International Journal of Fuzzy Logic and Intelligent Systems, vol. 6, no. 2, June 2006 pp. 167-172

Systematic Elicitation of Proximity for Context Management

Chang Suk Kim*, Sang-Yong Lee, Dong Cheul Son*****

***Dept. of Computer Education, Kongju National University, Chungnam, Korea**

****School of Computer Engineering, Kongju National University, Chungnam, Korea**

*****Dept. of ICE, Baekseok University, Chungnam, Korea**

Abstract

As ubiquitous devices are fast spreading, the communication problem between humans and these devices is on the rise. The use of context is important in interactive application such as handheld and ubiquitous computing. Context is not crisp data, so it is necessary to introduce the fuzzy concept. The proxity relation is represented by the degree of closeness or similarity between data objects of a scalar domain. A context manager of context-awareness system evaluates imprecise queries with the proximity relations. In this paper, a systematic proximity elicitation method are proposed. The proposed generation method is simple and systematic. It is based on the well-known fuzzy set theory and applicable to the real world applications because it has tuning parameter and weighting factor. The proposed representations of proximity relation is more efficient than the ordinary matrix representation since it reflects some properties of a proximity relation to save space. We show an experiments of quantitative calculate for the proximity relation. And we analyze the time complexity and the space occupancy of the proposed representation method.

Key words : Proximity relation, Context, Ubiquitous device, Fuzzy set, Similarity, Database system

1. Introduction

As ubiquitous devices are fast spreading, the communication problem between humans and these devices is on the rise. The use of context is important in interactive application such as handheld and ubiquitous computing. To manager a number of context, a context management technique is needed. Context is not crisp data, so it is necessary to introduce the fuzzy concept[1, 2, 3].

The relational data model has been extensively studied and widely used because of its simplicity in designing databases and its clarity based on mathematical background. This relational data model usually takes care of only well-defined data and precise queries. However, in real world applications data and queries are often partially known or imprecise rather than specific. A specific query establishes a rigid qualification and it is concerned only with data that match it precisely.

For example, "Find members whose technical interest is internet" is specific query. If the members do not exist, null value should be returned. On the other hand, an imprecise query establishes a target qualification and is concerned with data that are similar to this target. For example, consider "Find members whose technical interest is close to internet" If there is none, members that have a technical interest similar to internet such as WWW or web are returned. Because conventional relational database systems cannot deal with imprecise queries, Because conventional relational database systems cannot deal with imprecise queries, they are forced to retry a particular (specific) query repeatedly until it matched satisfactory data. If user is not aware of any close alternatives, then this solutions is infeasible.

The above example shows that it is necessary to elicit a quantitative value of proximity or closeness between two data objects of a scalar domain. A fuzzy database system evaluates imprecise queries with the proximity relations. One of the obstacles to building practical fuzzy database system is to acquire semantic data such a proximity relation. Most of all the fuzzy database researchers assume that proximity relations are already givens. Only few of researchers have considered to systematically generated proximity relations up to now[12]. We need a systematic elicitation of degrees of proximity for building a practical fuzzy database system.

The fuzzy database system frequently accesses the proximity relation which is matrix forms of degrees of proximity. Thus, the other problems is to represent the proximity relation in computer efficiently. The matrix representation in numerical applications is not suitable for proximity relation of fuzzy databases.

In this paper, we propose a systematic method to elicit degrees of proximity and efficient representation in numerical applications is not suitable for proximity relation of fuzzy databases. The proposed proximity generation method is simple and systematic since it is based on the well-known fuzzy set theory. The method is applicable to the real world applications with tuning parameter.

The proposed representations of proximity relation are more efficient than the ordinary sparse matrix representation since

Manuscript received Jun. 3, 2006; revised Jun. 19, 2006.
This work was supported by grant No.(B-1220-0501-0079) from MIC of Korea.

the representation methods reflect some properties of the proximity relation i. e. reflexive, symmetric sparse in most case. We analyze the time complexity and the space occupancy of the representations.

2. Extension of RDB for Context Manager

Context is not crisp data, so it is necessary to introduce the fuzzy concept. To manager a number of context, a context management technique is needed. Several extensions of the classical relational model have been proposed in order to deal with fuzzy query [5, 6, 7, 8, 9]. Fuzzy relational data model can be classified into two approaches i. e. Crisp Data and Fuzzy Query (CDFQ) and Fuzzy Data and Fuzzy Query (FDFQ) approaches. The CDFQ approach precesses queries with fuzzy concepts and crisp data values that are stored in the database.

Fuzzy relational data items need to have compatibility with conventional relational database systems for practical applications. Even though the FDFQ approach can much enhance database expressional power, they are not compatible with conventional database systems. In view of actuality, the FDFQ approach is far from commercial database systems. We consider the CDFQ approach that can enhance database functionalities significantly and can be compatable with conventional databases.

The simplest approach is to extent an ordinary relation to a fuzzy relation by adding grades of membership. Thus, a fuzzy database Df is defined by a set of fuzzy relations

$$Df = \{R_1, R_2, R_3, ..., R_n\}$$

where Ri is a fuzzy relation characterized by membership function

$$\mu_{Ri} : U_{i1} \times U_{i2} \times U_{i3} \times ... \times U_{in} \rightarrow [0, 1]$$

where U_{ij} is the domain of the j-th attribute of relation R_i.

For an example of fuzzy database shown in Fig.2.1.Consider the query "Retrieve the subjects similar to internet?". The ordinary relational database systems can not retrieve any data value but null value. But, fuzzy database systems list the close to database, Web(0.9) and Electronic Commerce(0.7). Here, we can see that fuzzy database systems use proximity relation for finding the closest data to given query.

Our CDFQ model consists of Crisp DataBase(CDB) and Semantic Data(SDB). The CDB is general relation of conventional relational databases. The SDB is a collection of relations or functions such as proximity relations or fuzzy set definitions. The SDB is used to explain how to compute the degree of compliance of a given data value with a user's query. This SDB is used to explain how to compute the degree of compliance of a given data value with a user's query. This SDB can be used reflect the subjective knowledge of a user without changing the CDB. The CDB and the SDB have different data structure, because of the efficiency of fuzzy query processing.

Subject	Lecturer	μ
Internet	Kim	1.0
Electronic Commerce	James	0.7
Web	Suzan	0.9
Operating System	Henry	0.3

(a) An example of fuzzy relation

	Internet	EC	Web	OS	DB
Internet	1.0	0.7	0.9	0.2	0.6
EC	0.7	1.0	0.6	0.1	0.5
Web	0.9	0.6	1.0	0.2	0.4
OS	0.2	0.1	0.2	1.0	0.3
DB	0.6	0.5	0.4	0.3	1.0

(b) An example of proximity relation

Fig. 2.1 An example of fuzzy database

Shenoi and Melton[11] proposed proximity relation instead of similarity relation by Buckles and Petry[10]. A proximity relation by Buckles and Petry[10]. A proximity relation is defined in the following manner.

Definition 2.1 A proximity relation is a mapping, s_j : $D_i \times D_j$ [0, 1], such that for $x, y \rightarrow D_j$,

(i) $s_j(x, x) = 1$ (refexivity),

(ii) $s_j(x, y) = s_j(y, x)$ (symmetry)

The properties of reflexity and symmetry are appropriate for expressing the degrees of "closeness" or "proximity" between elements of a scalar domain. Moreover, proximity relations include the similarity relations.

3. Elicitation for Proximity Relation

3.1 Elicitation function of the degrees of proximity

The degree of proximity is a quantitative value of 'closeness' or 'similarity' between two data objects of a scalar domain. The degrees of proximity can be represented by a matrix form (fuzzy binary relation) in a given domain. The fuzzy binary relation that is reflexive and symmetric is usually called a proximity relation [11, 12]. we derive a generation function of the degrees of proximity which is based on the measurement theory of fuzziness [13].

- **ızzification of crisp data, categories and feature values**

Data in relational data model have crisp and atomic properties. There is no meanings or semantics between data objects of a scalar domain. It is impossible to measure closeness between a crisp data and . We can only compare with by means of ASCII codes not the meanings. To acquire a degrees of proximity between the crisp data objects, we should assign properties to the data objects, we should assign properties to the data objects of a domain in an aspect. Two definitions are needed to derive a generation function of the degrees of proximity.

Definition 3.1 An aspect of a domain can be decomposed into sub-properties. Category is a sub-property of the aspect and is represented as follows

x_i (where, 1 ≤ ≤ l, l is finite integer)

Definition 3.2 A feature value is constant which can be assigned in a category. A feature value is represented as

$\mu\, x_i) \in 0, 1]$

In order to get the degree of closeness between given data objects, we should assign a set of feature value to each data. We can discriminate a data θ_1 with a data θ_2 by its feature values. Prior to assign feature values on data, a domain should be divided into categories in an aspect. Categories and feature values of data object 'pitcher' are depicted in Fig. 3.1 Data object is defined as follows

$$\theta = \sum_{i=1}^{l} \mu\ ;i)/xi$$

where is a sequence of categories and is feature values. The is number of fixed length of categories.

For example, there are five specialized contexts in a living room. Various degrees of proximity exist between pair of the context domain has a number of data as follows

Θ = {standing, watching, reading, walking, talking}
x_i = {who, what, where, when, why}
θ_1(standing) = $\mu_1/x_1 + \mu_2/x_2 + \mu_3/x_3 + \mu_4/x_4 + \mu_5/x_5$

Context categories	Categories values	Feature values
who	kim	1.0
what	watching	0.8
where	living room	0.9
when	pm 12:00	0.7
why	relax	0.8

Fig. 3.1 An example of context categories

The acquisition method of feature values is similar to that of fuzzy set. It is not described here because it is out of scope in this paper.

- **:rivation of the generation of the function**

The principle of measure of fuzziness is applied to measure a degree of proximity. First of all, a domain is extended to two dimensional relation $\Theta \times \Theta$. A generation function of a degree of proximity p is defined as follows:

$$p: \Theta \times \Theta \rightarrow [0, 1] \quad (1)$$

The degrees of proximity is large if sum of the distance of feature values of the two data items is small.

$$\sum_{x\in X}[\mu_{\theta i}(x)-\mu_{\theta j}(x)] \geq \sum_{x\in X}[\mu_{\theta i}(x)-\mu_{\theta k}(x)]$$
$$\text{if and only if } p(\theta_i, \theta_j) \leq p(\theta_i, \theta_k) \quad (2)$$

Eq.for finding distance of feature of feature value is defined by Eq.(3), $\delta_{\theta i,\theta j}(x)$ is distance of one feature value between θ_i and θ_j.

$$|\mu_{\theta i}(x)-\mu_{\theta j}(x)| = \delta_{\theta i,j}(x) \quad (3)$$

All the feature values belongs to one data object as Eq.(4).

$$\text{dis}(\theta_i, \theta_j) = \delta_{\theta i,\theta j}(x) \text{ (here, } x \in X) \quad (4)$$

To introduce two data objects (θ_H, θ_L) is needed to generalize Eq. (4). All the feature value of the data θ are I and all the feature value the data are θ . The distance between the two data objects (θ_H, θ_L) minus a distance between any two data objects (θ_i, θ_j) is just degree of proximity as following

$$p(\theta_i, \theta_j) = \text{dis}(\theta_H, \theta_L) - \text{dis}(\theta_i, \theta_j) \quad (5)$$

The range of P(θ θ is as follows.

$$0 \leq p(\theta_i, \theta_j) \leq \text{dis}(\theta_H, \theta_L) \quad (6)$$

The normalized forms of Eqs. (5) and (6) which have degree of proximity from 0 to 1 are defined Eqs. (7) and (8).

$$p(\theta i, \theta j) = 1 - \frac{dis(\theta_i, \theta_j)}{dis(\theta_H, \theta_L)} \quad (7)$$

$$0 \leq p(\theta_i, \theta_j) \leq 1 \quad (8)$$

dis(θ_H, θ_L) can be rephase as |X| (number of feature values). To get the Eq. (9), the Eq. (4) and |X| is inserted into Eq. (7)

$$p(\theta i, \theta j) = 1 - \frac{\sum_{x\in X}\delta_{\theta i,j}(x)}{|X|} \quad (9)$$

Inserting the Eq.(3) into Eq.(9) results in the Eq.(10)

$$p(\theta i, \theta j) = 1 - \frac{\sum_{x\in X}|\mu_{\theta i}(x) - \mu_{\theta j}(x)|}{|X|} \quad (10)$$

Finally, we get the generalized function to elicitate degree of proximity.

$$p_q(\theta i, \theta j) = 1 - \frac{[\sum_{x\in X}|\mu_{\theta i}(x) - \mu_{\theta j}(x)|^q]^{1/q}}{|X|} \quad (11)$$

The variable q ∈ 1, ∞ we called it tuning parameter, is used to tune a interval of degree of proximity.

Eq. (10) is a special case of Eq.(11) in q=1. As q increases, the interval of degrees of proximity decreases and vice versa. It is required to tune q for the real world applications.

3.2 Experiments

The procedure to generate proximity relation is as follows. Assuming that the input is a domain Θ)f the ordinary relation which has θ_i (1 ≤ ≤ ı). First, the categories of the given input domain Θ s defined and the feature values for each data object θ_i is assigned in an aspect. Next, The degrees of proximity between each feature value θ_i and θ_j ($1 \leq i \leq n$) is calculated. Result of the procedure are degrees of proximity between data objects of the given domain.

case 1 : Consider data objects θ_1 (standing), θ_2 (walking) and

θ_3 (watching) which have five feature values are given as follows

θ_1 = {μ_1=1.0, μ_2=0.7, μ_3=0.9, μ_4=0.9, μ_5=0.8},
θ_2 = {μ_1=0.8, μ_2=1.0, μ_3=1.0, μ_4=0.9, μ_5=0.9},
θ_3 = {μ_1=0.6, μ_2=1.0, μ_3=0.4, μ_4=1.0, μ_5=0.5}

Let us find the degree of proximity $\mathbf{p}(\theta_1, \theta_2)$ and $\mathbf{p}(\theta_1, \theta_3)$ with the tuning parameter q=1.

To calculate the degree of proximity of between θ_1 and θ_2 (θ_3), we apply the feature values of θ_1 and θ_2 (θ_3) and to the Eq. (11) with the tuning parameter q=1. Then, the results are as followings.

c

The above result gives $p(\theta_1, \theta_2)$ that is more similar than $p(\theta_1, \theta_3)$. On the other words. θ_1 (standing) is similar to θ_2 (walking), but θ_1 (standing) is less similar to θ_3 (watching).

case 2 : Consider the case 1, with q=2, q=3 and q=4. An effect of the tuning parameter q is shown.

The degrees of proximity of between and is calculated by Eq. (11) with q=2, q=3 and q=4.

i) q = 2:
$\mathbf{p}(\theta_1, \theta_2)$ = 0.92 $\mathbf{p}(\theta_1, \theta_3)$ = 0.85

ii) q = 3:
$\mathbf{p}(\theta_1, \theta_2)$ = 0.93 $\mathbf{p}(\theta_1, \theta_3)$ = 0.88

iii) q = 4:
$\mathbf{p}(\theta_1, \theta_2)$ = 0.94 $\mathbf{p}(\theta_1, \theta_3)$ = 0.89

We show that as q increase, then the interval of $\mathbf{p}(\theta_1, \theta_2)$ and $\mathbf{p}(\theta_1, \theta_3)$ decrease.

4. Efficient Representation of Proximity Relation

Proximity relation can be represented by a matrix form, which are frequently accessed by fuzzy database systems for imprecise query processing. The proximity relation for fuzzy databases have the following properties.

1) reflexive
2) symmetric
3) row and column indices are data value which are not integer but can be enumerated
4) may be sparse matrix
5) needs random access for each elements in the proximity relation.

Thus, it is not suitable to use the representation of sparse matrix in numerical applications. In this section, We describe some efficient data structures and algorithms for retrieving proximity relation [14,15].

4.1 Efficient Representation and Algorithms for Proximity Relation

A fuzzy proximity relation usually represented by a matrix as Fig. 4.1(a) where is a set of data items in a given domain. The ordinary representation of matrix in computer science is a two-dimensional array defined as a[ROWS][COLS]. With this representation, we can locate quickly any element by writing a[i][j], where i is the row index and j is the column index.

P	a	b	c	d	e	f
a	1.0	0.7	0.0	0.9	0.0	0.0
b	0.7	1.0	0.5	0.2	0.0	0.0
c	0.0	0.5	1.0	0.0	0.4	1.0
d	0.9	0.2	0.0	1.0	0.0	0.0
e	0.0	0.0	0.4	0.0	1.0	0.4
f	0.0	0.0	1.0	0.0	0.4	1.0

Fig. 4.1 Representation of R(Θ Θ

However, there are some problems in the ordinary matrix representation because of wasting of space. Especially, in case Θ is a large set, amount of memory space is wasted, since it may be a sparse matrix. As shown in Fig. 4.1, Proximity relation is symmetric and a diagonal elements of this matrix are 1. Thus, we may store only the upper triangular of **R(**Θ Θ The two efficient representation of a fuzzy proximity relation **R(**Θ Θ are proposed. The first is triple array <row index, column index, value>in each column and the second is a single array containing only <value>.

- **·iple array**

The triple array stores triples <row index, column index, value(**p**)>for a degree of proximity **p**. It stores only non-zero degrees of proximity. The Fig. 4.2 shows that Fig. 4.1 is represented by the triple array.

	a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]
row index	a	a	b	b	c	c	e
column index	b	d	c	d	e	f	f
P	0.7	0.9	0.5	0.2	0.4	1.0	0.4

Fig. 4.2 The triple array representation of R(Θ Θ

The proximity retrival algorithm using the triple array appeares in Algorithm 1.Input query (pair of data item) should be transferred lexicographical order if it is not ordered (step 1). And we search **p** using binary search method (step 2).

Algorithm 1: Proximity retrieval algorithm in triple array
input : pair of domain data elements (θ_i, θ_j)
output : degree of proximity **p**

```
first := a[1].c   r
last := a[m].c   r
step 1. if ((θ  < θ  ) then (θ   θ   := (θ   θ  ;
step 2. found := false;
     while first ≤  ast and not found do
       index :=  ⌊    ·st + last) / 2 ⌋
       if p = TA[index] then found := true
                                   // TA ·triple array
       else if p < TA[index] then last := index − 1
       else first := index + 1
       end;
     end;
     if not found then p := 0 end;
```

The size of the triple array representation will be $(m+1)*3$ where m is the number of non-zero **p** in the upper triangular matrix of the Fig. 4.1and n is the number of data elements in θ The time complexity of the triple array's retrieval algorithm is

$$O(\log m).$$

This method is good when number of data elements in θ is fairly large and a lots of zero-element are existed.

4.2 Evaluation

The single array is always better than the triple array in retrieval time. We analyze which method is better space occupancy of the single array and the triple array. To find when the triple array is better than the single array in space cost, we solve for inequality,

$3(m+1) < n(n-1) / 2$.

After simplification we get

$$m < n(n-1) / 6 - 1. \quad (12)$$

We get the m interesting integer in n= 1,2,...,1000 into formula (13). In the above calculation, the number of non-zero data items is calculated by m dividing the number of upper triangular data items. We conclude the triple array has better space occupancy has less than about 33%. The single array has always O(1) retrieval time.

Given a proximity relation **R**(θ θ which have n = 100 data items and m-1024 non-zero items in upper triangulsr. Find the each space cost and retrieving time cost of the triple array and the ordinary matrix representation (n*n)?

(a) Triple array: Space cost STA = 3(1024 + 1) = 3075.
Time cost TTA = log 1024 = 10.
(b) Ordinary array: Space cost SOA = 102 = 10000.
Time cost TOA = 1.

We can see that the triple array has good space occupancy if m=1,024(i.e. the percent of non-zero data items is 21% (1024/1950 =21%)). But it has a little worse retrieving time. The ordinary array always worse than the single array.

Table 4.1 Comparision of proximity relation representation methods

m: number of non-zero elements in upper triangular matrix
n: number of data object in a domain θ

method	retrieving time	space occupancy
Triple array	O(log m)	3(m + 1)
Ordinary array	O(1)	n^2

In Table 4.1, the triple array, the single array and the ordinary array representation methods are compared.

5. Conclusions

The main purpose of this paper is to show a elicitation method of the proximity relation based on the fuzzy set theory. This method is more quantitative and systematic than an intuitive method because the proposed method calculates degrees of proximity by measuring distances of feature values. We have shown the quantitative calculation of the degree of proximity between 'standing' and 'walking' context through the experiments. It is considerably useful when a lots of data objects are given in fuzzy relational database domains.

We also studied efficient representation and retrieval algorithms have been proposed, which is better than an ordinary matrix representation. We have shown the time complexity and the space occupancy of the proposed representations. And we have analyzed different properties of the proposed method. Future research will concentrate on evaluation of the proposed generation method through experiments of real world applications.

References

[1] S. Jang and W. Woo, "Uni-UCAM: A unified context-aware application model," *Lecture Note Artificial Intelligence*, Vol. 2680, pp. 178-189, 2003.

[2] A. Dey, "Towards a better understanding of context and context-awareness," *CHI 2000 Workshop on the What, who, Where, When and How of Context-Awareness*, 2000.

[3] B. Schilit, N. Adams and R. Want, "Context-Aware computing Applications," *1st International Workshop on Mobile Computing Systems and Applications*, pp. 85-90, 1994.

[4] E. Codd, "A relational model for large shared data banks," *Comm. of the ACM*, Vol. 13, No. 6, pp. 377-387, 1970.

[5] S. C. Park, C. S. Kim and D. S. Kim, "Fuzzy querying in relational database," *Fifth IFSA World Congress*, pp. 533-536, 1993.

[6] D. H. Lee and M. H. Kim, "A fuzzy relational data model and extended semantics of relational operations," *Proceedings InfoScience 93*, pp. 275-281, 1993

[7] M. Umano and S. Miyamoto, "Recent development of fuzzy database systems and applications," *Fifth IFSA World Congress*, pp. 537-540, 1993

[8] T. Ichikawa and M. Hirakawa, "ARES: A relational database with the capability of performing flexible interpretation of queries," *IEEE Trans. on Software Engineering*, Vol. SE-12, No. 5, pp. 624-634, 1986.

[9] A. Motro, "VAGUE: A user interface to relational database that permits vague queries," *ACM Trans. on Office Information System*, Vol. 6, No. 3, pp. 187-214, 1988.

[10] B. Buckles and F.Petry, "A fuzzy representation of data for relational database," *Fuzzy Sets and Systems*, Vol. 7, pp. 213-226, 1982.

[11] S. Shenoi and A. Melton, "Proximity relations in the fuzzy relations in the fuzzy relational database model," *Fuzzy Sets and Systems*, Vol. 7, pp. 285-296, 1989.

[12] D. H. Lee and M. H. Kim, "Elicitation of semantic knowledge for fuzzy database systems," *Conf. on Korea Information and Science Socieoty*, pp. 113-116, Oct. 1993.
[13] G. Klir and T. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice-Hall International Editions, 1988.
[14] L. Kohout and M. Harris, "Computer representation of fuzzy and crisp relations by means of threaded trees using foresets and aftersets," *Journal of Fuzzy Logic and intelligent Systems*, Vol. 3, No. 1, pp. 41-64, 1993.
[15] E. Horowitz, S. Sahni and Freed, *Fundamentals of Data Structures in C*, Computer Science Press, 1993.

Chang Suk Kim
He received the B.S., M.S. and Ph.D. degrees in Computer Engineering from Kyungpook National University, Daegu, Korea, in 1983, 1990 and 1994, respectively. He was a post-doctoral researcher at University of California, San Diego. He worked for ETRI from 1983 to 1994. At present, he is an Associate Professor at the Department of Computer Education, Kongju National University, since 1998. His research interests include intelligent databases, fuzzy theory and XML based information integration.

Phone : 041-850-8822
E-mail : csk@kongju.ac.kr

Sang-Yong Lee
He received the B.S. degree from the Chungang University, Seoul, Korea in 1984, the M.S. degree in Dept. of System Science, Tokyo Institute of Technology, in 1988, and the Ph.D. degree in Computer Science from Chungang University in 1993. He was a researcher of NEC Research Center in Japan. He has been a Professor at the School of Computer Engineering, Kongju National University, since 1993. His current research interests include artificial intelligent, agent system, ubiquitous computing and context-awareness.

Phone : 041-850-8523
E-mail : sylee@kongju.ac.kr

Dong Cheul Son
He received the B.S., M.S. degrees in Computer Engineering from Kyungpook National University, Daegu, Korea, in 1983 and 1985, respectively. and Ph.D. degrees in Electrinic Egineering from Chungbuk National University, Chungju, Korea, in 2001. From 1983 to 1998, he worked for ETRI. At present, he is an Associate Professor at the Department of Information and Communication Eng., Baekseok University, since 2002. His research interests include AI, fuzzy theory and XML based information integration, Intenet engineering.

Phone : +82-41-620-9536
E-mail : dcson@cheonan.ac.kr