

# Querying Nested Historical Relations in Heterogeneous Databases Environment

Ping-Tsai Chung  
Department of Computer Science  
Long Island University, New York

Hsin-Hua Hsiao  
American Express Corp, New York

**ABSTRACT** – *We study schema integration problems for consolidating historical information from nested relational databases in heterogeneous databases environment. These nested relations are for supporting complex objects. In heterogeneous databases systems, probabilistic partial values have been used to resolve some schema integration problems. In this paper, we extend the concept of probabilistic partial values in database relations to the nested historical relations in heterogeneous databases environment. Our study combines the research in nested relational databases, temporal databases, and corporate data warehousing for database applications. We develop a full set of temporal relational operators over nested historical relational data model.*

*Key words: Information integration, schema integration problems, nested relational databases, temporal databases*

## I. INTRODUCTION

The advance in communication and database technologies has facilitated the modern database system evolvments and developments. In the heterogeneous databases environment, a large family of new *information integration* applications have been proposed. Such applications take data that is stored in two or more databases (i.e., information sources) and build from them one large database, possibly virtual, containing information from all the sources, so that the data can be queried as a unit. The information sources maybe conventional

databases or other types of information, such as collections of Web pages [2].

There are three principle approaches for information integration in heterogeneous databases environment [2]:

1. *Federated databases*. The sources are independent, but one source can call on others to supply information.
2. *Warehousing*: Make Copies of the data sources are stored at a central site ( i.e., *data warehouse* ) and transform it to a common schema.

3. *Mediation*: A mediator is a software component that supports a *virtual database* - a view of all sources, as if they were integrated. The mediator answers a user's view query by translating it to terminology of the sources and querying them.

The schema integration process may present a large number of problems caused by various aspects of semantic discrepancy and design autonomy [6]. The data in the various sources, while having the same meaning, can be represented in many different ways.

1. *Data type conflicts*: This occurs when semantically related data items are represented in different data types.

2. *Name conflicts*: This occurs when semantically related data items are named differently or semantically unrelated data items are named equivalently.

3. *Data scaling conflicts*: This occurs when semantically related data items are represented in different databases using different units of measure.

4. *Value differences*: The same concept might be represented by different constants at different sources.

5. *Missing data or values*: This occurs when relations with different sets of attributes are to be integrated into a global schema.

6. *Missing values*: This occurs when a source might not record information of a type that all or most of the other sources provide.

Each of these inconsistencies among sources requires a form of translation that must be implemented before the integrated database can be built.

In this paper, we study schema integration problems for consolidating historical information from nested relational databases in a data warehouse in heterogeneous databases environment. Conventional Databases in general carry *real time data* (i.e., the most recent data). As the new *up-to-date* information becomes available through updating operations, the existing values are overridden from the database. Such databases are called *real-time databases*, since they only contain current information which is a snapshot of the reality. On the other hand, data warehousing systems are widely used in many large corporations for their future investments by handling high volume of *historical data* in the data warehousing environment.

Temporal database modeling with an object-oriented approach has been discussed in [7] Temporal nested database relations were studied in [8]. The nested relations are for supporting complex objects [1]. Oracle9i database system supports a type of collection object called *nested table*, which are tables within tables [9]. The nested relation schema usually be represented by means of a tree structure, *the Schema Tree*, where the root is an external relation schema, the leaves are simple attributes, and the internal nodes are internal relation schema. Notice the similarity between this hierarchical representation and XML hierarchical (tree) data model, where internal nodes represent complex elements whereas leaf nodes represent simple elements. Note that a semistructured data has been developing for a new data model designed to cope with problems of information integration whereas XML is a standard language for describing semistructured data schemas and representing data [1].

In order to query nested tables, a special *THE* operator must be used to flatten the nested table. Similar *Unnest* operator has been discussed in [14]; this operation replaces a higher order name by its descendants in the next lower level in the schema tree. Repeated application of *Unnest* operation *flattens* a nested relation.

In Section II, we present our model of nested historical relation. In Section III, we extend the concept of probabilistic partial values in database relations to the nested historical relations in heterogeneous databases environment. We discuss the schema integration with probabilistic partial values in nested historical relations. In Section IV, we develop a full set of temporal relational operators over nested historical relational data model. Finally, in Section V, we make conclusions of this paper and discuss future possible directions of this research work.

## II. BASIC CONCEPTS AND ENVIRONMENT

In [14], Chung and Hsiao presented a generalized nested historical relational model for handling time variation of complex objects in heterogeneous databases environment. In this Section, we follow some terminologies for that model of nested historical relation defined in [14].

### *Nested Historical Relation*

In the *basic relational database model* (also called *flat relational database model*), attributes are required to be single-valued and to have atomic domains. The nested relational database model allows composite and multivalued attributes, and thus is also

known as the *Non-1NF* or *Non-First Normal Form (NFNF) relational database model*. It supports complex tuples with a hierarchical structure, and is useful for representing objects that are naturally hierarchically structured. In Figure 1, it shows a nested relational database model – DEPT Schema (Actually, it is a nested historical relational database model) and an example of a Non-1NF tuple in DEPT (see Figure 2). The attributes of a nested historical relation are hierarchically organized as a tree which is called a *schema tree*. In Figure 3, it shows a Schema Tree for the DEPT Schema

Now, let's define some terms of the concept of temporal set [8] [1]. Let  $T$  be the set of time points mapped to natural numbers, i.e.,  $\{0, 1, 2, \dots, n\}$  which is well ordered under less-than-or-equal-to relationship. 0 is the relative beginning point. The symbol  $n$  denotes the present time instant and its value increments as the clock ticks (time advances). *We do not consider a time unit*. It is user defined and can be any of seconds, minutes, hours, days, etc. A *time interval* is a set of consecutive time points. For example,  $[a, b)$  is a time interval including the time points between  $a$  and  $b$ , and  $a$  but not  $b$ . A *temporal set* is a set of disjoint time intervals. In other words, a temporal set is a set of time points which can be grouped into disjoint time intervals. Some example of temporal sets are  $\{[5, 10)\}$ ,  $\{[15, 20)\}$ ,  $\{[20, 30)\}$  etc.

The fundamental construct of our model is a *temporal atom*,  $\langle t, v \rangle$ , where  $t$  is a temporal set and  $v$  is a value. The temporal atom denotes that the value  $v$  is valid over the time duration represented by the temporal set  $t$ . It represents an attribute's value. The

history of an attribute is represented as a set of temporal atoms.

*In Heterogeneous Databases Environment., Each Database Contains Nested Historical Relations To Support Object-Oriented Features*

In our model, each database contains nested historical relations to support object-oriented features. We assume that all attributes are modeled as temporal atoms. We then use nested (Non-1NF) relations to model histories of complex objects. In other words, attribute values can even be relations whose tuple components are made up of temporal atoms or previously defined relations. There are more distinct features for our model compared with the model in [8].

*Data Updating Needs Keys Information*

In our model, we assume that for each new data insertion in heterogeneous databases environment. We need to provide the keys information. Note that the attribute values in the DEPT Schema are temporal atoms. Suppose attribute A has temporal atom  $\langle t_A, v_A \rangle$ , attribute B has temporal atom  $\langle t_B, v_B \rangle$ , and suppose that  $t_A \supseteq t_B$ , we say that attribute A determines attribute B in a time perspective (i.e.,  $A \rightarrow_t B$ ) if A determines attribute B (or if B is functionally dependent of A) in a static case (i.e.,  $A \rightarrow B$ ).

An example of nested historical relation schema DEPT is shown in Figure 1. It could be represented by a schema tree shown in Figure 2. A DEPT database relation is illustrated in Figure 3.

Note that there is only one Dno value in DEPT Schema in Figures 1 and there is a well-defined set of temporal atoms for Dmanager attribute, so, the functional dependency,  $Dno \rightarrow Dmanager$  of the static case turns into a multivalued dependency,  $Dno \twoheadrightarrow Dmanager$  in the temporal database. Also, note that  $Dno \twoheadrightarrow Dmanager, Dloc; Dno, Projects.Pno, Employees.Eno \twoheadrightarrow Employees.Eno, Employees.Salary; Projects.Pno \twoheadrightarrow Projects.Ploc$

Therefore, if we want to inserted  $\langle 5, 10, 60000 \rangle$  to Employees.Salary object, we should specify that  $Dno = \langle 0, n, 3 \rangle$ ,  $Projects.Pno = \langle 5, n, 1 \rangle$ ,  $Employees.Eno = \langle 5, 10, 34 \rangle$ , and then we could insert  $\langle 5, 10, 60000 \rangle$  to Employees.Salary object in DEPT Schema. Because  $[5, 10] \subset [5, n] \subset [0, n]$ .

DEPT(Dno, Dmanager, Employees(Eno, Salary), Projects(Pno, Ploc), Dloc)

Figure 1. DEPT Nested Historical Relation Schema

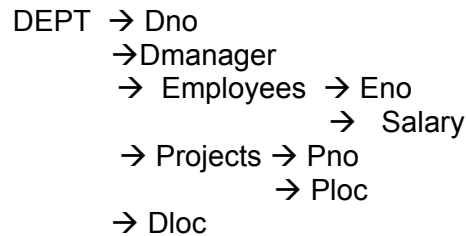


Figure 2. Schema Tree for DEPT Schema

Dno	Dmanager	Employees		Projects		Dloc
		Eno	Salary	Pno	Ploc	
<0, n, 3>	<0, 15, Smith> <15, n, Lord>	<5, n, 34>	<5, 10, 60000> <10, 15, 70000> <15, n, 75000>	<5, n, 1>	<5, n, NJ>	<0, n, NJ>
		<8, n, 25>	<8, 20, 45000> <20, n, 50000>	<8, n, 2>	<8, n, MA>	<0, n, NJ>

Figure 3. Example of Tuples of Nested Historical Relation DEPT

### III. CONSOLIDATION - SCHEMA INTEGRATION IN HETEROGENEOUS DATABASES ENVIRONMENT

In this section, we discuss the data in data warehouse in heterogeneous databases environment. *Domain mapping* and *common virtual attributes* were proposed for resolving the *domain mismatch problems* [5]. In relational database design, a *domain mapping* defines a correspondence (could be *one-to-many*) between domains of two different attributes. By mapping values in the attributes to ones in the common virtual attribute, relations can be transformed into a compatible form suitable for query executions. The concept of probabilistic partial values has been discussed in [6].

### Data Consolidation In Data Warehouse In Heterogeneous Databases Environment

We will illustrate the probabilistic partial values concept by an example of consolidated nested historical relation Emp\_Proj in data warehouse shown in Figure 7. Note that in Figure 7, the contents of Emp\_Proj relation is a consolidation results from relations from source 1 and source 2 shown in Figures 5, 6, respectively.

Dno	Dmanager	Employees		Projects		Dloc
		Eno	Salary	Pno	Ploc	
<0, n, 3>	<0, 15, Smith> <15, n, Lord>	<5, n, 34>	<5, 10, 60000> <10, 15, 70000> <15, n, 75000>	<5, n, 1>	<5, n, NJ>	<0, n, NJ>
<0, n, 3>	<0, 15, Smith> <15, n, Lord>	<8, n, 25>	<8, 20, 45000> <20, n, 50000>	<8, n, 2>	<8, n, MA>	<0, n, NJ>

Figure 4. Example of Tuples of Nested Historical Relation Dept after applying Unnest  $\mu_{\text{Employees}}$  (DEPT) operation.

In our heterogeneous databases environment, each database contains nested historical relations to support object-oriented features. We further assume that we perform temporal relational operator *Unnest* to each database of information source. Some

temporal relational operators for the nested historical relation have been discussed in [14]. In Figure 4, we showed that an example of tuples of nested historical relation Dept after applying *Unnest*  $\mu_{\text{Employees}}(\text{DEPT})$ .

Eno	Title	Projects	
		Pno	Ploc
<0, n, 34>	<0, 10, Prog> <10, n, Mgr>	<0, n, 1>	<0,n, NJ>
<8, n, 25>	<8,n, DBA>	<8, n, 2>	<8, 15, NY> <15, 25, NJ> <25, 30, MA> <30, n, NY>
<5, n, 91>	<5,n, NTech>	<5, n, 6>	<5,n,NJ>

Figure 5. Emp\_Proj1 Nested Historical Relation from information source 1.

Eno	Job	Projects	
		Pno	Pcity
<0,n, 34>	<0, n, SE>	<0,n, 1>	<0,n, Newark>
<8, n, 25>	<8,15, DBA> <15, n, DBdv>	<8, 15, 2> <15, n, 5>	<8, 15, NYC> <15, 30, Jersey City>
<5, n, 91>	<5,n,N Eng>	<5, n, 6>	<5,n, Jersey City>

Figure 6. Emp\_Proj2 Nested Historical Relation from information source 2.

Now, suppose we have two relations collected from source 1 and source 2. After

applying *Unnest on Employees* (i.e.,  $\mu_{\text{Employees}}$ ) to both relations, we have the results shown in Figures 5, 6, respectively.

Eno	Title	Projects	
		Pno	Ploc
<0, n, 34>	<0, 10, Prog <sup>0.5</sup> , SE <sup>0.5</sup> > <10, n, Mgr <sup>0.5</sup> , SE <sup>0.5</sup> >	<0, n, 1>	<0,n, NJ>
<8, n, 25>	<8,15, DBA > <15, n, DBA <sup>0.5</sup> , DBdv <sup>0.5</sup> >	<8, 15, 2> <15, n, 2 <sup>0.5</sup> , 5 <sup>0.5</sup> >	<8, 15, NY> <15, 25, NJ> <25, 30, NJ <sup>0.5</sup> , MA <sup>0.5</sup> > <30, n, NY>
<5, 15, 91>	<5,n, NTech <sup>0.5</sup> , NEng <sup>0.5</sup> >	<5, n, 6>	<5,n,NJ>

Figure 7. Consolidated Emp\_Proj Nested Historical Relation in Data Warehouse with Probabilistic Partial Values Concept.

In 7, we illustrate the probabilistic partial values concept by an example of consolidated nested historical relation Emp\_Proj in data warehouse shown in Figure 7. Note that in Figure 7, the contents of Emp\_Proj relation is a consolidation results from databases from source 1 and source 2 shown in Figures 1, 2, respectively. Note that the values in the attribute Ploc are represented by states, therefore, Pcity  $\subset$  Ploc. Also note that both Newark and Jersey

City are located in New Jersey (NJ) State. The value  $\langle 25, 30, NJ^{0.5}, MA^{0.5} \rangle$  means that in time interval  $\langle 25, 30 \rangle$ , the Ploc's value is NJ with probability 0.5 and the Ploc's value is MA with probability 0.5. In Figure 8, we list the titles/jobs and their descriptions used in Figures 5, 6, 7.

No.	Title/Job	Title/Job Description
1	Prog	Programmer
2	SE	Software Engineer
3	Mgr	Manager
4	DBA	Database Administrator
5	DBdv	Database Developer
6	NTech	Network Technician
7	NEng	Network Engineer

Figure 8. Titles/Jobs and their descriptions used in Figures 5, 6, 7.

Now, we could specify the relational operator to query information from the Consolidated Emp\_Proj nested historical relation in Figure 7. For example, if we specify that

$$\sigma_{\langle Ploc=NY \rangle \wedge \langle Title=DBA \rangle \wedge \langle 8,40 \rangle} (Emp\_Proj),$$

we could get the result shown in Figure 9. Furthermore, when we provide another query.

$$\sigma_{\langle Ploc=NY \rangle \wedge \langle Title=DBA \rangle \wedge \langle 8,40 \rangle \wedge \langle Pno=2 \rangle} (Emp\_Proj)$$

We could get the result in Figure 10.

It contains a column of probability, in where  $\langle 8, 15, 1 \rangle$  denotes that the probability is 1 for Eno=2 of Ploc=NY and Title=DBA and Pno=2 .

Eno	Title	Projects	
		Pno	Ploc
$\langle 8, 40, 25 \rangle$	$\langle 8,15, DBA \rangle$ $\langle 15, 40, DBA^{0.5}, DBdv^{0.5} \rangle$	$\langle 8, 15, 2 \rangle$ $\langle 15, 40, 2^{0.5}, 5^{0.5} \rangle$	$\langle 8, 15, NY \rangle$ $\langle 30, 40, NY \rangle$

Figure 9.

$$\sigma_{\langle Ploc=NY \rangle \wedge \langle Title=DBA \rangle \wedge \langle 8,40 \rangle} (Emp\_Proj)$$

Prob	Eno	Title	Projects	
			Pno	Ploc
$\langle 8, 15, 1 \rangle$	$\langle 8, 40, 25 \rangle$	$\langle 8,15, DBA \rangle$ $\langle 15, 40, DBA^{0.5}, DBdv^{0.5} \rangle$	$\langle 8, 15, 2 \rangle$ $\langle 15, 40, 2^{0.5}, 5^{0.5} \rangle$	$\langle 8, 15, NY \rangle$ $\langle 30, 40, NY \rangle$

Figure 10.

$$\sigma_{\langle Ploc=NY \rangle \wedge \langle Title=DBA \rangle \wedge \langle 8,40 \rangle \wedge \langle Pno=2 \rangle} (Emp\_Proj)$$

## IV. The Extended Relational Operators

In this Section, we present the basic temporal relational algebra operators with the probabilistic partial values concept. Note that we will not intend to define these operators by the formal definitions, but we would like to shed light on the main ideas of these operators and their usefulness for the nested historical relational model in the heterogeneous databases environment.

(1) Selection

A more general selection,  $\alpha$ -selection could be defined as follows.

$$\hat{\sigma}_{\langle t_1, t_2 \rangle}^\alpha(\mathbf{R}) \equiv \{r \mid r \in \mathbf{R} \wedge \text{prob}_{\langle t_1, t_2 \rangle}(r) \geq \alpha\}$$

where  $\text{prob}_{\langle t_1, t_2 \rangle}(r)$  is the probability of a tuple  $r$  in  $\langle t_1, t_2 \rangle$ .

(2) Projection

*Project* ( $\hat{\pi}$ ): is the same as the standard projection operation  $\pi$ . In a nested relation, when a higher order name is selected from relation, all of its descendants are also discarded. Duplicate tuples are eliminated. Tuples are coalesced (i.e., union of temporal sets are taken) if their values parts agree on all their components.

(3) Intersection, Set difference, Union Temporal Relational Algebraic Operators Union( $\hat{\cup}$ ), Intersection( $\hat{\cap}$ ), Minus( $\hat{-}$ )

Same as the conventional relational operators, before applying the extended intersection, set difference, and Union operators to two relations, we need to check the given two relations are *union-compatible*. That is, two relations have the same number of attributes, and each corresponding pair of attributes has the same domain. We illustrate these operators by the following example. Suppose we have two relations.  $T_1$  :

Eno	Salary
$\langle 0, n, 11 \rangle$	$\langle 0, 15, 45K^{0.5}, 50K^{0.5} \rangle$ $\langle 15, n, 60K \rangle$

and  $T_2$  :

Eno	Salary
$\langle 0, n, 11 \rangle$	$\langle 0, 20, 40K \rangle$ $\langle 20, n, 55K \rangle$

$$T_1 \hat{\cup} T_2 =$$

Eno	Salary
$\langle 0, n, 11 \rangle$	$\langle 0, 15, 40K^{1/3}, 45K^{1/3}, 50K^{1/3} \rangle$ $\langle 15, 20, 40K^{0.5}, 60K^{0.5} \rangle$ $\langle 20, n, 55K^{0.5}, 60K^{0.5} \rangle$

$$T_1 \hat{\cap} T_2 =$$

Eno	Salary

$$T_1 \hat{-} T_2 = T_1$$

(4) Cartesian Product

*Cartesian Product* ( $\hat{\times}$ ): The attributes of the operand relations are juxtaposed and each tuple of the first relation is concatenated with each tuple of the second relation. The schema tree of the result is formed by replacing the two roots of the operand trees with one root containing all their children.

(5) Join

Note that a *temporal relational algebraic operator*, *Join*  $\triangleright \hat{\bowtie}$  can be formed by cartesian product and selection operations. We assume that *the extended join*  $\triangleright \hat{\bowtie}$  should involve a *primary key in the join condition*. This is consistent with our assumption that each new data insertion in heterogeneous databases environment. We need to provide the keys information.

Therefore, Join on non-key attributes is not a meaning operation in our environment.

*(6) Integration*

Let the two relations be  $A(K, C, X)$  and  $B(K, C, Y)$ , where  $A(K, C)$  and  $B(K, C)$  are union-compatible but  $A(X)$  and  $B(Y)$  are not. Then the integration  $A \uplus B$  operator generates an integrated scheme  $(K, C, X, Y)$ .

Now, suppose  $T_3 =$

Eno	Salary	Mgr
<0,n,11>	<0,15, 45K <sup>0.5</sup> , 50K <sup>0.5</sup> > <15, n, 60K>	<0,20,Smith> <20,n, Lee

and let  $T_4 =$

Eno	Salary	WPhone
<0,n,11>	<0,20, 40K> <20, n, 55K>	<0,n, 718- 789-4321

$T_3 \uplus T_4 =$

Eno	Salary	Mgr	WPhone
<0, n,11 >	<0,15, 40K <sup>1/3</sup> , 45K <sup>1/3</sup> , 50K <sup>1/3</sup> > <15, 20, 40K <sup>0.5</sup> , 60K <sup>0.5</sup> > <20,n, 55K <sup>0.5</sup> , 60K <sup>0.5</sup> >	<0,20,Smit h> <20,n, Lee	<0,n, 718- 789-4321

*(7) Unnest*

Unnest ( $\mu$ ): This operation replaces a higher order name by its descendants in the next lower level in the Schema Tree. Repeated application of unnest operation flattens a nested historical relation.

*(8) Nest*

Nest ( $\nu$ ): It is the reverse of the unnest operation. For example, if we apply nesting DEPT' on Eno, Salary, Age, it recreates the original relation DEPT. That is,  
 $Nest_{Eno, Salary, Age}(\nu) = DEPT$ .

The following two temporal relational aggregate operators are useful for manipulating temporal sets of attributes.

*(9) Slice*

Slice: Given two attributes, it aligns the first attribute's time with respect to the time of second attribute by intersecting of temporal sets of these attributes is assigned to the first attribute as its time reference.

*(10) Transfer-time*

Transfer-time: it replaces the temporal set of the first attribute by the temporal set of the second attribute. Note that

$Slice_{Dmanager, Eno} (DEPT')$  is equivalent to apply  $Transfer-time_{Dmanager, Eno} (DEPT') = X$  followed by  $DEPT' \cap X$ .

*(11) Select-time:*

Select-time: it selects tuples that satisfy the specified temporal set. Note that Select-time is a redundant operation which can be directly expressed by Select operator with specified temporal set in the selection condition.

## V. CONCLUSIONS AND FUTURE WORK

In this work, we studied schema integration work for nested historical relations which are managing temporal variation of complex objects with imprecise information in the heterogeneous databases environment. We extended the concept of probabilistic partial values in database relations to the nested historical relations. We then developed a full set of temporal relational operators over nested historical relational data model. For the extension of this work, we will formally define and study the properties and computation aspects of those temporal relational operators discussed in this paper.

## VI. REFERENCES

- [1] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*, Fifth Edition, Addison-Wesley, 2006.
- [2] H. Garcia-Molina, J. D. Ullman, and J. Widom, *Database Systems: The Complete Book*, Prentice-Hall, 2002.
- [3] C. E. Dyreson, "Temporal Coalescing with Now, Granularity, and Incomplete information, ACM SIGMOD 2003, June 9 -12, 2003.
- [4] D. Dubois and H. Prade, *Possibility Theory: An Approach to Computerized Processing*. New York: Plenum Press, 1986.
- [5] L. G. DeMichiel, "Resolving Database Incompatibility: An Approach to performing Relational Operations over Mismatched Domains," *IEEE Trans. Knowledge and Data Engineering*, Vol. 1, No. 4, pp. 485-493, 1989.
- [6] F. S. C. Tseng, A. L. P. Chen and W. P. Yang, "Answering Heterogeneous Database Queries with Degrees of Uncertainty", *Distributed and Parallel Databases: An International Journal*, Vol. 1, No. 3, pp. 281 – 302.
- [7] R. Elmasri, V. Kouramajian, and S. Fernando, "Temporal Database Modeling: An Object-Oriented Approach", *ACM CIKM'93*, November, 1993, D.C. , U.S.A., pp. 574 -585.
- [8] A. U. Tansel and L. Garnett, "Nested Historical Relations", *ACM*, 1989.
- [9] R. Sunderraman, *Oracle 9i Programming – A Primer*, Addison-Wesley, 2004.
- [10] M. A. Roth, H. F. Korth and A. Silberschatz, "Extended Algebra and Calculus for Nested Relational Databases, *ACM Transactions on Database Systems*, Vol. 13, No. 4, December 1988, pp. 389-417.
- [11] J. Paredaens and D. Van Gucht, "Converting Nested Algebra Expressions into Flat Algebra Expressions", *ACM Transactions on Database Systems*, Vol. 17, No. 1, March 1992, pp. 65-93.
- [12] A. Dekhtyar, R. Ross and V. S. Subrahmanian, "Probabilistic Temporal Databases, I: Algebra, *ACM Transaction on Database Systems*, Vol. 26, No. 1, March 2001, pp. 41-95.
- [13] D. Dey and S. Sarkar, "Generalized Normal Forms for Probabilistic Relational Data," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 14, No. 3, pp. 485 - 497, May/June, 2002.
- [14] Ping-Tsai Chung and Hsin-Hua Hsiao, "On Computing Aggregate Functions for Nested Historical Relations in Heterogeneous Databases Environment", *Proceedings of the 2005 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2005)*, World Academy of Science.