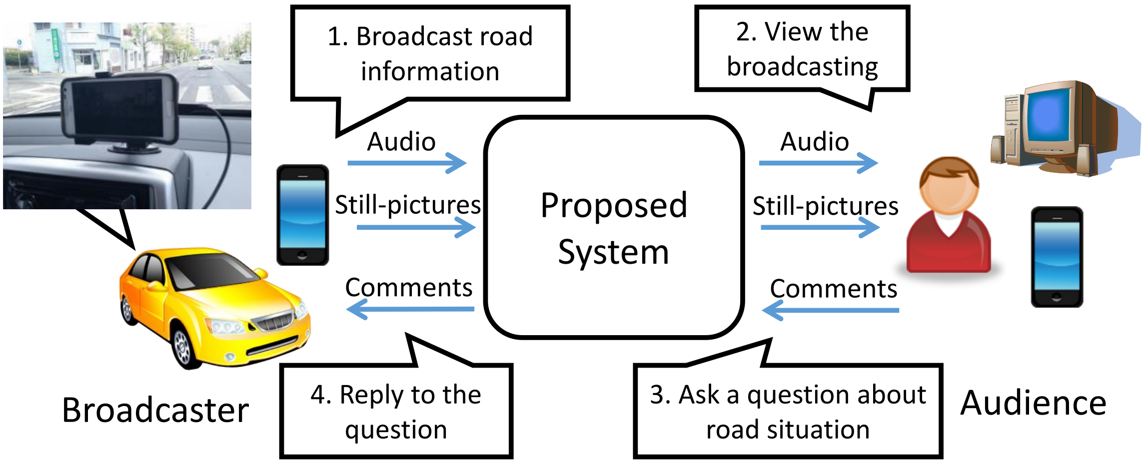



Figure 1: The proposed system model

There are several issues in the probe vehicle system. At first, the road information from probe vehicle system lacks timeliness. For example, a service provider checks road information to avoid traffic jams in advance but the road could be backed up when the user arrived. This is because the probe vehicle system takes time to collect and analyze the road information from probe vehicles and the information lacks timeliness. Secondly, the road information is not flexible and intuitive. Typical road information which is provided by the probe vehicle system and VICS is predetermined by service providers and shown as text and icons. The users cannot know road situation in detail. Thirdly, the probe vehicle system requires dedicated sensor devices such as a specific car navigation system and it is difficult to prepare many probe vehicles. To get wide-are road information in real-time, the dedicated sensor devices should be eliminated. From these issues, it is important to provide flexible road information timely in an easy-to-understand way without dedicated sensor devices.

We focused on drive broadcasting which is one of the broadcasting styles to show driving landscape using in-vehicle camera and communication devices. The audience enjoys the driving landscape and communicating with the broadcaster. The drive broadcasting is a popular content in live streaming services such as Ustream [7] and NicoNico Live [8]. The drive broadcasting can share road information timely. However, it has an issue about network communication. In the drive broadcasting, 3G/4G cellular network devices are used generally. Although 3G cellular network covers wide-area, it is too low-speed for video streaming. The video can be frequently stopped and low-quality. While 4G cellular network provides enough network bandwidth, it usually has limitation of amount of data traffic per day and month. Cellular carriers in Japan make communication speed slow when the subscriber uses hundreds of megabytes in a day or several gigabytes in a month. Therefore, the data traffic should be reduced for drive broadcasting.

## 3 PROPOSED SYSTEM

We have studied a still-picture Internet broadcasting system using smartphones which uses still-pictures and audio streaming instead of video streaming to reduce data traffic. Even if only 3G cellular network is available, the system realizes stable broadcasting. However, the previous study did not specify the use cases. In this research, we use the still-picture Internet broadcasting system for drive broadcasting to share road information.

The proposed system provides road information in real-time by live still-picture broadcasting so that users can choose safe and comfortable roads. The live still-picture broadcasting enables the users to understand road situation intuitively and also realize stable broadcasting anywhere by reducing data traffic. The users also can communicate with any broadcasters and ask a question about road situation.

Figure 1 shows the proposed system model. The broadcaster sets a smartphone on his/her car. The smartphone sends still-pictures and audio stream to the proposed system in order to share road information. It is on the assumption that the still-pictures are taken and sent automatically at the right timing. The broadcast programs are shown on a map.

Audience can select a broadcast program on the map and view the broadcasting by using their PCs or smartphones. The PCs/smartphones receive the still-pictures and audio stream from the proposed system. The audience can interactively ask a question about road situation to the broadcaster through the proposed system and the broadcaster can reply to the question to complement the road situation which is not understood by the still-pictures and audio stream.

The proposed system realizes timely road information sharing between broadcasters and audience and helps audience understand the road situation by still-pictures and interactive communication with broadcasters. Because it also does not require dedicated sensor devices and just uses smartphones, anyone can share road information and wide-area road information can be covered when there are a lot of broadcasters.

The use case of the proposed system is as follows. The broadcasters are people who drive for commuting or trip. The motivation of the broadcasters is to enjoy communicating with audience like fellow passengers. The audience are people who have a plan to go to the broadcasting place for commuting or trip and want to know

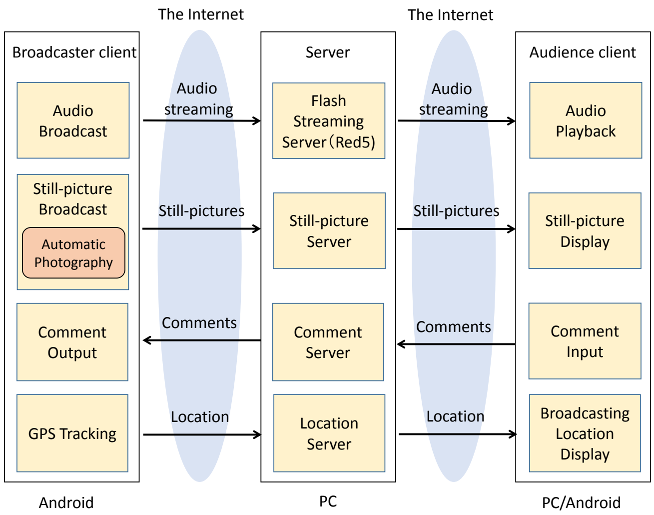



Figure 2: The system architecture

the road situation. The audience can confirm traffic, road surface, and weather and so on by viewing the broadcasting to select routes.

## 4 PROTOTYPE SYSTEM

We implemented a prototype system based on the system model to conduct a preliminary experiment. A client software for broadcasters was developed on a smartphone and two client software for audience were developed on a smartphone and PC. Android smartphones were used for the client development.

Figure 2 shows the system architecture. At first, a broadcaster launches a broadcaster client on the smartphone and starts broadcasting. The audio broadcast function on the broadcaster client sends audio stream to a server using RTMP. The still-picture broadcast function on the broadcaster client sends still pictures which are encoded by JPEG2000 to the server. Since the broadcaster cannot touch the smartphone in driving, the automatic photography function takes still-pictures at right timing automatically. In this implementation, the timing is fixed time interval but it is variable.

On the server, Red5 which is a flash streaming server receives the audio stream from the broadcaster client. When an audience client connects to the server and select a broadcasting, the Red5 sends the audio stream to the connected audience clients. The audience client receives and plays the audio stream. The still-pictures are also sent to the connected audience clients through a still-picture server which is implemented in Java. The audience client receives and display the latest still-picture.

The audience client can send text comments to the broadcaster through a comment server which is implemented in Java. The broadcaster client display the comments and read out the comments so that the broadcaster can communicate with audience without watching the smartphone in driving. The broadcaster client does not have the comment input function because the broadcaster cannot touch the

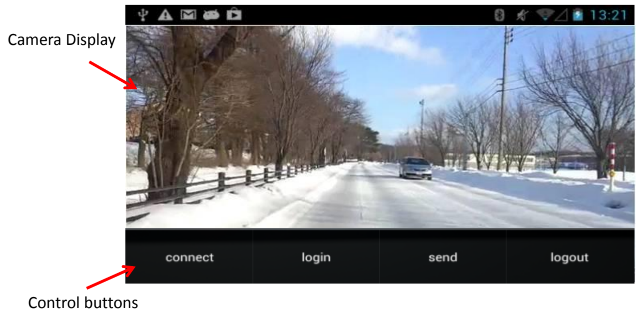



Figure 3: The user interface for the broadcaster client

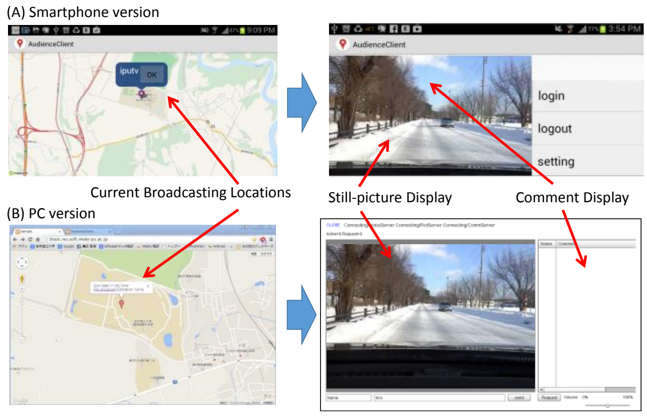



Figure 4: The user interfaces for the audience clients (A: smartphone version, B: PC version)

smartphone in driving. The broadcaster hears the comments from audience and speaks about the reply to them.

The location of the broadcaster client is tracked by GPS on the smartphone and sent to the server. The server associates the location with the broadcasting and stores the location information in real-time. The audience can view the list of broadcastings on the map and select a broadcasting which they want to watch.

Figure 3 shows the user interface for the broadcaster client. The real-time camera image is displayed on the center. On the bottom part, there are control buttons. The connect button is used for connecting to the server. The login button starts to send audio stream and still-pictures to the server. The send button is used for manually sending still-picture to the server. The logout button stops the broadcasting. The comments from audience are displayed over the camera image and read out.

Figure 4 shows the user interfaces for the audience clients. The upside is smartphone version and the downside is PC version. The smartphone version is developed as an Android application and the PC version as a Web application. On start-up, the both audience clients show current broadcasting points on the map. When a broadcasting point is selected, the user interface is changed and the correspondent broadcasting is started. The audience can input comments. The inputted comments are displayed over the still-picture on the smartphone version and the comment display field on the PC version.

Table 1: Feedbacks in the preliminary experiment

| Categories | Feedbacks | Countermeasures |
|---|---|---|
| Broadcasting information | * I would like to know where the still-pictures were taken.<br>* I started watching from the middle of the broadcast. I didn't understand where the broadcaster came from. | * The location information should be added to the still-picutres.<br>* The history of the still-pictures should be given to the audience. |
| Automatic photography timing | * I would like to shorten the interval of the automatic photography.<br>* I do not need so many still-pictures because the still-pictures are similar ones.<br>* During night-time, many rayless still-pictures were displayed and I did not understand the road situation when there were not streetlights. | * The automatic photography timing should be not fixed and more flexible.<br>* The photography timing sould be determined when the audience can understand the road situation from the still-picture |

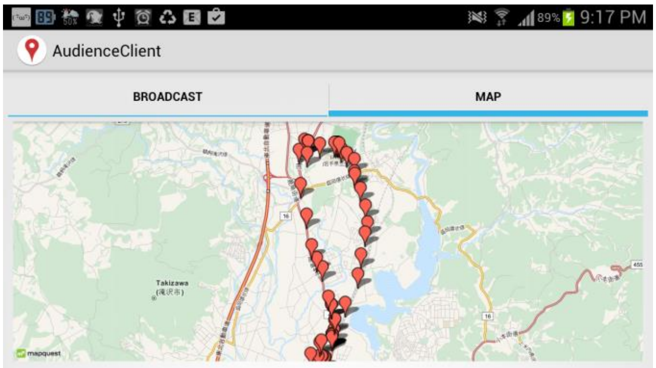



Figure 5: Location and history of still-pictures

# 5 PRELIMINARY EXPERIMENT

To check operation of the prototype system and find its issues, we conducted a preliminary experiment using a real car. The experimental period was from May 5th to July 4th in 2014 and the broadcasting was performed for 30 minutes in twice during daylight hours and night-time hours in a day. The subjects were 14 students of Iwate Prefectural University. They viewed the broadcasting using the audience client of PC version. The interval of the automatic photography was set to 30 seconds per a still-picture and the resolution of the still-pictures was 320x240.

## 5.1 Feedbacks and Countermeasures

From the experiment, we got several feedbacks from the subjects. Table 2 shows the feedbacks. The feedbacks can be categorized by two groups. The first group of feedbacks is about “broadcasting information.” The feedbacks were “I would like to know where the still-pictures were token” and “I started watching from the middle of the broadcast. I didn’t understand where the broadcaster came from.” These feedbacks point out the lack of information about still-pictures. Since the prototype system only showed current broadcasting points at the start, the audience could not understand location of the still-pictures and prior still-pictures on the route. To solve the problems, the location information should be added to the still-pictures and the history of the still-pictures should be given to the audience. We added these functions to the prototype system. Figure 5 shows the implemented function for location and history of still-pictures. The audience can confirm the location and history of still-pictures shown as pins on the map and the still-pictures are displayed by clicking on the pins.

The second group of feedbacks is about “automatic photography timing.” Many subjects told about the automatic photography timing. The feedbacks were “I would like to shorten the interval of the automatic photography” and “I do not need so many still-pictures because the still-pictures are similar ones.” Furthermore, some subjects told about what still-pictures were required to help users understand the road situation. For example, the typical feedback was “During night-time, many rayless still-pictures were displayed and I did not understand the road situation when there were not streetlights.” The still-pictures have higher demand in well-lighted area than in dark area. These results show the automatic photography timing should not be fixed and more flexible. If the timing was a fixed interval, poor demand still-pictures could be sent to audience. To enable audience to understand road situation more easily, the photography timing should be determined when the audience can understand the road situation from the still-picture.

## 5.2 Discussion

Since there were strong demand for improvement of the automatic photography timing, we focus on developing new photography timing algorithm in this paper. To improve the photography timing, we make two hypotheses about the photography timing. The first hypothesis is that the demand of road information is changed depending on the road situation. For example, the demand of road information can be higher in a good weather condition than in a bad weather condition, and higher in a congested road than in a no traffic road. The second hypothesis is that the demand of road information is different from person to person. In the experiment, we got different feedbacks about the photography timing even if they watched same broadcasting. These are also mentioned in a related work. Münter [9] found drivers need more support when they don’t have spatial knowledge and sense of direction of the person, and

Table 2: Utilization of road information in similar services

| Service Name | Vehicle Speed | Sudden Braking | Road Surface | Traffic | Weather |
|---|---|---|---|---|---|
| SAFETY MAP | | ○ | | | |
| EuroRAP | ○ | | ○ | | |
| SafeRoadMaps | | | ○ | ○ | ○ |
| Mi Drive | ○ | | | | ○ |

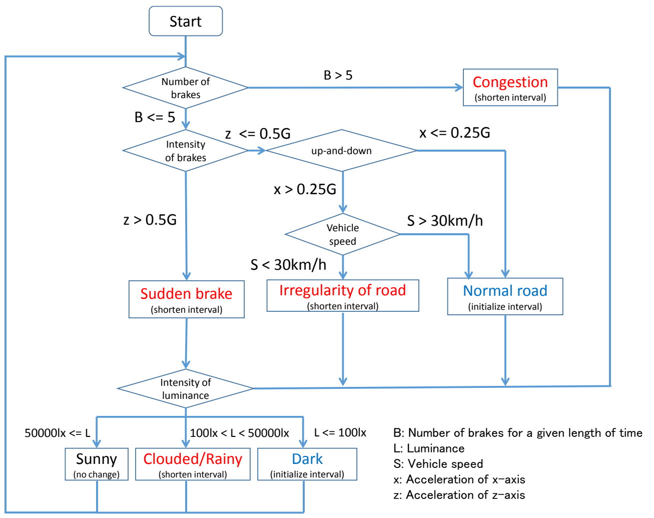



Figure 6: the photography timing algorithm based on road situation

weather condition is bad. An effective photography timing algorithm needs to be studied based on the hypotheses to provide high demand still-pictures to the audience.

## 6 PHOTOGRAPHY TIMING ALGORITHM

We developed two photography timing algorithms to verify the hypotheses. The first algorithm changes the photography timing based on road situation utilizing sensors of a smartphone. This algorithm takes into account the first hypothesis. The second algorithm changes the photography timing based on audience request in addition to road situation. This algorithm takes into account the second hypothesis.

To develop the first algorithm, we researched similar road information services and what types of road information is utilized. Table 2 shows the result. The SAFETY MAP [10] which is provided by Honda uses sudden braking information for detecting unsafe points. The EuroRAP (European Road Assessment Programme) [11] which aims to reduce death and serious injury uses vehicle speed and road surface information. SafeRoadMaps [12] which is developed by University of Minnesota and Claremont Graduate University uses road surface, traffic and weather information for safety alerts. Mi Drive [13] which is provided by Michigan Department of Transportation (MDOT) uses vehicle speed and weather information for safety information. From these results, the first algorithm collects vehicle speed, sudden braking, road surface, traffic, and weather information by smartphone sensors and changes the photography timing based on these information.

Figure 6 shows the photography timing algorithm based on road situation. The algorithm starts operation when the driver puts on the brake. If the number of brakes for a given length of time is greater than 5, the algorithm shorten the interval of photography timing because a lot of brakes mean current road is congested [14]. If the number of brakes is below 5, it checks intensity of the brake to detect a sudden brake. The interval of photography timing is shortened when acceleration of z-axis which is anteroposterior acceleration of the vehicle is greater than 0.5G. If the acceleration of z-axis is below 0.5G, the algorithm checks acceleration of x-axis which is vertical acceleration of the vehicle and its speed to detect irregularity of road [15]. If the road is bumpy, the algorithm shortens the interval of photography timing. Otherwise, the interval will be initialized. After that, the algorithm checks weather and daylight sensing intensity of luminance. If the weather is clouded or rainy, the algorithm shorten the interval of photography timing. If it is in dark, the algorithm initializes the interval because the still-picture will be black one.

For the second algorithm which introduces audience request, in addition to the first algorithm, it shortens the interval when an audience request is received. The audience requests are sent from audience clients by pushing on a request button on the user interface. If our hypotheses are true, the first algorithm based on road situation will be more effective than the fixed interval one and the second algorithm will be more effective than the first algorithm.

## 7 EVALUATION

We introduced the two algorithms of photography timing to the prototype system and evaluate how well the system provide profitable road information to the audience. We compared the effects of the fixed interval scheme and the algorithm based on road situation and the algorithm based on road situation and audience request in an evaluation experiment.

### 7.1 Environment

A broadcaster drove on a predefined route near our university as shown in Fig. 7 and broadcasted the driving scene with the prototype system switching the photography timing algorithms. The fixed or initial interval was set to 60 seconds. The smartphone which was used for the experiment was the au Galaxy S II. The maximum upload speed was 1.8 Mbps and download speed was 3.1 Mbps. The route included a broad road with heavy traffic, a narrow road in the neighborhood of housing estate, and a narrow road with many slopes and curves.

Figure 8 shows the user interface for evaluation. Subjects viewed the broadcasting using the audience client of PC version. A button for audience request, a question and answer section for the evaluation were added to the audience client. The subjects were 20 students who were from 19 to 22 years old, 17 male students and 3 female students. We conducted five broadcastings under the condition as shown in Table 3.


Figure 7: The driving route of the experiment

Table 3: The condition of the broadcastings

| Date | Time | Weather | Num of Subjects |
|---|---|---|---|
| 11/17 | 15:40~15:50 | Rainy | 5 |
| 11/25 | 13:00~13:40 | Cloudy | 2 |
| 11/26 | 15:30~16:10 | Rainy | 3 |
| 12/11 | 13:00~13:40 | Cloudy | 5 |
| 12/24 | 13:45~14:30 | Snowly | 5 |

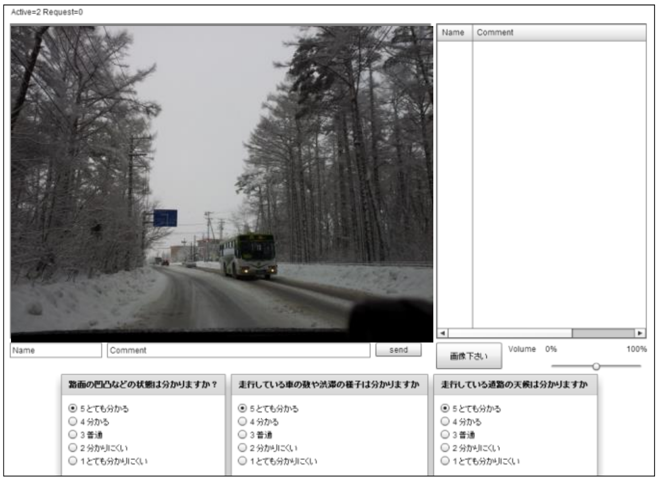



Figure 8: The user interface for evaluation

## 7.2 Results

Figure 9 shows the evaluation result comparing with each algorithm. We asked 5 questions to the subjects and they scored each question on 5-point scale. The blue bar shows the scores of the fixed interval one, the red bar is the algorithm based on road situation, and the green bar is the algorithm based on road situation and audience request. The first question shows usefulness of the proposed system for the route selection. The second question shows adequateness of the photography timing. The third, fourth and fifth questions show the understandability of weather, congestion and irregularity on the road respectively. For all questions, the green bar which is the score result of the algorithm based

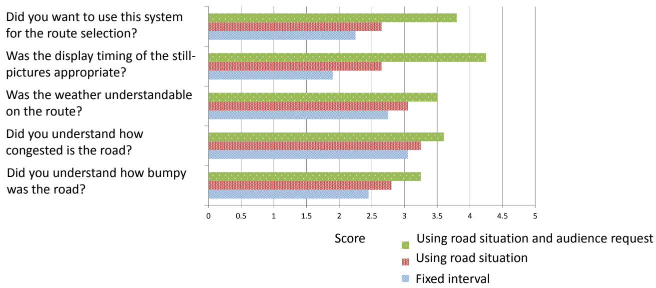



Figure 9: The result of the questionnaire comparing with each algorithm

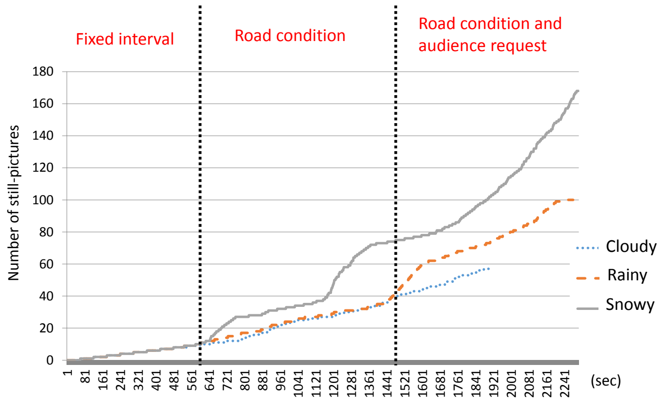



Figure 10: The number of still-pictures of each algorithm

on road situation and audience request is the highest. The score exceed 3 point which is the average score. Especially, the score of the second question exceeds 4 point. The red bar which is the score result of the algorithm based on only road situation is higher than the blue bar which is that of the fixed interval one. Meanwhile the red bar scores below the average score on first, second and fifth questions. This result shows the audience request is effective to provide timely still-pictures for sharing road information. This means that one of our hypotheses, "the demand of road information is different from person to person" is verified.

Figure 10 shows the number of still-pictures of each algorithm in different weather conditions. The horizontal axis indicates elapsed time from the beginning of the broadcasting. The vertical axis indicates the accumulated number of still-pictures. The algorithms were switched at the boundary of the vertical dotted line. From this graph, we found the number of still-pictures increased in worse weather conditions. Thus, the snowy condition increased the number of still-picture than rainy and cloudy conditions because the road surface was in the worse condition by fallen snow. The sensors of the broadcaster smartphone detect it and the algorithms shorten the photography interval. Considering the result of the questionnaire in Fig. 9 and the result of the number of still-pictures in Fig. 10, one of our hypotheses, "the demand of road information is changed depending on the road situation" is verified.

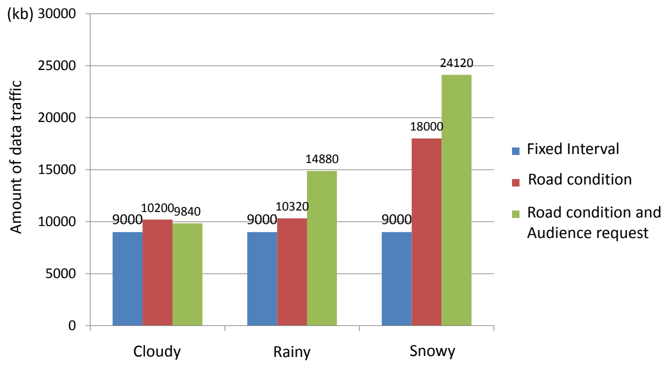



Figure 11: The amount of data traffic of each algorithm

At last, we evaluated the amount of data traffic of each algorithm in order to realize stable broadcasting without large increase in data traffic. Figure 10 shows the amount of data traffic of each algorithm. The horizontal axis indicates weather and the vertical axis indicates the amount of data traffic. The green, red and blue bar means the same as Fig. 11. From this graph, we found the data traffic increased in rainy and snowy weather conditions but the amount could be acceptable value. The amount of data traffic was highest in the snowy weather condition with audience request and it was about 24 MB. Comparing with the red bar, the data traffic increased about 30 % in the snowy weather condition. However, if we used a video streaming for sharing road information by Ustream in the same condition, the data traffic got about 114 MB. Since the algorithm based on road condition and audience request reduces about 80 % data traffic comparing with the video streaming scheme and realizes high user satisfaction, our proposed system can be effective and the photography timing algorithm should be based on road situation and audience request.

Additionally we asked the subjects about attention points in driving a vehicle for the future research. Figure 12 shows the results. The most subjects answered paying attention to the weather and it coincide with the result of figs. 10 and 11. Road irregularity and frequency of congestion which were introduced to our algorithm were also important for half the number of the subjects. These factors improved scores of our algorithms. On the other hand, road structure such as number of lanes, distance and time to the destination were important for more than half the number of the subjects. These factors will be effective to improve our algorithms for the future.

## 8 CONCLUSION

In this paper, we proposed a road information sharing scheme with a still-picture Internet broadcasting system. From the preliminary experiment, the right photography timing was an issue for the system. About the photography timing, we proposed two hypotheses that “the demand of road information is changed depending on the road situation” and “the demand of road information is different from person to person”. Based on the hypotheses, we developed two photography timing algorithms which changes the photography timing based on road situation utilizing sensors of

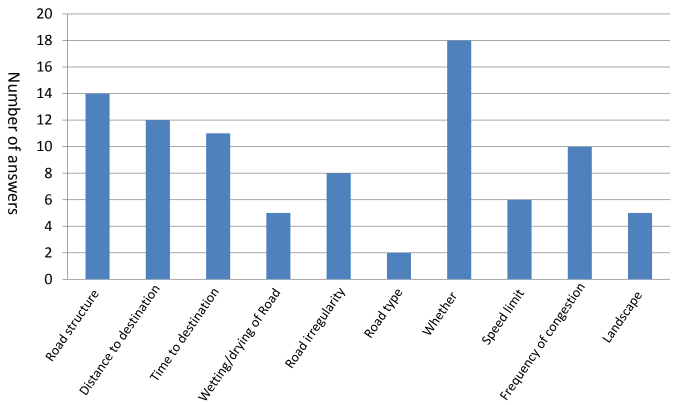



Figure 12: Attention points in driving a vehicle

a smartphone, and based on road situation and audience request. From the evaluation, we found the algorithm based on road situation and audience request was most effective and our two hypotheses were proven.

In future work, we will improve the usability of the proposed system (e.g. the audience can see the vehicle information of the broadcasting in addition to still-pictures, and can watch the past broadcast programs by archiving environment.). We will also study a business model to provide more motivation to broadcast the driving.

## REFERENCES


[1] VICS, http://www.vics.or.jp/english/vics/index.html

[2] Y. Nakano, Y. Saito and Y. Murayama: “A Proposal for a Still Picture Internet Broadcasting System with Dynamic Picture Quality Adjustment based on Audience Requests for Smartphone Broadcasters,” The 1st IEEE Global Conference on Consumer Electronics, pp. 360-364 (2012).

[3] Y. Nakano, N. Hirota, Y. Saito and Y. Murayama: “An Implementation of a Still Picture Internet Broadcasting System with Audience-oriented QoS Control for Smartphones and PCs Audience,” Computer Software and Applications Conference Workshops (COMPSACW), pp.523-527 (2013).

[4] Y. Nakano, Y. Saito and Y. Murayama: “Still Picture Internet Broadcasting System with Audience-oriented Bandwidth Control for Smartphone Broadcasters,” The Ninth International Conference on Systems, pp. 102-105 (2014).

[5] T. Nakata and J. Takeuchi: “Mining traffic data from probe-car system for travel time prediction,” Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 817-822 (2004).

[6] Honda vehicle tracking map, http://www.honda.co.jp/internavi/LINC/service/disastermap/

[7] Ustream, http://www.ustream.tv/

[8] NicoNico Live, http://live.nicovideo.jp/

[9] D. Munter, A. Kotteritzsch, T. Islinger: “Improving Navigation Support by Taking Care of Drivers’ Situational Needs,” Proceedings of the 4th International

Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '12), pp.131-138 (2012).
[10] Honda SAFETY MAP, https://safetymap.jp/
[11] EuroRAP, http://www.eurorap.org/
[12] SafeRoadMaps, http://saferoadmaps.org/
[13] MDOT – Mi Drive, http://mdotnetpublic.state.mi.us/drive/
[14] P. Mohan, V. Padmanabhan, R. Ramjee: Nericell: “rich monitoring of road and traffic conditions using mobile smartphones,” Proceeding SenSys'08 Proceedings of the 6th ACM conference on Embedded network sensor systems, pp. 323-336 (2008).
[15] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, H. Balakrishnan: “The pothole patrol: using a mobile sensor network for road surface monitoring,” Proceeding MobiSys '08 Proceedings of the 6th international conference on Mobile systems, applications, and services, pp. 29-39 (2008).

(Received November,17,2015)

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

**Yuko Murayama** is a professor at the Department of Computer Science at Tsuda College, Japan. Her research interests include internet, network security, human aspects of security and trust. She had a B.Sc. in Mathematics from Tsuda College, Japan in 1973 and had been in industry in Japan. She had M.Sc. and Ph.D. both from University of London (attended University College London) in 1984 and 1992 respectively. She had been a visiting lecturer from 1992 to 1994 at Keio University, and a lecturer at Hiroshima City University from 1994 to 1998. She has been with Iwate Prefectural University from April 1998 to 2015 and with Tsuda College since 2016. She is IFIP Vice President as well as TC-11 Chair. She serves as the Chair of the Security Committee for IPSJ (Information Processing Society of Japan) as well as a secretary for IPSJ Special Interest Group on Security psychology and trust (SIG on SPT). She is a senior life member of IEEE, as well as a member of ACM, IPSJ, IEICE, ITE and Information Network Law Association.

# Parallel Multiple Counter-Examples Guided Abstraction Loop — Applying to Timed Automaton—

Kozo Okano†, Takeshi Nagaoka‡, Toshiaki Tanaka‡, Toshifusa Sekizawa§, and Shinji Kusumoto‡

†Faculty of Engineering, Shinshu University, Japan
‡Graduate School of Information Science and Technology, Osaka University, Japan
§College of Engineering, Nihon University, Japan
†okano@cs.shinshu-u.ac.jp,
§sekizawa@cs.ce.nihon-u.ac.jp,
‡kusumoto@ist.osaka-u.ac.jp

***Abstract*** - A model checking technique proves that a given system satisfies given specifications by searching exhaustively a finite transition system which represents the system's whole behavior. If the system becomes large, it is impossible to explore the whole states in reasonable time due to both of CPU time used and memory space where the model is stored. This is called the state explosion problem. One of the solutions to avoid the state explosion problem is using a model abstraction technique. In usual, constructing such an abstract model from the original model becomes error-prone. Hence, automatic generation techniques of abstract models are studied. Especially, Counter-Example Guided Abstraction Refinement (CEGAR) is considered as a promising technique because it automatically refines abstract model if the result is spurious, starting from a small abstract model. We have already proposed a concrete CEGAR loop for a timed automaton. This iteration loop refines the model in fine granularity level. It avoids the state explosion, however, the number of loops increases. This paper proposes a revised technique where multiple counter-examples are simultaneously applied in the refinement step of CEGAR. This device reduces the number of iteration loops. Experimental results show the improvement.

***Keywords***: CEGAR, Timed Automaton, Model Checking

## 1 INTRODUCTION

Recently, information systems play important roles in social activities, thus software reliability becomes important. Model checking techniques [6] prove that a given system satisfies given a specification by searching exhaustively a finite transition system which represents the system's whole behavior. As systems become larger and more complicated, however, it is difficult to prove the reliability of the systems by model checking, because they need searching for whole states completely. For a large system, it is impossible to explore the whole states in reasonable time. Sometimes its model size becomes larger than physical memory size of typical computers. This is called the state explosion problem.

One of the solutions to avoid the state explosion problem is a model abstraction technique.

In usual, constructing such an abstract model from the original model becomes error-prone. Hence, automatic generation techniques of abstract models are studied.

However, such abstraction techniques in general cannot appropriately control the model size. We want an appropriate abstract model which is small enough to perform model checking and also precise enough to obtain a correct answer by model checking.

In order to obtain a better abstract model, automatic iteration techniques to perform whole cycle of abstraction, model checking, simulation, and refinement, have been studied.

Counter-Example Guided Abstraction Refinement (CEGAR) [8] is the root of such studies.

In verification of real-time systems, a timed automaton is used [4], which can represent behavior of a real-time system. For a timed automaton, a real-valued clock constraint is assigned to each state of finite automaton (called a location). Therefore, it has an infinite state space which is represented in a product of discrete state space made by locations and continuous state space made by clock variables. In a traditional way of model checking for a timed automaton, using the property that we can treat the state space of clock variables as a finite set of regions, thus we can perform the model checking on a timed automaton. The size of the model, however, increases exponentially with clock variables; thus, an abstraction technique is needed.

Paper [18] firstly shows a concrete CEGAR loop for timed automata based on predicate abstraction techniques. It uses two abstraction models, over-approximation and under-approximation, while our previous approach [19] constructs an abstraction model based on only over-approximation. Their approaches are similar to our approach in a sense that a location is divided into two state while abstraction. Paper [19] proposed a concrete CEGAR loop for timed automaton. This iteration loop refines the model in fine granularity level. It avoids the state explosion, however, the iteration grows.

### 1.1 Contributions

This paper proposes a revised technique where multiple counter-examples are simultaneously applied. This device reduces the number of iteration loops.

CEGAR automatically generates a moderate model to perform model checking, but sequential application of counter-examples might consume time and memory space. Our method reduces the number of iteration loops, therefore, it also reduces time and space.

The concrete contributions are summarized as follows.

1. We consider a new CEGAR loop (algorithm) in which multiple counter-examples are simultaneously applied.

2. We give proofs for the algorithm including its termination.

3. We prototyped the algorithm, performed experiments and obtained results which show effective performance of our proposed algorithm.

### 1.2 Related Work

Other related work include papers [13, 9, 11, 7, 3, 14], and [16].

He et al. [13] has proposed a time abstraction technique and CEGAR loop with time abstraction technique and a compositional technique. The compositional technique reduces state explosion occurring when we produce a product automaton from a network of timed automata. Paper [9] proposes abstraction using lattice structure and its based model checker. Paper [11] mentioned an abstraction method for timed automata. Papers [7] and [3] deal with hybrid automata and provide CEGAR for the model. Paper [14] proposes CEGAR loop for probabilistic automata. Paper [16] use SAT solvers for model abstraction.

None of these approaches, however, deals with refinement with multiple counter-examples.

### 1.3 Organization of the Paper

This paper is organized as follows. Section 2 presents introductory material related to timed automata. Section 3 presents a short review of our previous proposed CEGAR for timed automata. Section 4 will provide our proposed multiple counter-examples abstraction refinement loop. Section 5 will shortly give explanation on our prototype system. Section 6 and 7 provide experimental results and discussions, respectively. The final section concludes the paper.

## 2 PRELIMINARIES

Here, we give a definition of a timed automaton and its related notions.

### 2.1 Timed Automaton

No one can control flow of time. One can only measure time using clocks.

A timed automaton also uses clocks to refer time. The clocks can be regard as precise analog clocks. Every clock autonomously uniformly and at the same rate increases the value, independently from the behavior of timed automaton. A timed automaton cannot control the clocks except for reset; it can neither put some clocks forward, backward nor stop them. It can only reset some of clocks. The reset clocks make their values 0. they, however, immediately increase their values again.

**Definition 2.1** (Clock set $C$)**.** *By $C$ we denote a finite set of clocks. By $x_i$ $(0 \leq i \leq |C|-1)$ we denote an element (each clock) in $C$.*

When there is no confusion we might use literals (without index) $x, y, z$, and so on to denote clocks.

Since each clock has its time value as a non-negative real, notion of "clock evaluation" is needed.

**Definition 2.2** (Clock Evaluation)**.** *Clock evaluation $\nu (\in \mathbb{R}_{\geq 0}^{|C|})$ for clock set $C$ is a $|C|$-dimension vector over $\mathbb{R}_{\geq 0}$.*

*An $i$-th element $\nu^i$ of $\nu$ corresponds to the time value of clock $x_i$.*

We use the term "evaluation" according to the original paper [1]. Paper [1] defines the evaluation as a mapping from clocks to reals, however, we define $\nu$ just as a real vector, in this paper. Since clock evaluation changes according to the elapsed time, and a timed automaton might reset some of clocks to $0$ when a transition fires, we introduce two operations on clock evaluation.

**Definition 2.3** (Operations on Clock Evaluation)**.** *For a real value $d$, $\nu + d = (\nu^0 + d, \nu^1 + d, \ldots, \nu^{|C|-1} + d)$.*

*For a set of clocks $r$, $r(\nu) = (r(\nu^0), r(\nu^1), \ldots, r(\nu^{|C|-1}))$, where*

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

The first operation $+d$ means that every clock increases its value uniformly and at the same rate. The second operation $r(\cdot)$ means that every clock specified in $r$ are reset.

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

**Definition 2.4** (Differential Inequalities on $C$)**.** *Syntax of a differential inequality $in$ on a clock set $C$ is given as follows:*

$$in ::= x_i - x_j \sim a$$

$$\mid x_i \sim a,$$

*where $x_i$ and $x_j \in C$, $a$ is a literal of an integer constant, and $\sim \in \{\leq, \geq, <, >\}$.*

*Differential inequalities $x_i \sim a$ and $x_i - x_j \sim a$ are* true *iff $\nu^i \sim a$ and $\nu^i - \nu^j \sim a$ are* true*, respectively.*

**Definition 2.5** (Clock Constraints on $C$)**.** *Syntax of a clock constraint $cc$ on a clock set $C$ is given as follows:*

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

*where $in$ is a differential inequality on $C$.*

*$cc_i \wedge cc_j$ is* true *iff both $cc_i$ and $cc_j$ are* true*.*

*By $c(C)$, we denote whole set of clock constraints on a clock set $C$.*

Since clock constraint $f$ can be regarded as a function

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

we introduce a notation $f(\nu)$. It is evaluated to true or false by evaluating each clock $x_i$ as $\nu^i$.

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

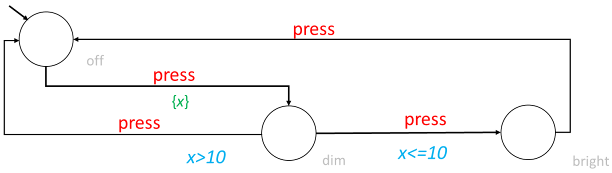



Figure 1: An Example Timed Automaton Representing Mug-light

**Definition 2.6** (Timed Automaton)**.** *A timed automaton $\mathscr{A}$ is a six-tuple $(A, L, l_0, C, I, T)$, where*
*$A$: a finite set of actions;*
*$L$: a finite set of locations;*
*$l_0 \in L$: an initial location;*
*$C$: a clock set;*
*$I : L \to c(C)$: a mapping from a location to a clock constraint, called a location invariant, or simply an invariant; and*
*$T \subset L \times A \times c(C) \times 2^C \times L$ is a set of transitions, where $c(C)$ is a set of clock constraints; and $2^C$ is a super set of sets of clocks.*

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

We denote $(l_1, a, g, r, l_2) \in T$ by $l_1 \overset{a,g,r}{\to} l_2$.

**Example 1.** *Figure 2.1 is an example of a timed automaton, $\mathscr{A}_L = (\{\text{press}\}, \{\text{off}, \text{dim}, \text{bright}\}, \text{off}, \{x\}, \emptyset, T)$, where $T =$*
$\{\text{off} \overset{\text{press},\text{true},\{x\}}{\to} \text{dim},$
$\text{dim} \overset{\text{press},x\leq 10,\emptyset}{\to} \text{bright},$
$\text{dim} \overset{\text{press},x>10,\emptyset}{\to} \text{off},$
$\text{bright} \overset{\text{press},\text{true},\emptyset}{\to} \text{off}\}$ .

*Please note that guards with value* true*, and empty clock resets are omitted in Fig. 2.1,*

Example 1 shows a timed automaton representing behavior of a mug-light with two brightness modes. Here, we informally explain the behavior of this time automaton. The initial state is location "off" and the value of clock $x$ is 0. If "press" action fires, then state is changed to location "dim", which means that the mug-light is dim. With this transition the value of clock $x$ is reset to 0. The control of a timed automaton can stay in a location as long as its invariant is satisfied. Unfortunately, the example has no location invariants. At location "dim," the control can stay any unit of time. If the value of clock $x$ is greater than 10 units of time, "press" action changes the location to location "off," which means the mug-light is switched off. Otherwise, *i.e.*, the value of clock $x$ is less than or equal to 10 units of time, "press" action changes the location to location "bright," which means that pressing twice immediately makes the mug-light bright. At location "bright," "press" action changes the location to location "off," regardless of the value of clock $x$.

Example 2 is another example to explain evaluation of a guard and an invariant.

**Example 2.** *Let assume that $C$ and $I(l_2)$ (a location invariant for $l_2$) are $\{x, y\}$ and $y > 6$, respectively. Consider a transition $l_1 \overset{a,x>0\wedge y\geq 3,\{y\}}{\to} l_2$.*

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

Dynamic of a timed automaton can be expressed via a set of locations and a set of clock evaluations. Changes of one state to a new state can be as a result of firing of an action or elapse of time.

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

**Definition 2.7** (Labelled Transition System)**.** *A labelled transition system (LTS) is three-tuple $(S, s_0, T)$, where $S$, $s_0 \in S$ and $T$ are a finite set of states, an initial state, and a set of transitions, where $T \subset S \times (A \cup \mathbb{R}_{\geq 0}) \times S$.*

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

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

**Definition 2.8** (A Run of an LTS)**.** *A run of LTS $(S, s_0, T)$ is defined as follows.*
*$s_0 \overset{\alpha}{\Rightarrow} s'$ is a run of $(S, s_0, T)$, if $s_0 \overset{\alpha}{\Rightarrow} s' \in T$.*
*Let $\sigma_i$ be a run of $(S, s_0, T)$, ending with state $s_i$. For $s_i \overset{\alpha}{\Rightarrow} s_j \in T$, and $\sigma_i$, $\sigma_i \overset{\alpha}{\Rightarrow} s_j$ is also a run of $(S, s_0, T)$.*

**Definition 2.9** (Semantics of a timed automaton)**.** *For a given timed automaton $\mathscr{A} = (A, L, l_0, C, I, T)$, its corresponding LTS$(S, s_0, T)$ can be formalized as follows.*
*$S = L \times \mathbb{R}^{|C|}_{\geq 0}$.*
*$s_0 = (l_0, \mathbf{0})$, where $\mathbf{0}$ is a $|C|$-dimension vector and each of whose elements is 0.*

*Transition $s \overset{\alpha}{\Rightarrow} s'$ is defined by Definition 2.10.*

**Definition 2.10** (Semantics of transion of a timed automaton)**.** *For transition $l_1 \overset{a,g,r}{\to} l_2$, its corresponding transition of LTS can be defined as follows.*

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

The first one is called an action transition, while the other is called a delay transition.

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

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

Please note that an action transition does not consume time, while a delay transition consumes time staying in the same location.

**Definition 2.11** (run of a timed automaton)**.** *For a timed automaton $\mathscr{A}$, a run $\sigma$ is a finite or infinite run of its corresponding LTS.*
$\sigma = (l_0, \nu_0) \overset{\alpha_1}{\Rightarrow} (l_1, \nu_1) \overset{\alpha_2}{\Rightarrow} (l_2, \nu_2) \overset{\alpha_3}{\Rightarrow}, \dots$ ,
*where $\alpha \in A \cup \mathbb{R}_{\geq 0}$.*

In usual, as a run of a timed automaton, we only consider an alternate run of delays and actions, in which delay transitions and action transitions alternately occur.

**Example 3.** *One of possible runs of $\mathscr{A}_L$ is*
$(\mathrm{off}, (0)) \overset{0.5}{\Rightarrow} (\mathrm{off}, (0.5)) \overset{\mathrm{press}}{\Rightarrow} (\mathrm{dim}, (0)) \overset{9.8}{\Rightarrow} (\mathrm{dim}, (9.8)) \overset{\mathrm{press}}{\Rightarrow} (\mathrm{bright}, (9.8)) \cdots$ .

Please note that in the run of Example 3, delay transitions and action transitions alternately occur.

For further detail about time automata, refer to [4] and [20].

## 2.2 Model Checking

Model checking of an automaton can be formulated as follow.

**Definition 2.12** (Model Checking)**.**
*Input1: an automaton $A$*
*Input2: a temporal logic expression $p$*
*Output: $A \models p$ or $A \not\models p$*
*Output(optional): If $A \not\models p$, then a counter-example $CE$*

In usual, Computational Tree Logic (CTL) is used as a temporal logic for a timed automaton [4].

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

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

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

**Definition 2.13** (Reachability Problem)**.** *Model checking of a property $\neg\mathbb{EF}s$ (on $A$) is called reachability problem (on $A$).*

Reachability problem is a fundamental and essential problem for model checking since the algorithm for reachability problem is core of that of general model checking algorithm.

In this paper, we consider only reachability problem.

Counter-example $CE$ is usually a run of automaton $A$ which specifies concretely that property $p$ does not hold.

For reachability problem, its counter-example is a run to reach state $s$.

Nevertheless the number of states produced by a timed automaton is infinite due to the cardinality of reals, reachability problem is decidable [4], since time space can be divided into finite equivalence classes.

In papers [4] and [10], a data structure DBM is introduced to represent clock constraints. Several operations on DBM are also introduced. Using these operations, we can efficiently calculate time space of timed automata.

**Definition 2.14** (DBM (Difference Bound Matrix))**.** *DBM is a set of differential inequalities on two clock variables, and represents a state space which satisfies all inequalities over it (the state space is called a* **zone***).*

*DBM represents these set of inequalities as a $|C_0| \times |C_0|$ matrix, where $C_0 = C \cup \{ \mathbf{0} \}$. Symbol* **0** *is a special variable which means a constant value $0$.*

*The $(i, j)$-th entry ($D_{i\ j}$) of the matrix stands for a differential inequality of $x_i - x_j$ for $x_i, x_j \in C_0$.*

*Suppose there is an inequality $x_i - x_j \preceq n$ for $\preceq \in \{ <, \leq \}$, the $(i, j)$-th entry $D_{i\ j}$ is represented by $(n, \preceq)$. When $x_i - x_j$ is unbounded, the entry $D_{i\ j}$ is represented by $\infty$.*

*In addition, the upper bound and lower bound of $x_i$ itself are indicated by $D_{0,i}$ and $D_{i,0}$, respectively.*

A zone is the solution set of a clock constraint that is the maximal set of clock assignments satisfying the constraint [4]. It is well-known that such sets can be efficiently represented and stored in memory as DBMs.

There are several model checkers. Typical model checkers produce one counter-example when a property does not hold.

Algorithms of model checking are essentially exhaustive search of whole possible runs. Therefore, if the number of states becomes larger, the complexity becomes larger exponentially or intractable. Such a situation is called "state explosion." Thus, we have to reduce the number of states by automatic abstraction.

# 3 CEGAR FOR TIMED AUTOMATA

In usual, CEGAR loop firstly generates small abstract model from the original model. The first abstract model is small enough to perform model checking, however it is usually "over-approximated," *i.e.*, many states are extremely merged into a same state. Therefore, model checking process usually produces a spurious counter-example for the first abstract model. Using the counter-example, CEGAR loop automatically generates a next abstract model, which has more states than the former. Using the next abstract model, we perform model checking again. Such iteration relaxes the over-approximation step by step. At some point of the iteration, we would obtain an appropriate abstract model for model checking.

## 3.1 Basic Algorithm

This section provides the base algorithm on abstraction refinement technique for the timed automata given in [18] and [19]. As mentioned above the algorithm in [18] and that of [19] is similar in abstract level. However, this paper proposes an extended method of [19], therefore, we describe the base algorithm based on [19].

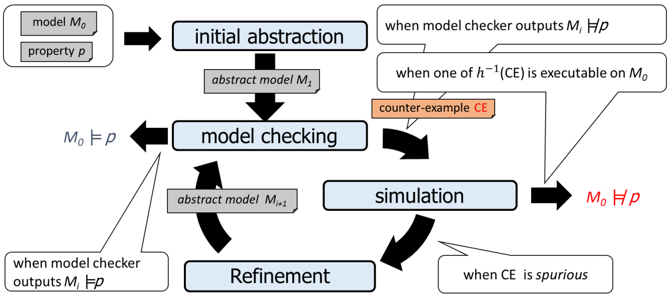



Figure 2: Basic Flow of CEGAR

**Definition 3.1** (Abstraction assumption)**.** *The following condition is called Abstraction assumption.* $\forall i > 0 : (M_i \models p \rightarrow M_0 \models p)$*, where* $M_i$ *is* $i$*-th abstract model. Model* $M_0$ *is the original model.*

The abstraction assumption should hold during CEGAR loop.

CEGAR loop [8] consists of the following four steps, namely *Initial abstraction*, *Model checking*, *Simulation*, and *Refinement*.

Figure 2 shows the basic flow of CEGAR loop.

1. **Initial abstraction**
   An original model $M_0$ and a property $p$ are given as input, and we abstract the original model $M_0$ and obtain an initial abstract model $M_1$.

   We abstract the model preserving the abstraction assumption.

2. **Model checking**
   We perform model checking on the abstract model $M_i$. If a model checker outputs $M_i \models p$, then we can conclude that $M_0 \models p$ by the abstraction assumption. Then, we stop the loop. Otherwise, *i.e.*, the model checker outputs $M_i \not\models p$. Also a counter-example $\hat{\rho}_i$ is generated. We have to check every counter-example in $P_i$ on the original model $M_0$, where $P_i$ is a set of concretized runs on $M_0$, each of which is obtained from $\hat{\rho}_i$ by applying inverse of abstraction function $h$.

3. **Simulation**
   We check every concretized run in $P_i$ on the original model $M_0$. If one of them is executable on $M_0$, then we conclude that $M_0 \not\models p$, because the found run is a real counter-example on $M_0$ and the property $p$. If none of them is executable on $M_0$, we have to refine $M_i$ so that model checking on $M_{i+1}$ does not produce the counter-example $\hat{\rho}_i$.

   We should notice that checking every run in $P_i$ on $M_0$ can be performed symbolically using symbolical presentation on $P_i$ or $\hat{\rho}_i$. We say that $\hat{\rho}_i$ is spurious when none in $P_i$ is executable on $M_0$.

4. **Refinement**
   If $\hat{\rho}_i$ is spurious, then we refine $M_i$ so that model checking on $M_{i+1}$ does not produce the counter-example $\hat{\rho}_i$.

   The $M_{i+1}$ is obtained automatically using $\hat{\rho}_i$. We repeat the loop by go to Model checking with $M_{i+1}$.

In our previous work [19], we give a concrete algorithm of CEGAR for a timed automaton. In the work, we only consider the reachability property as $p$. Thus, we check that $\neg\mathbb{EF}l_e$, where $l_e$ is an error location. The error location is a location where we think the control never reach.

The following subsections describe the details of each step.

## 3.2 Initial Abstraction

In Initial abstraction, we remove all of clock attributes from the given timed automaton [19].

**Definition 3.2** (Abstraction Function $h$)**.** *For a timed automaton* $\mathscr{A}$ *and its semantic model (LTS)* $(S, s_0, \Rightarrow)$*, an abstraction function* $h : S \rightarrow \hat{S}$ *is defined as follows:*

$$h((l, \nu)) = l.$$

*The inverse function* $h^{-1} : \hat{S} \rightarrow 2^S$ *of* $h$ *is also defined as* $h^{-1}(\hat{s}) = (l, D_{I(l)})$ *where* $\hat{s} = l$ *and* $D_{I(l)}$ *is a region satisfying* $I(l)$ *representing by DBM.*

**Definition 3.3** (Abstract Model)**.** *An abstract model* $\hat{M} = (\hat{S}, \hat{s}_0, \hat{\Rightarrow})$ *of a given timed automaton* $\mathscr{A}$ *is defined as follows:*

- $\hat{S} = L$;
- $\hat{s_0} = l_0$*; and*
- $\hat{\Rightarrow} = \{(l_1, a, l_2) \mid l_1 \overset{a,g,r}{\hat{\Rightarrow}} l_2 \in T\}$.

For $\mathscr{A}$, using its LTS $(S, s_0, \Rightarrow)$, we can say that $\hat{\Rightarrow}$ is $\{(h(s_1), a, h(s_2)) \mid s_1 \overset{d}{\Rightarrow} s_{1'}, s_{1'} \overset{a}{\Rightarrow} s_2 \in \Rightarrow\}$.

The $i$-th abstract model $\hat{M}_i = (\hat{S}_i, \hat{s}_{i,0}, \hat{\Rightarrow}_i)$ is obtained from the $i$-th timed automaton $\mathscr{A}_i = (A_i, L_i, l_{i,0}, C_i, I_i, T_i)$ by Definition 3.3.

**Definition 3.4** (Abstract Counter-Example)**.** *A counter-example on* $\hat{M} = (\hat{S}, \hat{s}_0, \hat{\Rightarrow})$ *is a run of states of* $\hat{S}$ *and labels. An abstract counter-example* $\hat{\rho}$ *of length* $n$ *is represented in* $\hat{\rho} = \hat{s}_0 \overset{a_1}{\hat{\Rightarrow}} \hat{s}_1 \overset{a_2}{\hat{\Rightarrow}} \hat{s}_2 \overset{a_3}{\hat{\Rightarrow}} \cdots \overset{a_{n-1}}{\hat{\Rightarrow}} \hat{s}_{n-1} \overset{a_n}{\hat{\Rightarrow}} \hat{s}_n$*. A set* $P$ *of runs on* $\mathscr{A}$ *obtained by concretizing a counter-example* $\hat{\rho}$ *is also defined as follows using the inverse function* $h^{-1}$*:*

$$P = \{s_0 \overset{d_0}{\Rightarrow} s'_0 \overset{a_1}{\Rightarrow} s_1 \overset{d_1}{\Rightarrow} s'_1 \overset{a_2}{\Rightarrow} s_2 \overset{d_2}{\Rightarrow} \cdots \overset{a_n}{\Rightarrow} s_n \mid \bigwedge_{i=0}^{n-1} (s_i \in h^{-1}(\hat{s}_i) \ \wedge d_i \in \mathbb{R}_{\geq 0} \wedge s_i \overset{d_i}{\Rightarrow} s'_i \ \wedge s'_i \overset{a_i}{\Rightarrow} s_{i+1}\}.$$

We assume that a counter-example is a finite run [19]. We restrict the property to check as reachability, this assumption is reasonable. For a case of loop structures, see [19].

Example 4 shows an example of Initial Abstraction.

**Example 4.** *Figure 3 shows a timed automaton and its abstract model.*

*The original timed automaton is* $A_0$ $(= M_0)$*. Its abstract model* $\hat{M}_0$ *is just an automaton without clock constraints.*

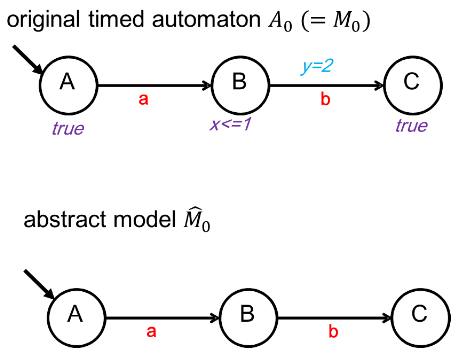



Figure 3: An Example of Abstraction

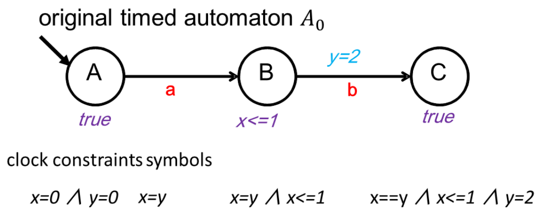



Figure 4: An Example of Simulation

## 3.3 Model Checking

Abstract model $\hat{M}_i$ is a just automaton, therefore, we can use several model checkers at this step. In Paper [19], we use UPPAAL to model check. In our new proposed method, however, we use our original model checker in order to produce multiple counter-examples.

**Example 5.** *For an abstract model $\hat{M}_0$ in Fig. 3, a property $\neg \mathbb{EF}\ C$ does not hold, since clearly we can reach state $C$ from the initial state $A$.*

*Any appropriate model checker outputs $\hat{M}_0 \not\models \neg \mathbb{EF}\ C$ and its counter-example $A \to B \to C$.*

## 3.4 Simulation

Using the DBM library provided by UPPAAL team, we have developed a simulation program. Let $P_i$ be a set of concretized counter-examples produced by $\hat{\rho}_i$, which is a counter-example of $\hat{M}_i$.

Instead of checking each element of $P_i$, we use DBM and $\hat{\rho}_i$ to simulate on $\mathscr{A}_0$ using symbolic simulation technique.

Example 6 shows an example process of Simulation.

**Example 6.** *Figure 4 shows an example process of Simulation.*

*Simulation checks whether $A \to B \to C$ is possible on the original $A_0$ using symbolic simulation technique. At location $A$, we use a DBM structure representing $x = 0 \wedge y = 0$, which stands for the initial state. Since $A$ has no invariant, we change the DBM structure to represent $x = y$, which shows that clocks $x$ and $y$ increase their values at the same rate. According to the counter-example, we move to location $B$. At location $B$, we obtain a DBM structure representing $x = y \wedge x \leq 1$. At this point, we find that transition $B \to C$ cannot fire since the guard of transition $B \to C$, i.e., $y = 2$ and current clock constraint $x = y \wedge x \leq 1$ are conflict. Thus, $x \leq 1 \wedge y = 2 \wedge x = y$ is* false.

*We can find that at location $B$ there are no transitions due to time constraints.*

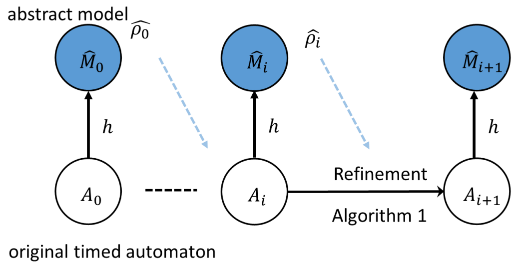



Figure 5: Relation among the Models derived by Refinement

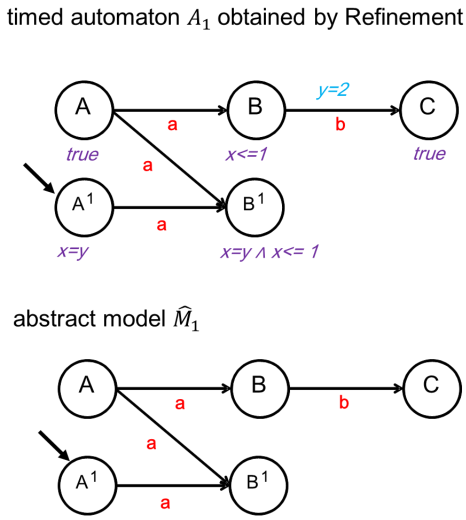



Figure 6: An Example of Refinement

## 3.5 Refinement

The $(i+1)$-th abstract model $\hat{M}_{i+1}$ is obtained from a timed automaton $\mathscr{A}_{i+1}$ using the abstraction function $h$. The $(i+1)$-th timed automaton $\mathscr{A}_{i+1}$ is obtained from the $i$-th timed automaton $\mathscr{A}_i$ and a counter-example $\hat{\rho}_i$.

Paper [19] shows a concrete algorithm for refinement (see **Appendix A**). We call the algorithm Algorithm 1 (or Refinement). Algorithm 1 has two inputs $\mathscr{A}_i$ and $\hat{\rho}_i$, and outputs $\mathscr{A}_{i+1}$.

Figure 5 summarizes the relation among the models.

In usual, Algorithm 1 appends additional locations and transitions to $\mathscr{A}_i$ so that $\hat{M}_{i+1}$ can tell two states which are merged in $\hat{M}_i$ as a result of over-approximation.

Example 7 shows an example process of Refinement.

**Example 7.** *We use the same example in Example 6. Figure 6 depicts the result of applying of Refinement t the original timed automaton.*

*Applying Algorithm 1, the refinement algorithm, we can obtain the refined timed automaton $\mathscr{A}_1$ and its corresponding abstract automaton $\hat{M}_1$.*

*We can also see that on $\hat{M}_1$ we cannot reach the error location $C$.*

Paper [5] shows that clock conditions in a form of $x-y<c$ cannot be dealt with. Therefore, we assume that the following assumptions in the paper.

**Assumption 1.**

1. *We only check reachability: $\neg\mathbb{EF}l_e$ for model checking.*
2. *The target timed automaton is diagonal-free, which means that the timed automaton does not contain clock conditions in a form of $x-y<c$[5].*
3. *We assume that a counter example is a finite run.*

Hereafter, we assume that Assumption 1 always holds in this paper.

# 4 OUR NEW REVISED CEGAR LOOP

Our revised CEGAR loop differs in Model Checking, Simulation, and Refinement from the previous one.

Here, we describe each of them.

## 4.1 Model Checking

Normally, a model checker produces at most one counter-example. In our algorithm, we use master-worker configuration. Each worker performs model checking and generates a counter-example which we expect to be different to others. We describe how each worker generates a counter-example which we expect to be different to others, in Section 5.

## 4.2 Simulation

If one of counter-examples obtained by workers can be executed on $\mathscr{A}_0 = M_0$ symbolically, then we conclude that $\mathscr{A} \not\models \neg\mathbb{EF}l_e$. Otherwise we perform Refinement using the counter-examples.

## 4.3 Refinement

The master gathers counter-examples from the workers, and performs MultipleRefinement (Algorithm 2) shown in Fig. 7.

Using Algorithm 1, Algorithm 2 in Fig. 7 applies each counter-example $\hat{\rho}$ in a given $S_i$. The result is sequentially reflected in the given timed automaton $\mathscr{A}_{i+1}$. In the "for loop body," if a $\hat{\rho}$ is not executable on the current tentative $\mathscr{A}_{i+1}$, then for such a counter-example, Algorithm 1 is not applied. The next counter-example in $S_i$ is chosen and the process is repeated.

## 4.4 The Difference between Our Previous Approach and the New Approach

Here, we describe the difference between our previous approach [19] and the new proposed approach.

MultipleRefinement
Inputs $\mathscr{A}_i, S_i$
Output $\mathscr{A}_{i+1}$

/* $S_i = \{\hat{\rho}_0, \hat{\rho}_1, \cdots, \hat{\rho}_k\}$ */
/* $\hat{\rho}_j$ is a counter example produced by worker $j$ */
$\mathscr{A}_{i+1} := \mathscr{A}_i$
**for** $\hat{\rho} : S_i$ **do**
  **if** $\hat{\rho}$ is executable on $\mathscr{A}_{i+1}$ **then**
    $\mathscr{A}_{i+1}$ :=Refinement($\mathscr{A}_{i+1}, \hat{\rho}$)
  **end if**
**end for**
**return** $\mathscr{A}_{i+1}$

Figure 7: Algorithm 2: MultipleRefinement

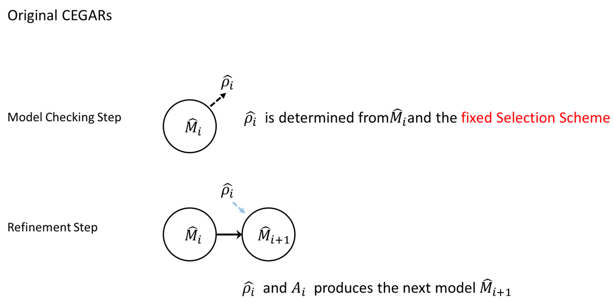



Figure 8: Relation between CE and Abstract Models

Hereafter, we use a simple figure showing the relation between counter-examples and abstract model like as Fig. 8, instead of Fig. 5.

The previous approach [19] performs model checking on an abstract model $\hat{M}_i$. If a model checker says false, then it also produces a counter-example. Based on the counter-example, it performs Refinement and obtains the next abstract model $\hat{M}_{i+1}$.

In general, for a model, multiple counter-examples may exit, if the property to check is not valid for the model. The conventional CEGAR loop including our previous method, the criterion which chooses a counter-example as its output from such candidates of counter-examples, is fixed through the loop.

For example, we might use a criterion that always chooses the fast found one; we might use other criteria that always chooses the shortest counter-example; and so on. For ease of discussion, we call such a criterion a "Selection Scheme."

The conventional approach first fixes its Selection Scheme, then it repeats model checking and refinements from the initial abstract model $\hat{M}_0$ and it finally obtains an adequate abstract model $\hat{M}_n$ . Let us assume that the sequence of counter-examples used in the process is $[\hat{\rho}_0, \cdots, \hat{\rho}_{n-1}]$.

Please note that $\hat{\rho}_i$ is generated from the fixed Selection Scheme and the current abstract model $\hat{M}_i$.

On the other hand, our new proposed approach first generates simultaneously multiple counter-examples $\hat{\rho}'_0, \cdots, \hat{\rho}'_{k-1}$ from the the initial abstract model $\hat{M}_0$. It then applies these $\hat{\rho}'_0, \cdots, \hat{\rho}'_{k-1}$, regardless of the order, and obtains an abstract

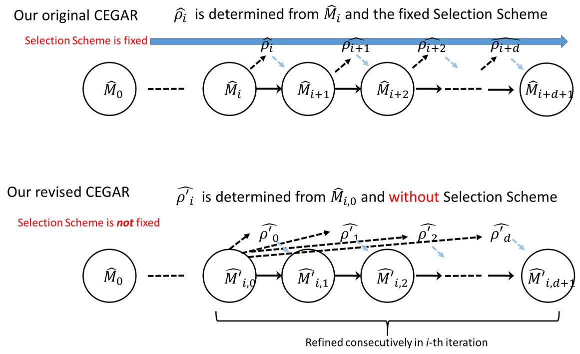



Figure 9: Difference between the Original and the Proposed CEGARs

model $\hat{M}'_k$.

Please note that our new method does not fixes its Selection Scheme. In other words, its Selection Scheme dynamically changes in every sequence in the loop.

We summarize the difference between the original CEGAR and the proposed CEGAR in Fig. 9.

The following question arises.

Even we assume that $k < n$ holds, we cannot conclude that $\hat{M}'_k$ is the same as one of $\hat{M}_1, \ldots, \hat{M}_n$. The reason is that the set $\{\hat{\rho}'_0, \cdots, \hat{\rho}'_{k-1}\}$ is not subset of $\{\hat{\rho}_0, \cdots, \hat{\rho}_{n-1}\}$. Please recall that $\hat{\rho}_i$ is determined by the fixed Selection Scheme and a current abstract model $\hat{M}_i$, while $\hat{\rho}'_i$ is determined by any uncertain Selection Scheme and the initial abstract model $\hat{M}_0$ (of course in general, $\hat{M}_i$).

Regardless the difference, we have to prove that $\hat{M}'_k$ is an adequate abstract model.

In this paper, we don't prove that $\hat{M}'_k$ is one of $\hat{M}_1, \ldots, \hat{M}_n$, because it is not correct in logical.

Instead of it, we prove that $\hat{M}'_k$ preserves Abstraction assumption for the reachability problem.

## 4.5 Proof of the Algorithm

The problem is to ensure that Abstraction assumption is preserved for simultaneous application of multiple counter-examples.

Theorem 1 proves that Abstraction assumption is always preserved nevertheless the order of applying multiple CEs might vary.

**Theorem 1.** *For a given set of counter-examples $S_i$, each of which are generated from model checking on $\mathscr{A}_i$, $\hat{M}_{i+1}$ obtained by Algorithm 2 (and abstraction function $h$) preserves Abstraction assumption.*

Before the proof of Theorem 1, we describe the following propositions.

**Proposition 1.** *Termination of Algorithm 1 [19]*
*Algorithm 1 terminates for reachability problem.*

**Proposition 2.** *Preservation of Abstract Assumption [19]*
*Algorithm 1 preserves Abstraction assumption.*

First we give proof of Theorem 2, which is weaker than Theorem 1.

**Theorem 2.** *For a given set of counter-examples $S_i$, each of which are generated from model checking on $\mathscr{A}_i$, if $\hat{M}_i$ preserves Abstraction assumption, then $\hat{M}_{i+1}$ obtained by Algorithm 2 (and abstraction function $h$) also preserves Abstraction assumption.*

Please note that $S_i$ is a set of counter-examples generated from $\mathscr{A}_i$ with one application of model checking.

The previous approach obtains each $\mathscr{A}_{i+1}$ by applying $\rho_i$ which is generated from $\mathscr{A}_i$ to $\mathscr{A}_i$.

We have to take care of a counter-example which is in $S_i$ but not in $S$.

Theorem 2 holds nevertheless above difference exists. The proof, therefore, uses divide cases.

The following proposition Lemma 4.1 is used in both proofs of Theorems 2 and 3.

**Lemma 4.1.** *If $\rho_j$ is executable on $\mathscr{A}_{(i+1)(j-1)}$ then $\rho_j$ is also a counter-example on $\hat{M}_{(i+1)(j-1)}$.*

A proof of Theorem 2 can be given by induction on the number of application of "for loop body" of Algorithm 2.

*Proof.* Let $j$ be the number of application of "for loop body" of Algorithm 2.

We denote a tentative timed automaton and its abstract model by $\mathscr{A}_{ij}$ and $\hat{M}_{ij}$, respectively. $\mathscr{A}_{ij}$ stands for a tentative timed automaton obtained from $\mathscr{A}_i$ by $j$ times application of "for loop body." $\hat{M}_{ij}$ also stand for its corresponding abstract model. Therefore, $\mathscr{A}_i = \mathscr{A}_{(i+1)(0)}$ and $\hat{M}_i = \hat{M}_{(i+1)(0)}$ hold.

We use proof by induction, induction on $j$.

**Basis**:
$\mathscr{A}_{(i+1)(0)} = \mathscr{A}_i$. Thus, $\hat{M}_{(i+1)(0)}$ is also $\hat{M}_i$. Hence, from the precondition of Theorem 2, we can say $\hat{M}_{(i+1)(0)}$ preserves Abstraction assumption.

**Inductive Step**:
Let us assume that we have already performed $(j-1)$ times the loop body, and obtained a tentative timed automaton namely, $\mathscr{A}_{(i+1)(j-1)}$.

Let also assume that $\hat{M}_{(i+1)(j-1)}$ preserves Abstraction assumption as an inductive assumption.

Now we consider a counter-example $\rho_j$ in $S_i$.

**Case 1**: $\rho_j$ is not executable on $\mathscr{A}_{(i+1)(j-1)}$:
In such a case, Algorithm 1 is not applied. Thus $\mathscr{A}_{(i+1)(j)} = \mathscr{A}_{(i+1)(j-1)}$ holds. Hence, $\hat{M}_{(i+1)(j)} = \hat{M}_{(i+1)(j-1)}$ also holds. $\hat{M}_{(i+1)(j)}$ also preserves Abstraction assumption by the inductive assumption.

**Case 2**: $\rho_j$ is executable on $\mathscr{A}_{(i+1)(j-1)}$:
In such a case, Algorithm 1 can be applied. The condition "$\rho_j$ is executable on $\mathscr{A}_{(i+1)(j-1)}$" implies that "$\rho_j$ is also a counter-example on $\hat{M}_{(i+1)(j-1)}$," by **Lemma** 4.1. By this fact, Proposition 2 and the inductive assumption, we can concoclude that $\hat{M}_{(i+1)(j)}$ also preserves Abstraction assumption.

In any case, $\hat{M}_{(i+1)(j)}$ preserves Abstraction assumption.

Proof by induction, we can say that $\hat{M}_{i+1}$ preserves Abstraction assumption. □

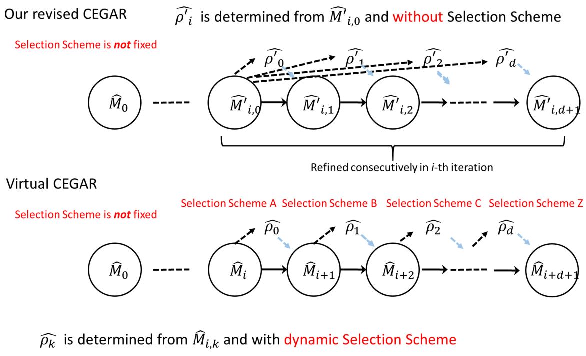



Figure 10: Model Correspondence between the Methods

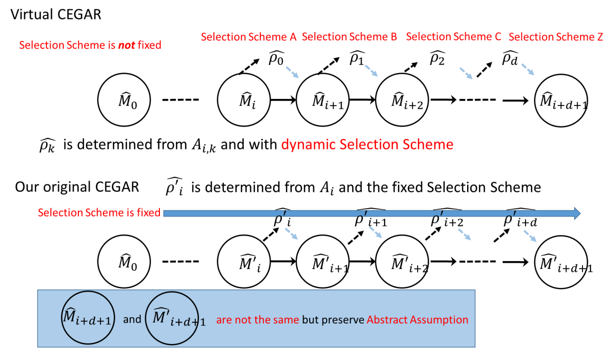



Figure 11: Difference between VIRTUAL CEGAR and our Original CEGAR

Now a proof of Theorem 1 can be also given By induction on $i$.

*Proof.* **Basis**:
Since $\hat{M}_0$ preserves Abstraction assumption [19], we can say $\hat{M}_0$ preserves Abstraction assumption.
**Inductive Step**:
We assume that $\hat{M}_i$ preserves Abstraction assumption. By Theorem 2 we can prove that $\hat{M}_{i+1}$ preserves Abstraction assumption. □

**Theorem 3.** *Termination of CEGAR loop*
*CEGAR loop using Algorithm 2 terminates.*

*Proof.* By Proposition 1 and the fact that $S_i$ is finite set, Algorithm 2 also terminates.

Next we prove the termination of CEGAR loop.

Let a sequence $\hat{\rho}_0, \hat{\rho}_1, \ldots, \hat{\rho}_d$ which are executable counter-examples selected from $S_i$ using Algorithm 2. The index is selection order in Algorithm 2.

From Lemma 4.1, we can say that there is a corresponding loop sequence where each of the loop is application of $\hat{\rho}_j (0 \leq j \leq d)$, in a VIRTUAL CEGAR loop (See Fig. 10). VIRTUAL CEGAR loop is a simlar CEGAR to our original CEGAR, but it uses different Selection Schemes for each application of the loop body.

Please note that the corresponding $\hat{\rho}$ is choosed from the set of possible counter-examples of the corresponding timed automaton with a certain Selection Scheme. Thus, each Selection Scheme is not the same but dynamically changed in VIRTUAL CEGAR.

In a similar way to paper [19], we can say the size of states in $\hat{M}_i$ is also finite. Therefore, CEGAR loop terminates. □

Figure 11 shows the difference between VIRTUAL CEGAR and our Original CEGAR [19]. Please note that Selection Scheme is dynamically changed in VIRTUAL CEGAR. Therefore, the refined abstract models might be different.

Example 8 shows that the case the order does not affect the refined abstract model.

**Example 8.** *Let consider a timed automaton in Fig. 4.5. Due to the clock constraints, neither a transition from $B$ to $C$ nor from $D$ to $E$ is firable.*

*There are two counter-examples: $A \rightarrow B \rightarrow C \rightarrow E$ and $A \rightarrow B \rightarrow D \rightarrow E$.*

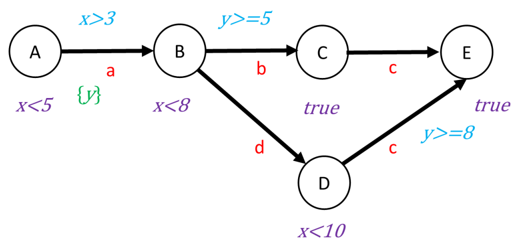



Figure 12: An Example Timed Automaton

*First, let's consider the case the first counter-example is applied. Algorithm 1 generates a copy $B_1$ from location $B$, and generates a transition from $B_1$ to $D$ as well as a transition from $A$ to $B_1$. Finally it removes transition from $A$ to $B$ (in Fig. 4.5). Next it applies the second counter-example. It generates a new location $D_1$ and removes a transition from $B_1$ to $D$ (in Fig. 4.5).*

*Next let's consider the case the second counter-example is firstly applied. It generates a new location $D_1$ then generates also $B_1$. Finally, it removes a transition from $A$ to $B$. The result is the same as Fig. 4.5. The application of the first counter-example does not affect the shape of the timed automaton.*

*Therefore, this example produces the same refinement. Note that we cannot reach location $E$ from $A$ in Fig. 4.5.*

### 4.6 Modularity of Our Parallel Execution Scheme

Our proposed parallel execution algorithm is independent of the original CEGAR loop algorithm, therefore for any correct CEGAR loop algorithm, our parallel execution scheme also works correctly. The above proofs also are performed independently of the original CEGAR loop algorithm because it uses assumptions on correctness of it. In other words, these proofs are performed based on modularity scheme.

## 5 PROTOTYPE SYSTEM

Figure 15 depicts the overview of our prototype system. We use RMI framework on Java for communication between

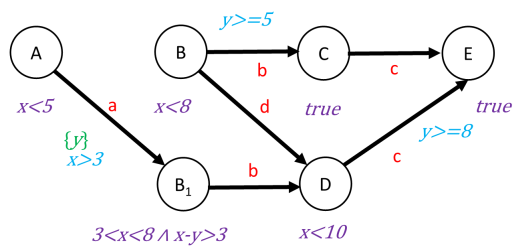



Figure 13: Timed Automaton Refined with $B$-$C$ Transition

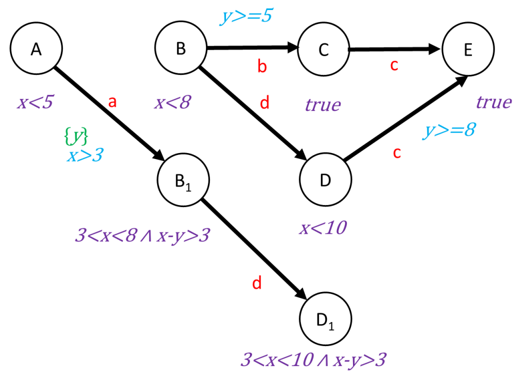



Figure 14: Timed Automaton Refined with $D$-$E$ Transition

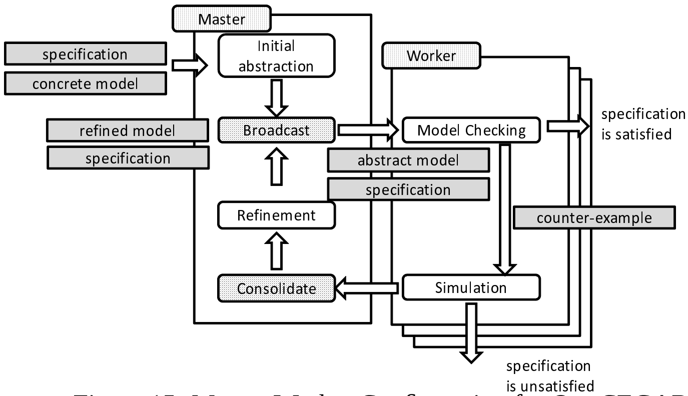



Figure 15: Master-Worker Configuration for Our CEGAR

MutltipleRefinement (revised)
Inputs $\mathscr{A}_i, S_i$
Output $\mathscr{A}_{i+1}$

/* $S_i = \{\hat{\rho}_0, \hat{\rho}_1, \cdots, \hat{\rho}_k\}$ */
/* $\hat{\rho}_j$ is a counter example produced by worker $j$ */
$\mathscr{A}_{i+1} := \mathscr{A}_i$
**for** $\hat{\rho} : S_i$ **do**
  $\mathscr{A}_{i+1}$ :=Refinement$(\mathscr{A}_{i+1}, \hat{\rho})$
**end for**
**return** $\mathscr{A}_{i+1}$

Figure 16: Algorithm 2': MultipleRefinement Algorithm (Revised)

the master and workers. Each worker performs Model Checking and Simulation for its assigned abstract model.

For efficiency, we introduce a modified algorithm, Algorithm 2' shown in Fig. 16.

The major differences between Algorithms 2 and 2' is that Algorithm 2' does not check the executability. It improves the efficiency. However, it means that Algorithm 2' might perform Refinement using pseudo counter-example information. Such a situation, however, does not occur because Algorithm 1 reconstructs $succ\_list = \langle(l_0, D_0), (l_1, D_1), \cdots, (l_k, D_k)\rangle$ before it transforms the timed automaton. $succ\_list$ is a feasible path with regard to the counter-example. Therefore, the counter-example is not executable if and only if $succ\_list$ is an empty list. If $succ\_list$ is empty, no transformation is performed. Consequently, Algorithm 2' also works correctly.

In our implementation, each worker performs model checking and simulation in the same cycle. This invent reduces cost of exchange of data among model checking and simulation steps.

The abstract model is the same among workers. Thus, we have to give different parameters to workers in order for each worker to generate different counter-examples.

As described after we use two strategies to generate counter-examples: shortest traces and the fastest traces. For both of the shortest traces and the fastest traces, the following parameter is used to generate different counter-examples. There might be many shortest (fastest) counter-examples. Among them, what counter-example is chosen by the worker can be a parameter. In order to select different counter-example, we use worker id and random selection for the selection. Of course, if the number of worker is less than that of shortest counter-examples, then some workers might choose the same counter-example.

# 6 EXPERIMENTS

## 6.1 Overview

We have performed experiments using two typical examples. One is Fischer's mutual exclusion protocol. Several $p$ processes with the same shape of an automaton share a critical section. Mutual exclusion is established in a protocol using clock variables. Therefore, it is a typical symmetric structure.

Another one is Gear Controller [17]. It is a model consisting of an engine, a gearbox, a human interface, a gear controller, and a clutch. It is a parallel system of hetero six components.

Before applying our tool, we need to obtain a single timed automaton presentation of Fischer's protocol (and Gear controller) since our proposed method cannot deal with a network of timed automata, which is used in UPPAAL verifier in general.

We performed the experiments under the following environment.

Master
  CPU: Intel(R) Core™ 2 Duo
    CPU L7700 1.80GHz
  MM: 2.00GB

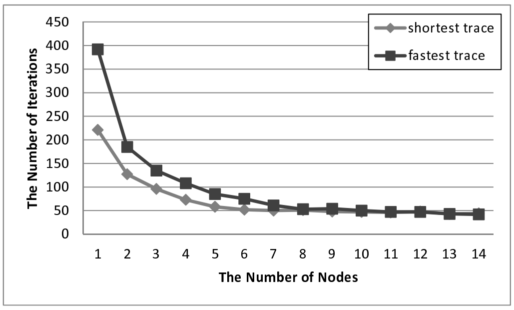



Figure 17: The Number of Iterations : Fischer's protocol

OS: Ubuntu 10.0.4

Workers (14 cpus)

CPU: Dual Core AMD Opteron™ Processor 2210 HE 1.80GHz

MM: 6.00GB

OS: CentOS 5.4

The purposes (research questions) of the experiments are as follows.

1. How efficiently our proposed method works?
2. Are there any difference between:
   (a) types of model structures?
   (b) types of counter-examples used in CEGAR?

Research question 1 can be observed from how CPU times and the number of iteration are reduced in increasing the number of workers.

Research question 2(a) can be observed by comparing the two examples.

Research question 2(b) is hard to answer. We, however, compare using two strategies, the fastest trace and the shortest trace. The fastest trace uses multiple counter-examples with smallest time delay. The shortest traces use multiple counter-examples with shortest (in number of steps) traces. There are many strategies on producing counter-examples. UPPAAL, however, only supports the above two options. Therefore, we think it is reasonable that we compare the two options.

## 6.2 Results

As CPU time, we measure the elapsed times for the computation. The results are averages of five trials of the same configurations.

Figures 17 and 18 show the results of the number of iteration. The number of nodes stands for the number of workers.

In both of Fischer's protocol and Gear Controller, the number of iteration decreases according to the number of workers. The shortest trace for Gear Controller has little effect.

Figures 19 and 20 show the results of the CPU times. The performance is improved according to the number of workers, in Fischer's protocol while Gear Controller shows worse behaviors. The fastest trace also loses its acceleration but the shortest trace requires more time from four workers.

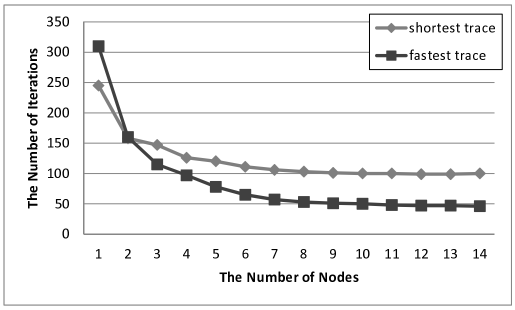



Figure 18: The Number of Iterations : Gear Controller

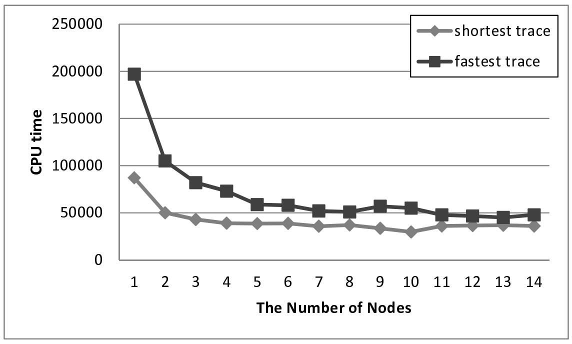



Figure 19: Execution Times : Fischer's protocol

# 7 DISCUSSIONS

We can see that the numbers of iteration are improved in both of the cases, while CPU times are not. This observation supports that our proposed method is potentially effective (w.r.t RQ1). Also w.r.t RQ2(a) and RQ2(b), we find there are some differences.

However, we have to consider the reason why CPU time is not improved. Two possibilities are considered on the results.

One is the following hypothesis: Refinement with multiple counter-examples certainly refines parts of the automaton, however, which are not essential parts of of the automaton for verification of property $p$. Thus, the refinement increases the size of the automaton, which increases CPU time.

The other one is the following hypothesis: The same counter-examples are generated. If some of workers generate the same counter-examples, then the efficiency becomes worse. Such a phenomenon occurs because the random selections do not guarantee that every counter-example is different to others.

Based on the above observations, we have performed the following additional experiments. For the first hypothesis, we have evaluated the number of states. If it increases according to the number of workers, then we can conclude that unnecessary states are generated.

Second we have also evaluated the ratio of unique counter-examples, which is a good index for the second hypothesis.

## 7.1 The Number of States

Figures 21 and 22 show the number of states. Fisher's protocol has gradual increase, while fastest trace of Gear Con-

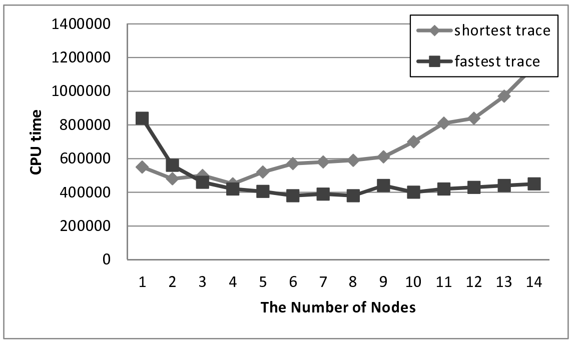



Figure 20: Execution Times : Gear Controller

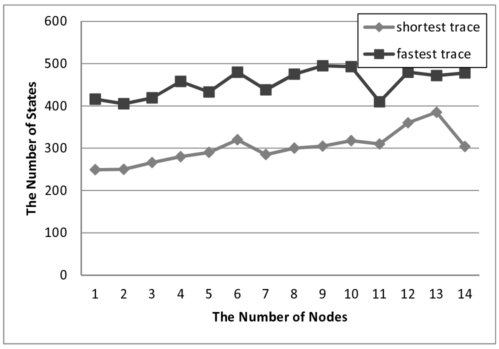



Figure 21: The Number of States : Fischer's protocol

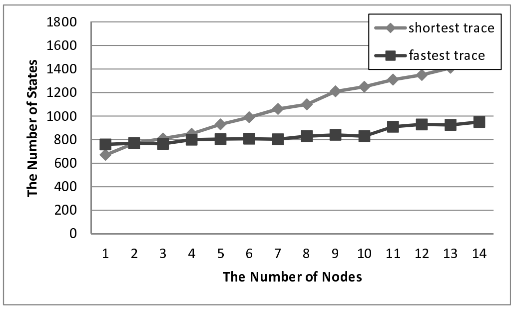



Figure 22: The Number of States : Gear Controller

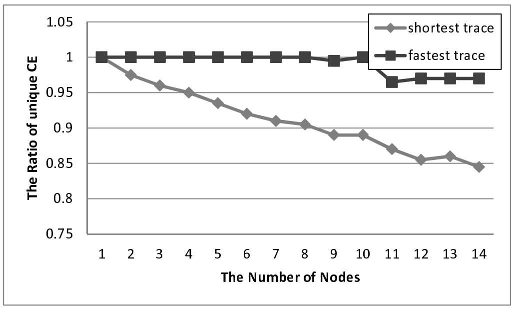



Figure 23: The Ratio of the Same Counter-Example : Fischer's protocol

troller has strong increase.

## 7.2 The Quality of Counter-Examples

Figures 23 and 24 show the ratio of unique counter-examples. If the ratio is equal to 1.0 then it means that every counter-example is different to each other. The shortest traces show that increase of the same counter-examples according to the number of workers.

## 7.3 A Solution

The results support both of the hypotheses. In order to icrease the quality of the counter-examples, priority among the counter-examples is considered. Using the priority, we can control level of the refinement by filtering counter-examples used for refinement. We think, however, that there is no silver bullet, in other words the priority cannot be determined statically and in advance. As an approximate solution, we adopt threshold on the length of counter-examples. The idea is that we only use shorter counter-examples than threshold by the length of the shortest counter-example. From Fig. 19 and 21, we can observe that the shortest trace option is good. Therefore, it is said that the shorter counter-examples are worth to use.

In order to avoid duplication of counter-examples, we think $k$-shortest path algorithm is worth to try. The algorithm is provided by Eppstein [12] and Jiménez [15].

Since UPPAAL uses more sophisticated data structure than DBM which we use and it also uses partial order reduction technique whereas we don't use any further improvements. Therefore, we show the comparison between naïve approach and our approach in order to show the improvements.

We think that the experiments show our approach reduces the number of iteration, which also will improve the size of states of abstraction models. The proposed method works better than naïve CEGAR loop does. It is because the proposed method can deal with larger system than the naïve CEGAR, in some cases. The CPU time is also improved. It implies that the main idea that we simultaneously apply the multiple counter-examples will improve the performance because it reduces the number of iteration. We also have to find further improvements such as detecting redundant counter-examples and reducing applies of counter-examples which do not contribute to refinement.

As a conclusion we can say that the main idea that we simultaneously apply the multiple counter-examples will improve the performance, however, there is some room to improve the performance.

# 8 CONCLUSION

## 8.1 Summary

This paper proposed a CEGAR loop for timed automata where multiple counter-examples are simultaneously applied. This device strongly reduces the number of iteration loops. The experiments show the promising results. Also we have obtained a candidate criterion for more effective multiple CE-GAR.

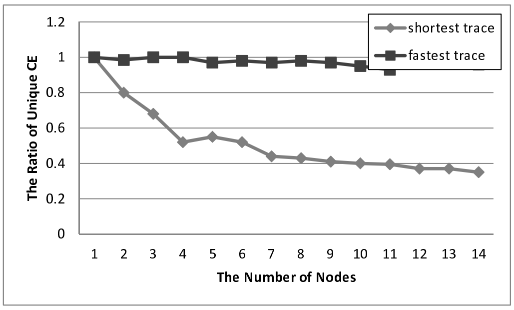



Figure 24: The Ratio of the Same Counter-Example : Gear Controller

## 8.2 Future Work

It is a good idea that if the model becomes too large against to a reasonable CPU time deadline, we reconstruct the model using a subset of the previous set of the counter-examples. Such a scheme can control the size of the abstract model finer.

Another idea of future work will be finding effective criteria for filtering better multiple counter-examples. We also want to try the idea that utilizing modular checking provided in paper [13] and to reconstruct our method based on approach in [18]. Extension of the class of the property is also considered. For example, we want to try to provide CEGAR loop for some subset TCTL [2].

## ACKNOWLEDGMENTS

This work is partially being conducted as Grant-in-Aid for Scientific Research S (25220003), C (26330092) and also C (16K00094).

## References

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

[2] R. Alur, C. Courcoubetis, and D. L. Dill, “Model-checking for real-time systems,” in Proceedings of the 5th Annual Symposium on Logic in Computer Science, pp. 414-425, IEEE (1990) .

[3] R. Alur, T. Dang, and F. Ivancic, “Counter-example guided predicate abstraction of hybrid systems,” in Proceedings of Tools and Algorithms for the Construction and Analysis of Systems TACAS 2003, pp. 208-223 (2003) .

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

[5] P. Bouyer, F. Laroussinie, and P.-A. Reynier, “Diagonal constraints in timed automata: Forward analysis of timed systems,” in FORMATS’05, Lecture Notes in Computer Science, Vol. 3829, pp. 112–126 (2005) .

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

[7] E. M. Clarke, A. Fehnker, Z. Han, J. Ouaknine, O. Stursberg, and M. Theobald, “Abstraction and counterexample-guided refinement in model checking of hybrid systems,” *International Journal of Foundations of Computer Science*, Vol. 14, No.4, pp. 583-604 (2003a) .

[8] 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 (2003b) .

[9] A. E. Dalsgaard, R. R. Hansen, K. Y. Joergensen, K. G. Larsen, M. C. Olesen, P. Olsen, and J. Srba, “opaal: A lattice model checker,” in Proceedings of the 3rd NASA Formal Methods Symposium (NFM’11), Lecture Notes in Computer Science, Vol. 6617, pp. 487–493 (2011) .

[10] A. David, J. Hakansson, K. G. Larsen, and P. Pettersson, “Model checking timed automata with priorities using dbm subtraction,” in Proceedings of the 4th International Conference on Formal Modelling and Analysis of Timed Systems, pp. 128-142 (2006) .

[11] H. Dierks, S. Kupferschmid, and K. G. Larsen, “Automatic abstraction refinement for timed automata,” in Proceedings of the 5th International Conference on Formal Modelling and Analysis of Timed Systems, Vol. 4763, pp. 114-129 (2007) .

[12] D. Eppstein, “Finding the k shortest paths,” in 35th Annual Symposium on Foundations of Computer Science, pp. 154–165 (1994) .

[13] F. He, H. Zhu, W. N. N. Hung, X. Song, and M. Gu, “Compositional abstraction refinement for timed systems,” in Proceedings of 2010 Fourth International Symposium on Theoretical Aspects of Software Engineering, pp. 168-176 (2010) .

[14] H. Hermanns, B. Wachter, and L. Zhang, “Probabilistic cegar,” in Computer Aided Verification, Lecture Notes in Computer Science, Vol. 5123, pp. 162-175 (2008) .

[15] V. M. Jimènez and A. Marzal, “Computing the k shortest paths: A new algorithm and an experimental comparison,” in Algorithm Engineering 1999, Lecture Notes in Computer Science, Vol. 1668, pp. 15–29 (1999) .

[16] S. Kemper and A. Platzer, “Sat-based abstraction refinement for real-time systems,” in Proceedings of the Third International Workshop on Formal Aspects of Component Software, Vol. 182, pp. 107-122 (2006) .

[17] M. Lindahl, P. Pettersson, and W. Yi, “Formal design and analysis of a gear controller: An industrial case study using uppaal,” in Lecture Notes in Computer Science, Vol. 1384, pp. 289-297 (1998) .

[18] M. O. Mollera, H. Rueß, and M. Soreab, “Predicate abstraction for dense real-time systems,” *Electronic Notes in Theoretical Computer Science*, Vol. 65, No.6, pp. 218–237 (2002) .

[19] 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) .

[20] F. Wang, K. Schmidt, G. D. Huang, F. Yu, and B. Y. Wang, “Formal verification of timed systems: A survey and perspective,” in Proceedings of the IEEE, Vol. 92, No.8, pp. 1283-1307 (2004) .

(Received September 30, 2015)
(Revised April 13, 2016)

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

**Takeshi Nagaoka** received the MI, DI degrees from Osaka University in 2007 and 2010, respectively. He currently works for Toshiba Solutions Corporation. His research interests include abstraction techniques in model checking, especially a timed automaton and a probabilistic timed automaton.

**Toshiaki Tanaka** received the BE, MI degrees from Kobe University in 2009 and from Osaka University in 2011, respectively. He currently works for Sony Corporation. His research interests include parallelization of model checking, especially a timed automaton.

**Toshifusa Sekizawa** received his MSc degree in physics from Gakushuin University in 1998, and Ph.D. in information science and technology from Osaka University in 2009. He previously worked at Nihon Unisys Ltd., Japan Science and Technology Agency, National Institute of Advanced Industrial Science and Technology, and Osaka Gakuin University. He is currently working at College of Engineering, Nihon University. His research interests include model checking and its applications.

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

Refinement
Inputs $\mathscr{A}_i, \hat{\rho}$
Output $\mathscr{A}_{i+1}$

/* $\hat{\rho} = l_0 \xrightarrow{a_1,g_1,r_1} l_1 \xrightarrow{a_2,g_2,r_2} \cdots \xrightarrow{a_n,g_n,r_n} l_n (l_n = e)$ */
/* $succ\_list = \langle (l_0, D_0), (l_1, D_1), \cdots, (l_k, D_k) \rangle$,
where $(l_j, D_j)$ represents the $j$-th reachable state set along with $\hat{\rho}$, and $l_k$ is the last location reachable from the initial state. */
$succ\_list := \mathrm{tr}(\hat{\rho})$
/* function tr () obtains $succ\_list$ form $\hat{\rho}$ */
$\mathscr{A}_{i+1} := \mathscr{A}_i$
**for** $j := succ\_list$.length **downto** $1$ **do**
  $e_j := (l_{j-1}, a_{j-1}, g_{j-1}, r_{j-1}, l_j)$
  $\mathscr{A}_{i+1} := \mathrm{Duplication}(\mathscr{A}_{i+1}, succ\_list_j, e_j)$
    /* Duplication of the Location and Transitions */
  **if** $\mathrm{IsRemovable}(\mathscr{A}_{i+1}, succ\_list_j, e_j)$ **then**
    $\mathscr{A}_{i+1} := \mathrm{RemoveTransition}(\mathscr{A}_{i+1}, e_j)$
      /* Removal of Transitions */
    **break**
  **else if** $j = 1$ **then**
    $\mathscr{A}_{i+1} := \mathrm{DuplicateInitialLocation}(\mathscr{A}_{i+1}, (l_0, D_0))$
      /* Duplicate the initial location and transitions from the initial location */
  **end if**
**end for**
**return** $\mathscr{A}_{i+1}$

Figure 25: Algorithm 1: Refinement Algorithm for a counter-example

# Appendix A

Figure 25 shows the algorithm of Refinement, Algorithm 1.
Algorithm 1 uses a counter-example $\hat{\rho}$ and generates a refined timed automaton. It uses functions, Duplication(), RemoveTransition(), and DuplicateInitialLocation(). Functions Duplication(), RemoveTransition() and DuplicateInitialLocation() are functions to duplicate locations and transitions, to remove unnecessary transactions, and to duplicate the initial location, respectively. For the definitions of these functions, please refer [19].

# 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