Applied Mathematics & Information Sciences
An International Journal

Mapping Composition Combining Schema and Data Level Heterogeneity in Peer Data Sharing Systems

Md. Anisur Rahman[1] *and Mehedi Masud*[2]

[1] EECS, University of Ottawa, Ottawa, Canada
[2] Computer Science Department, Taif University, KSA

Received: 9 Feb. 2012; Revised 13 Oct. 2012; Accepted 21 Oct. 2012
Published online: 1 Jan. 2013

Abstract: The mapping semantics that combines the schema-level and the data-level mappings is called bi-level mappings. Bi-level mappings enhance data sharing overcoming the limitations of the non-combined approaches. This paper presents an algorithm for composing two bi-level mappings by using tableaux. Composition of mappings between peers has several computational advantages in a peer data management system, such as yielding more efficient query translation, pruning redundant paths, and better query execution plans. We also present a distributed algorithm for computing direct mapping between two end peers of a series of peers connected by a chain of mappings.

Keywords: schema mappings, data interoperability, database exchange, peer to peer.

1. Introduction

In a Peer Data Management System (PDMS), each peer chooses its own database schemas, and maintains data independently of other peers. Contrary to the traditional data integration systems [1,2,8] where a global mediated schema is required for data exchange, in a PDMS the semantic relationships exist between a pair of peers, or among a small set of peers for sharing data. The data is shared globally among the peers by traversing the transitive relationships among semantically related peers.

Creating a unique global mediated schema is impractical in a PDMS due to the volatility, peer autonomy, and scalability issues. Instead, mappings are implemented only between pairs of peers to unify the heterogeneous data sources. These mappings describe the relationship between the terms used in different peers. *Schema mappings* [10–14] and *data mappings* [5] are used to address the schema-level and data-level heterogeneity, respectively. Several strategies are introduced for the schema mappings between the mediated and local schemas including, *global-as-view* (GAV) [9], *local-as-view* (LAV) [10], and *global-and-local-as-view* (GLAV) [8]. In GAV approach, the mediated schema is described in terms of local sources. In LAV, the local sources are described in terms of the mediated schema. GLAV is the combination of GAV and LAV approaches to integrate the mediated and local schemas.

Schema-level mappings are effective only when the differences between the schemas are mainly structural, i.e. attribute values represent the same information, or can be transformed to be the same. However, data-level mappings are necessary when semantically related attribute values differ. Data mappings are implemented by *mapping tables* [5] which are relations on the attributes being mapped. The tuples in the mapping tables show the correspondence between values in the mapped relations. These tables are treated as constraints (aka mapping constraints) on the exchange of data between peers.

In most of the PDMS systems, both schema-level and data-level heterogeneity occur simultaneously. Section 2 shows a situation of answering a query in a PDMS where neither schema-level nor data-level mappings are good enough to tackle the situation separately. Rather a combined approach is needed which will address both data-level and schema-level

* Corresponding author: e-mail: mmasud@tu.edu.sa

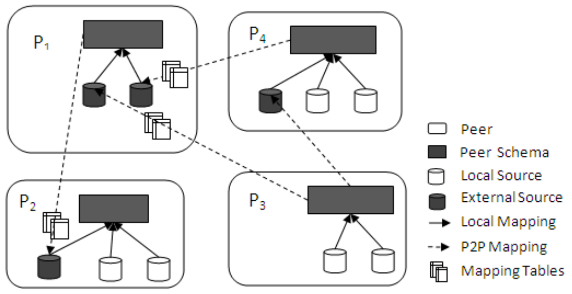


Figure 1 General Scenario of P2P System [16]

heterogeneity at the same time. Addressing the need of the combined mappings, authors in [15, 16] presented the semantics of mappings that combine both the schema-level and the data-level mappings. This combined mapping is called *bi-level* mapping.

Combining two mappings into a single one is called *mapping composition*. Composition of mappings between peers has several computational advantages in a PDMS, such as yielding more efficient query translation, pruning redundant paths, and better query execution plans. This paper introduces a tableau based [3] technique for composing the bi-level mappings. Essentially, the bi-level mappings are first expressed by tableaux [3]. Then the composition is performed by manipulating those tableaux. Sometimes it is necessary to compose a series of mappings to deduce direct mapping between the two end points of a chain of peers that is called *chain composition*. The paper also presents a distributed algorithm that distributes the computation load among the peers for computing the chain composition of the bi-level mappings.

The paper is organized as follows: Section 2 presents the general scenario of a PDMS and discusses the need of the bi-level mappings by giving a motivating example. Section 3 gives the model of a PDMS and discusses the semantics of bi-level mappings. Section 4 presents the composition process of bi-level mappings. Section 5 discusses the algorithm of chain composition of bi-level mappings. Section 6 reviews related work. Finally, Section 7 discusses some conclusive remarks.

2. PDMS with bi-level Mapping

This section discusses a peer database management system considering bi-level mappings. Figure 1 depicts a sample scenario of such a PDMS. As shown in figure, each peer has its own local source. The local source in a peer is designed independently during the creation of the local database of that peer. The result of a local query posed in a peer is produced from the local source of the peer. A peer also defines *external sources* that are used to access data from its acquainted peers. An external source is a view of a peer schema of an acquainted peer and is defined through GAV mappings. In order to provide a unique access view of local as well as remote data to the users, each peer defines a schema called *peer schema*. This schema is defined considering the local sources and external sources. GLAV mappings are used to create the peer schema.

There are some advantages of using external sources instead of making direct link between the peer schemas. Firstly, it can tackle the dynamic nature of a P2P system with ease. Whenever a peer finds an acquaintance peer for the first time, it creates some external sources and link it both to its own peer schema and with the peer schema of the acquainted peer. Thus the external source can be treated by the target peer in the same way as local sources. When the source peer of an external source becomes unavailable in the network, it simply becomes an empty relation. Once an external source is created, any time the corresponding source peer becomes available it gets activated. Secondly, treating external sources as local sources, peer schema can be defined in a straight forward and autonomous way.

Since peers are fully autonomous to design their database and store data with their own format, heterogeneity among the peers may come in two forms: (i) schema-level and (ii) data-level. Notice that by creating the mappings through GLAV, the schema-level heterogeneity can be resolved. Authors in [4] proposed such a scheme for creating mappings between peers that resolve schema-level heterogeneity. However, the GLAV approach can not resolve the data-level heterogeneity among peers. Authors in [5] proposed the mapping table semantics that is used to resolve data-level heterogeneity between peers. However, mapping tables are not sufficient for resolving schema-level heterogeneity. The following example shows the need for a mapping with higher expressivity.

Suppose, the peers P_1 and P_2 in Figure 2 store employee information to be shared with each other. Assume that P_1 has a local source *Empl_List(Id, Name, Position, salary)*. The attributes *Id*, *Name*, *Position*, and *Salary* represent identification number, name, position, and the salary of an employee respectively. Similarly, P_2 stores its employee information using the local sources *Employee(Id, Name, Jid)* and *Job_Desc(Jid, Job_Description)*. Attributes *Jid* and *Job_Description* represent the job identification number and the title of the job, respectively. Also assume that P_1 has an external source *E(Name, Position)* that illustrates that peer P_1 is interested in only the names and positions of employees stored in P_2. In order to give a unique access view to the data stored in P_1 and P_2, peer P_1 defines a peer schema $PS1$ which contains a single view *N_P(Name, Position)* for its users. Since *N_P* logically integrates information from the local source *Empl_List* and the external source E, their relationships with N_P are $N\_P \supseteq \pi_{Name,Position}(Empl\_List)$ and $N\_P \supseteq E$ respectively. In the same manner, peer P_2 also creates its peer schema $PS2$ which contains two views *P2E(Name, Jid)* and *P2J(Jid, Job_Description)*. This schema is designed considering its local source and other external sources from other peers (not mentioned in the figure).

From the Figure 2, notice that P_2 has a mapping table *mt* that maps the data vocabularies of the attribute *Job_Description* in source *Job_Desc* with the attribute *Position* in source *Empl_List* of peer P_1. This mapping table is created since the two peers store job information using two different vocabularies. External source E in P_1 is defined as a view on the relations of the peer schema of P_2. In the following, different situations are illustrated that may occur when a query is posed to a peer. The examples show the need for the bi-level mappings that this paper advocates for P2P systems. In summary, peer P_1 has the local schema $\{Empl\_List(Id, Name, Position, Salary)\}$, peer schema $PS1 = \{N\_P(Name, Position)\}$, and an external source $E(Name, Position)$. Peer P_2 has the local schema $\{Employee(Id, Name, Jid), Job\_Desc(Jid, Job\_description)\}$, and peer schema $PS2 = \{P2E(Name, Jid), P2J(Jid, Job\_Description)\}$.

Example 1(Considering only schema mappings). Assume that the mapping table *mt* is absent in peer P_2. In this case, peer P_1 has only the schema-level mappings with P_2. Suppose the query

$$q_1 : \pi_{Name}(\sigma_{Position=CEO}(N\_P))$$

is posed at P_1 through its peer schema. Considering the mappings between N_P and *Empl_List*, q_1 is translated for the local source at P_1 as

$$q_1^1 : \pi_{Name}(\sigma_{Position=CEO}(Empl\_List)).$$

Moreover, using the mappings between N_P and E, q_1 is translated for the external source at P_1 as

$$q_1^{1\prime} : \pi_{Name}(\sigma_{Position=CEO}(E)).$$

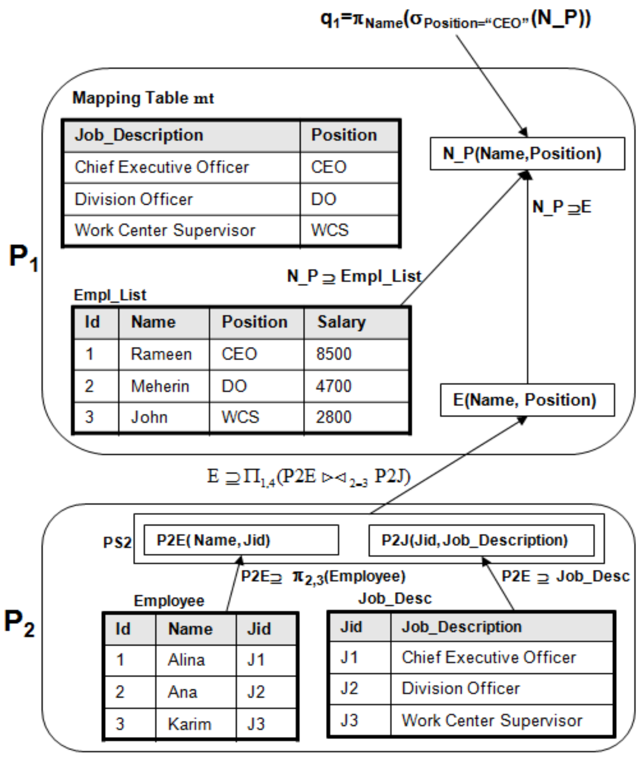
Based on the mappings between E and the peer schema at P_2, query $q_1^{1\prime}$ is translated as

$$q_1^2 : \pi_{Name}(\sigma_{Job\_Description=CEO}(P2E \bowtie P2J))$$

which is finally translated according to the local vocabulary of P_2 as

$$q_1^{2\prime} : \pi_{Name}(\sigma_{Job\_Description=CEO}(Employee \bowtie Job\_Desc))$$

Notice that the final result of the query q_1 is: $\{'Rameen'\}$ which is returned only from the local source at P_1. If 'CEO' and 'Chief Executive Officer' to be semantically equivalent then q_1 should extract 'Alina' from P_2. Due to absence of data-level mappings, the query can not produce this result. Now assume that the mapping table *mt* exists in P_1. Hence, q_1'' is translated for P_2 as $q_1^{2\prime} : \pi_{Name}(\sigma_{Position='CEO'}(Employee \bowtie Job\_Desc \bowtie mt))$.



Mapping Table mt

Job_Description	Position
Chief Executive Officer	CEO
Division Officer	DO
Work Center Supervisor	WCS

Empl_List

Id	Name	Position	Salary
1	Rameen	CEO	8500
2	Meherin	DO	4700
3	John	WCS	2800

Employee

Id	Name	Jid
1	Alina	J1
2	Ana	J2
3	Karim	J3

Job_Desc

Jid	Job_Description
J1	Chief Executive Officer
J2	Division Officer
J3	Work Center Supervisor

Figure 2 Motivating Example

This case returns more results for the query q_1 and the complete answer to this query becomes: $\{'Rameen', 'Alina'\}$

Example 2(Considering only data mappings). In Figure 2 the external source E of P_1 is defined in terms of $P2E$ and $P2J$ of peer P_2. *Projection* and *Join* operators are used in that definition. Mapping tables are not expressive enough to express *Projection* or *Join*. Schema mappings are needed for such association between two sources.

So, a mapping is necessary that is capable of dealing with both the syntactical (schema-level) and the semantic (data-level) heterogeneities at the same time.

3. Model of a PDMS

In this section we define a P2P system Π. Before defining the system we introduce the notions of a peer and peer mappings.

A peer $P_i \in \mathcal{P}$ is a tuple, where $P_i = (PS_i, R_i, L_i)$. Where,

-PS_i is the peer schema through which data in a peer is exposed to the external world.
-R_i is the set of sources comprised of local and external sources. This is called peer source or simply source.
-L_i is the set of GLAV local mappings which define the mappings between R_i and PS_i. Each local mapping, called *mapping assertion* (aka *tuple generating dependency*), in L_i has the form

$$\forall \mathbf{x}(\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}) \rightsquigarrow \exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z}))$$

where $\varphi(\mathbf{x}, \mathbf{y})$ and $\psi(\mathbf{x}, \mathbf{z})$ are conjunctive queries over the relations in R_i and PS_i respectively.

Peer mappings $\mathcal{M}_i \in \mathcal{M}$ is a set of mappings, called bi-level mapping or *peer mappings*, that define the schema and data-level mappings between peers. The construction of mappings $M_i^j \subseteq \mathcal{M}_i$ forms an acquaintance (i, j) between P_i and P_j. Each mapping $m \in \mathcal{M}_i$ is a pair $< m^S_{j,k}, m^D_{j,k} >$, where:

-$m^S_{j,k}$ is a GLAV mapping (practically GAV, since $s(\mathbf{x})$ is always a single relation) of the form

$$\forall \mathbf{x}(\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}) \rightsquigarrow s(\mathbf{x}))$$

where $\varphi(\mathbf{x}, \mathbf{y})$ is a conjunctive query over the peer schema of a peer P_j and $s(\mathbf{x})$ is the k^{th} external source of P_i.
-$m^D_{j,k}$=MT=$\{mt_1, mt_2, \ldots, mt_q\} \subseteq MT^i_j$ is a set of mapping tables. MT^i_j denotes the set of mapping tables used to map data of P_j to data of P_i.
m can alternatively be represented with the mapping assertion as follows:

$$\forall \mathbf{x}(\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}) \overset{MT}{\rightsquigarrow} s(\mathbf{x}))$$

A P2P system Π is defined by a pair $\langle \mathcal{P}, \mathcal{M} \rangle$, where $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ is a set of peers and $\mathcal{M} = \{\mathcal{M}_1, \cdots, \mathcal{M}_n\}$ is a set of peer mappings.

It is assumed that a curator with expertise in different domains is responsible for generating the mapping tables and the peer administrator maintains them in a peer. Schema mappings between two peers are initially created by the corresponding peer administrators when they agree to share data. Once created, the mappings are activated or deactivated depending on the presence of the corresponding peers in the network. Generating the mappings automatically is another research area and this paper does not address this issue.

The semantics of $\overset{MT}{\rightsquigarrow}$ is described in the following section.

3.1. Semantics of local mappings

For each peer P_i, a first order logic (FOL) theory F_i is introduced, called peer theory, where alphabet contains all the relation symbols in a peer schema PS_i and the relations in R_i. The axioms of F_i include all the constraints of PS_i and one logical formula representing each local mapping in L_i. For a local mapping of the form $\forall \mathbf{x}(\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}) \rightsquigarrow \exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z}))$ in L_i, a formula of the form $\forall \mathbf{x}(\exists \mathbf{y} \varphi(\mathbf{x}, \mathbf{y}) \rightarrow \exists \mathbf{z} \psi(\mathbf{x}, \mathbf{z}))$ is added to F_i. F_i does not consider peer mappings. Thus, modeling a peer as a GLAV integration system becomes equivalent to modeling a FOL theory F_i (ignoring the peer mappings in $\mathcal{M}_i$).

3.2. Semantics of peer mappings

Similar to the local mappings, the semantics of peer mappings can also be given in terms of FOL. However, to incorporate the mapping tables, following notations and definitions are used.

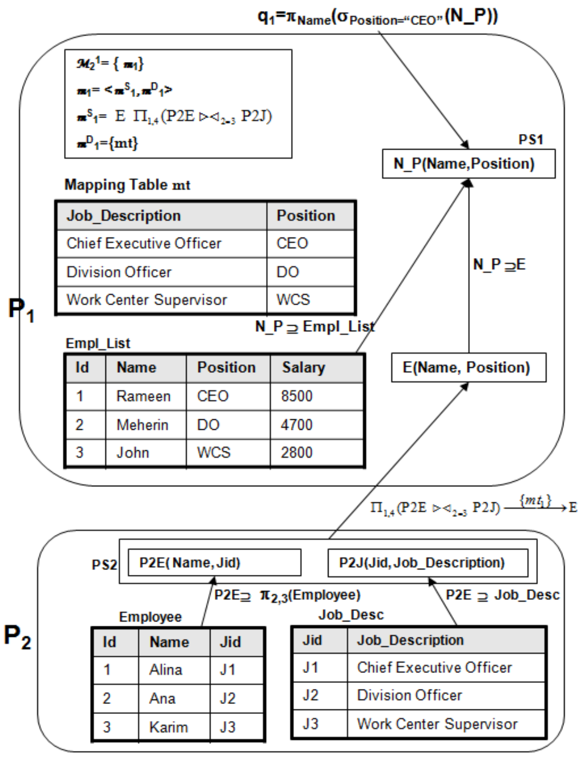
Given a tuple t and a set of attributes U, $t[U]$ denotes the values of tuple t corresponding to the attributes in U.

Definition 1(Mapping Table). Assume that U_i and U_j are non-empty set of attributes in two peers P_i and P_j respectively. A mapping table $mt[\mathbf{P}, \mathbf{Q}]$ is a finite relation over the attributes $\mathbf{P} \subseteq U_i$ and $\mathbf{Q} \subseteq U_j$. A tuple $t = (\mathbf{a},\ \mathbf{b})$ in the mapping table indicates that the value $\mathbf{a} \in dom(\mathbf{P})$ is associated with the value $\mathbf{b} \in dom(\mathbf{Q})$.

*Example 3.*Consider the instances of two peers P_1 and P_2 in Figure 2. Observe that employees' job titles are represented in two instances with different data vocabularies. In order to associate or relate employees in two peers for sharing employees' data and resolve data heterogeneity wrt employees' job title, a sample mapping tables $mt[Job\_Description, Position]$ is designed. Mapping table mt associates values of two attributes *Position* and *Job_Description*. Intuitively, according to the interpretation of the table mt, a tuple *t=(Chief Executive Officer, CEO)* indicates that a tuple in relation *Emp_List* with the value *CEO* in attribute *Position* is associated with a tuple in relations *Employee*, *Job_Desc* with the value *Chief Executive Officer* in attribute *Job_Description*.

The values appearing in the tables presented thus far only constants. However, to augment the expressiveness of tables variables are used [5]. Consider a mapping table $m[L, R]$ where both the domain of L and R are same, say D. A tuple $(v, D-v)$ in m, where v is a variable, can be used to denote that any value of L can be mapped to any value of R except to itself. Given the presence of variables in mappings, it is necessary to introduce the notion of a valuation.

Definition 2(Valuation). *[5]* A valuation ρ over a mapping table mt is a function that maps each constant value in mt to itself and each variable v of mt to the value in the intersection of the domains of the attributes where v appear.



$q_1=\pi_{Name}(\sigma_{Position=\text{“CEO”}}(N\_P))$
$\mathcal{M}_2{}^1=\{m_1\}$
$m_1=<m^S{}_1, m^D{}_1>$
$m^S{}_1=$ E $\Pi_{1,4}$(P2E $\bowtie_{2=3}$ P2J)
$m^D{}_1=\{mt\}$
PS1
N_P(Name,Position)
N_P ⊇E
N_P ⊇ Empl_List
E(Name, Position)
P1
$\Pi_{1,4}$(P2E $\bowtie_{2=3}$ P2J) $\xrightarrow{\{mt_1\}}$ E
PS2
P2E(Name, Jid)
P2J(Jid, Job_Description)
P2E⊇ $\pi_{2,3}$(Employee)
P2E ⊇ Job_Desc
P2

Mapping Table mt

Job_Description	Position
Chief Executive Officer	CEO
Division Officer	DO
Work Center Supervisor	WCS

Empl_List

Id	Name	Position	Salary
1	Rameen	CEO	8500
2	Meherin	DO	4700
3	John	WCS	2800

Employee

Id	Name	Jid
1	Alina	J1
2	Ana	J2
3	Karim	J3

Job_Desc

Jid	Job_Description
J1	Chief Executive Officer
J2	Division Officer
J3	Work Center Supervisor

(a) Bi-Level mappings for P2P mappings

E(Name, Position)

Name	Position
Alina	CEO
Ana	DO
Aurin	WCS

N_P(Name,Position)

Name	Position
Rameen	CEO
Meherin	DO
Ariya	WCS
Alina	CEO
Ana	DO
Aurin	WCS

(b) Extension of E and N_P

Figure 3 Bi-Level Mapping Example

Introduction of variables in mapping tables offers a compact and convenient way of representing common associations between values. An example of such a mapping table is illustrated in the following example.

*Example 4.*Consider Figure 3. Assume that both relations *Empt_List* and Employee have an attribute called *DOB* which represents the date of birth of employees. However, date values are stored in the format $DD-MM-YYYY$ and Name of Month DD, Year in relations *Empt_List* and Employee, respectively. There are two alternative mapping tables that can be used to represent the associations between the two sets of dates. The two tables are shown in Figure 4. Mapping table DOB2DOB in Figure 4(a) contains a set of mappings of the form (date1; date2), where date1 is a employee birth date in *Empt_List* and date2 is a employee birth date in *Employee* relation. Alternatively, we can construct a more succinct, data independent, mapping table DOB2DOB containing the single mapping, (v; v'), where v and v' are variables. The alternate mapping table is shown in Figure 4(b). A valuation function can be used that converts a date value in the *Empt_List* relation corresponding to a date value in the Employee relation.

DOB	DOB
05-02-1990	Feb. 05, 1990
16-05-1985	May 16, 1985

(a) Mapping table DOB2DOB

DOB	DOB
v	v'

(b) Alternate representation of DOB2DOB

Figure 4 Mapping tables

Definition 3($Map(mt, \mathbf{p})$**).** Let $mt[\mathbf{P}, \mathbf{Q}]$ be a mapping table from $\mathbf{P}$ to $\mathbf{Q}$ and $\mathbf{p}$ is an element of the domain of $\mathbf{P}$. $Map(mt, \mathbf{p})$ returns a set of values $\mathbf{\Phi}$. $\mathbf{q} \in \mathbf{\Phi}$ if for some $t \in mt$ there exists a valuation ρ such that $\rho(t[\mathbf{P}]) = \mathbf{p}$ and $\rho(t[\mathbf{Q}]) = \mathbf{q}$.

If the mapping for the value $\mathbf{p}$ is not defined in mt and type of $\mathbf{P}$ and $\mathbf{Q}$ matches then $\mathbf{\Phi} = \{\mathbf{p}\}$. If neither $\mathbf{p}$ is mapped to any value in mt, nor type of $\mathbf{P}$ and $\mathbf{Q}$ matches, then $\mathbf{\Phi} = \emptyset$.

*Example 5.*Consider the mapping table $mt[Job\_Description, Position]$ in Figure 3 and a value Chief Executive Officer in the attribute $Job\_Description$ of the relation Job_Desc. The $Map(mt, Chief\ Executive\ Officer)$ returns a set of values $\mathbf{\Phi}$=$\{CEO\}$ since there exists a valuation ρ and a tuple t=(Chief Executive Officer,CEO)$\in mt$ such that $\rho(t[Job\_Description])$=Chief Executive Officer and $\rho(t[position]) = CEO$. However, if Chief Executive Officer $\notin$ $\rho(t[Job\_Description])$ then $\rho(t[position])$=Chief Executive Officer.

Definition 4($Augmentation\ function\ \tau$**).** *Let* $\mathbf{x}$ *be a tuple whose schema contains the attributes* $\mathbf{P}$*. An augmentation function* $\tau(\mathbf{x}, \mathbf{P}, \mathbf{Q}, q)$ *returns a tuple* $\mathbf{x'}$*, where* $\mathbf{x'}$ *is exactly like* $\mathbf{x}$ *except the schema of* $\mathbf{x'}$ *has the attributes* $\mathbf{Q}$ *in place of* $\mathbf{P}$ *and* $\mathbf{x'}[\mathbf{Q}] = q$*.*

*Example 6.*Consider Figure 3 and the mapping table $mt[Job\_Description, Position]$. Let $\mathbf{x} = (1, Rameen, CEO)$ be a tuple with the schema $(id, Name, Position)$ that contains attribute $\mathbf{P}$=Position. Therefore, the augmentation function

τ((1,Rameen,CEO),(id,Name,Position), (id,Name,Job_Description),(Chief Executive Officer)) returns (1,Rameen,Chief Executive Officer) wrt to the mapping table mt.

Let $mt[\mathbf{P}, \mathbf{Q}]$ be a mapping table and $\mathbf{x}$ be a tuple, $mt[\mathbf{x}]$ denotes a set of tuples obtained by replacing the values of $\mathbf{P}$ attributes of $\mathbf{x}$ by the corresponding mapped values of $\mathbf{Q}$ in mt. Formally,

$$mt[\mathbf{x}] \equiv \{\tau(\mathbf{x}, \mathbf{P}, \mathbf{Q}, \mathbf{q}) \mid \mathbf{q} \in Map(mt, \mathbf{x}[\mathbf{P}])\}$$

Let $MT = \{mt_1[\mathbf{P_1}, \mathbf{Q_1}], mt_2[\mathbf{P_2}, \mathbf{Q_2}], \ldots, mt_n[\mathbf{P_n}, \mathbf{Q_n}]\}$ be a set of mapping tables, where for any pairs of mapping tables $(mt_i[\mathbf{P_i}, \mathbf{Q_i}], mt_j[\mathbf{P_j}, \mathbf{Q_j}])$, $mt_i \neq mt_j \implies \mathbf{P_i} \neq \mathbf{P_j}$, then $MT[\mathbf{x}]$ denotes a set of tuples resulted from transformation of $\mathbf{x}$ by all the member mapping tables of MT. Formally,

$$MT[\mathbf{x}] \equiv \{\tau(\ldots\tau(\tau(\mathbf{x}, \mathbf{P_1}, \mathbf{Q_1}, \mathbf{q_1}), \mathbf{P_2}, \mathbf{Q_2}, \mathbf{q_2})\ldots, \mathbf{P_n}, \mathbf{Q_n}, \mathbf{q_n}) \mid \mathbf{q_1} \in Map(mt_1, \mathbf{x}[\mathbf{P_1}]) \wedge \ldots \wedge \mathbf{q_n} \in Map(mt_n, \mathbf{x}[\mathbf{P_n}])\}$$

[ ] is overloaded in many of our definitions; its meaning, however, will be clearly understood from the context.

Let $\mathbf{x}$ be a tuple. $Schema(\mathbf{x})$ returns the schema of that tuple, i.e. it returns the set of attributes whose values constituted the tuple. *Schema()* can be overloaded by providing its parameter as a *relation/view*. In this case, it would return the schema of the relation.

The semantics of peer mappings is defined below. It is already mentioned that a mapping assertion of a peer mapping is of the form:

$$\forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x}, \mathbf{y}) \overset{MT}{\rightsquigarrow} s(\mathbf{x}))$$

Let us assume that the above assertion defines an external source s of peer P_i in terms of the peer schema of peer P_j. An interpretation of the schema of P_i and P_j satisfies the assertion if that interpretation satisfies the following formula

$$\forall\mathbf{x}\forall\mathbf{z}(\exists\mathbf{y}(\varphi(\mathbf{x}, \mathbf{y}) \wedge \mathbf{z} \in MT[\mathbf{x}]) \equiv s(\mathbf{z}))$$

A mapping can be interpreted as a definition of how the data of the external source would be instantiated by the data of other peers. The formula also tells us that before instantiating the external source, data is converted using the corresponding mapping tables of MT. However, if there is no data-level heterogeneity, no mapping table is needed. In that case, an empty mapping table ϕ is used in the assertion. In that case, a peer mapping is represented as $\forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x}, \mathbf{y}) \overset{\phi}{\rightsquigarrow} s(\mathbf{x}))$ which satisfies the FOL formula $\forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x}, \mathbf{y}) \equiv s(\mathbf{x}))$.

*Example 7.*Let us modify the peer mapping of Figure 2 by a bi-level mapping assertion m_1 which is expressed as follows.

$$\pi_{1,4}(P2E \bowtie_{2=3} P2J) \overset{\{\mathtt{mt}\}}{\rightsquigarrow} E$$

The new scenario is shown in Figure 3(a). To satisfy the assertion, the following formula has to be satisfied.

$$\forall rtt'(\exists s(P2E(r, s) \wedge P2J(s, t) \wedge (r, t') \in mt[(r, t)]) \equiv E(r, t'))$$

Given the source database in Figure 3(a), for satisfying the above formula, the extension of the intensional source $E(Name, Position)$ has to be as shown in Figure 3(b). Figure 3(b) also shows the ultimate extension of the intentional relation $N\_P(Name,\ Position)$ in the peer schema of peer P_1. Consequently, in response to the query q_1 : $\pi_{Name}(\sigma_{Position='CEO'}(N\_P))$ to peer P_1, the PDMS will return the result $\{$**Rameen, Alina**$\}$.

3.3. Semantics of a P2P system

The semantics of a P2P system Π is given in terms of a set of models that satisfy the local and peer mappings of Π. Let a source database $\mathcal{D}$ for Π be a disjoint union of a set of local databases in each peer P_i of Π. Given a source database $\mathcal{D}$ for Π, the set of models of Π relative to $\mathcal{D}$ is:

$$sem^{\mathcal{D}}(\Pi) = \{\mathcal{I} | \mathcal{I} \text{ is a finite model of all peer theories } F_i \text{ relative to } \mathcal{D}\text{, and } \mathcal{I} \text{ satisfies all peer mappings}\}$$

Given a query q of arity k posed to a peer P_i of Π, and a source database $\mathcal{D}$, the certain answers to q relative to $\mathcal{D}$ are

$$ans(q, \Pi, \mathcal{D}) = \{\mathbf{t} | \mathbf{t} \in q^{\mathcal{I}}, \text{ for every } \mathcal{I} \in sem^{\mathcal{D}}(\Pi)\}$$

4. Composing Bi-level Mappings

The problem that is covered in this section is the following. Consider three peers A, B, and C. Assume that $M_{A\to B}$ is the mapping between A and B. Also $M_{B\to C}$ be the mapping between B and C. Our target is to compute a direct mapping $M_{A\to C}$ between A and C that is equivalent to the composition of the two mappings $M_{A\to B}$ and $M_{B\to C}$. The following example shows a composition of two bi-level mappings.

*Example 8.*Consider three peers P_1, P_2 and P_3. Assume that P_3 has two relations R_{31} and R_{32} in its peer schema and it has also a mapping table mt_{32} which maps attribute C_{322} of R_{32} in P_3 to the attribute C_{212} of R_{21} in P_2. P_2 has two relations R_{21} and R_{22} in its peer schema, an external source E_{32}, and a mapping table mt_{21}. The external source E_{32} is populated by the data of P_3 according to the peer mapping $m_{3\to 2}$: $\pi_{C_{311},C_{322}}(R_{31} \bowtie_{C_{312}=C_{321}} R_{32}) \overset{\{mt_{32}\}}{\rightsquigarrow} E_{32}$. Local mapping $m_{2\to 2}$ of P_2 maps the data of E_{32} to R_{21}. In peer P_1, it has a relation R_{11} in its peer

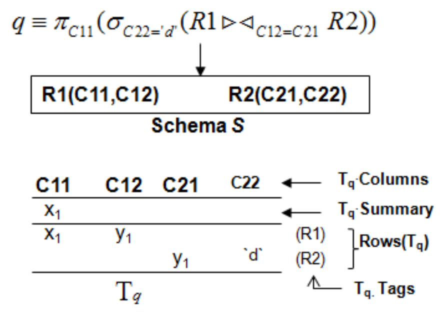


Figure 5 Converting query q to tableau T_q

schema and an external source E_{21}. E_{21} is populated by the data of P_2 according to the mapping $m_{2\to1}$:$R_{21} \overset{mt_{21}}{\rightsquigarrow} E_{21}$. Local mapping $m_{1\to1}$:$E_{32} \rightsquigarrow R_{11}$ of P_1 maps the data of E_{21} to R_{11}. Since $m_{3\to2}$ and $m_{2\to2}$ together connects the peer relations of P_3 to the peer relations of P_2, it can be represented by $M_{3\to2}$: $\pi_{C_{311},C_{322}}(R_{31} \bowtie_{C_{312}=C_{321}} R_{32}) \overset{\{mt_{32}\}}{\rightsquigarrow} R_{21}$ as a single mapping. Similarly, $m_{2\to1}$ and $m_{1\to1}$ can also be combined as as $M_{2\to1}$: $R_{21} \overset{mt_{21}}{\rightsquigarrow} R_{11}$.

Observing the above setting, there exist an indirect mapping between P_3 and P_1. Notice that R_{31} and R_{32} of P_3 is mapped to R_{11} of P_1 via R_{21} of P_2. If a direct mapping between P_3 and P_1 can be inferred from the existing mappings then P_1 will get the relevant data directly from P_3 even if P_2 is absent from the network. Observing the mappings, the direct mapping $M_{3\to1}$ between P_3 and P_1 is as follows:

$$M_{3\to1} : \pi_{C_{311},C_{322}}(R_{31} \bowtie_{C_{312}=C_{321}} R_{32}) \overset{\{mt_{32}\bowtie mt_{21}\}}{\rightsquigarrow} R_{11}$$

Section 4.2 shows how this composed mapping can be generated. The mapping composition depends on the tableaux representation of the bi-level mappings. Therefore, first expression of a bi-level mapping using tableaux is discussed.

4.1. Expressing bi-level mappings by tableaux

A *tableau query* T for the query $\forall\mathbf{x}\exists\mathbf{y}\varphi(\mathbf{x},\mathbf{y})$ is a tabular representation having the following components [3]

T.Summary- The first row of T representing the schema of the resulting tuple corresponding to $\mathbf{x}$.
T.Rows- Other rows of T each representing a relation in φ.
T.Columns- The collection of all attributes of all relations.
T.Tags- Relation names related to each rows. Columns not corresponding to attributes of the tag-relations are always blank.

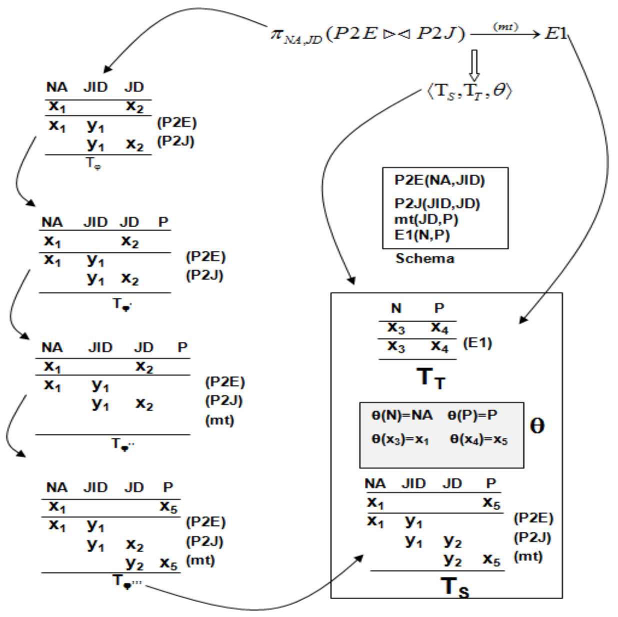


Figure 7 Steps for converting a peer mapping $\pi_{NA,JD}(P2E \bowtie_{JID=JID} P2J) \overset{\{mt\}}{\rightsquigarrow} E1$ to a triple $< T_S, T_T, \theta >$

T.Rows[R] Denotes the row of $T.Rows$ corresponding to the relation R
T.Rows[R][C] Denotes the value of C attribute for the row with tag R and T.Summary[C] denotes the value of C attribute of T.Summary. Moreover, constants and x_i are called *distinguished* symbols. Blanks and y_i are called *non-distinguished* symbols. Figure 5 shows the tableau T_q corresponding to a query q posed on a database with schema S.

A peer mapping $m_{S\to T} : \forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x},\mathbf{y}) \overset{MT}{\rightsquigarrow} e(\mathbf{x}))$ between two schema S and T can be viewed as a correspondence between two queries q_S and q_T. q_S represents the query $\forall\mathbf{x'}(\exists\mathbf{y}\varphi(\mathbf{x'},\mathbf{y}) \bowtie MT)$ over the schema S', where S' extends S by including the mapping tables of MT. $\mathbf{x'}$ is obtained by replacing $\mathbf{P_i}$ attributes in $\mathbf{x}$ by attributes $\mathbf{Q_i}$ for all $mt_i[\mathbf{P_i},\mathbf{Q_i}] \in MT$. q_T represents the query $\forall\mathbf{x}e(\mathbf{x})$ over the schema T. So, the algorithm converts the two queries into two tableaux, namely source tableau T_S and target tableau T_T, and define a homomorphism between them. The two tableaux and the homomorphism together, which forms a triple, can express the bi-level mapping. The algorithm for converting a bi-level mapping to a triple is shown in Figure 6.

*Example 9.*Figure 7 explains how the algorithm of Figure 6 works to convert the bi-level mapping $\mathcal{M}$: $\pi_{NA,JD}(P2E \bowtie_{JID=JID} P2J) \overset{\{mt\}}{\rightsquigarrow} E1$ into a triple. $Target(\mathcal{M})$(i.e. the relation on the right hand

```
Algorithm ConvertMappingToTriple($M_{S\to T}$)
Input: A bi-level mapping $M_{S\to T} \equiv \forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x},\mathbf{y}) \overset{MT}{\rightsquigarrow} e(\mathbf{x}))$
Output: A triple $< T_T, T_S, \theta >$ expressing the mapping
        $M_{S\to T}$ where $T_T$ and $T_S$ are two tableaux and $\theta$ is a homomorphism between $T_T$ and $T_S$.
begin
    $T_T \leftarrow ConvertQueryToTableau(e)$; $T_S \leftarrow ConvertQueryToTableau(\forall\mathbf{x}(\exists\mathbf{y}\varphi(\mathbf{x},\mathbf{y}))$
    $\{SA_1,\ldots,SA_n\} \leftarrow OrderColFromLToR(T_S)$; $\{TA_1,\ldots,TA_m\} \leftarrow OrderColFromLToR(T_T)$
    $numColInT_T \leftarrow |\{TA_1,\ldots,TA_m\}|$
    $j \leftarrow 1$
    for i= 1 to $numColInT_T$ do
        while $T_S.Summary[SA_j] = BLANK$ do
              $j = j + 1$
        endwhile
        $x_t \leftarrow T_T.Summary[TA_i]$; $x_s \leftarrow T_S.Summary[SA_j]$
        add $(TA_i, SA_j)$ to $\theta$ \\ i.e. $\theta(TA_i) = SA_j$
        add $(x_t, x_s)$ to $\theta$
        $j \leftarrow j + 1$
    endfor
    for each $mt[\mathbf{P},\mathbf{Q}] \in MT$ do
        add $\mathbf{Q}$ to $T_S$.Columns; add a new row to $T_S.Rows$ with tag $mt$
        $T_S.Rows[mt][\mathbf{P}] \leftarrow \mathbf{y}_{new}$; $T_S.Rows[mt][\mathbf{Q}] \leftarrow \mathbf{x}_{new}$
        $T_S.Summary[\mathbf{P}] \leftarrow$ BLANK; $T_S.Summary[\mathbf{Q}] \leftarrow T_S.Rows[mt][\mathbf{Q}]$
        if $\exists\mathbf{J}((\mathbf{J},\mathbf{P}) \in \theta)$ then replace $(\mathbf{J},\mathbf{P})$ by $(\mathbf{J},\mathbf{Q})$ in $\theta$
        for each $R \in T_S.Tags$ do
            $varP_{old} \leftarrow T_S.Rows[R][\mathbf{P}]$; $varP_{mt} \leftarrow T_S.Rows[mt][\mathbf{P}]$
            if $T_S.Rows[R][\mathbf{P}] \neq BLANK$ then $T_S.Rows[R][\mathbf{P}] \leftarrow varP_{mt}$
            if $\exists varJ((varJ, varP_{old}) \in \theta)$ then replace $(varJ, varP_{old})$ by $(varJ, varP_{mt})$ in $\theta$
        endfor
    endfor
    return $< T_S, T_T, \theta >$
end
```

Figure 6 Algorithm *ConvertMappingToTriple*

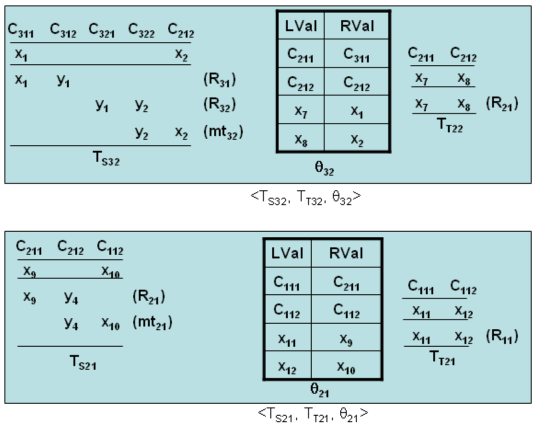


Figure 8 Tableau representation for the mapping $M_{3\to 2}$ and $M_{2\to 1}$

side of $\mathcal{M}$), i.e. $E1(N,P)$, is directly converted to a tableau T_T. *Source*($\mathcal{M}$) (i.e. the relations on the left hand side of $\mathcal{M}$), i.e. $\pi_{NA,JD}(P2E \bowtie_{JID=JID} P2J)$, is converted to a tableau T_φ (which is the initial version of T_S) as shown in Figure 7. Each entry in the homomorphism θ is a pair which associates each attribute $C_T \in T_T.Columns$ of the target schema to an attribute $C_S \in T_S.Columns$ of the source schema. When there is a pair (C_T, C_S) in θ then the distinguished variable pair $(T_T.Summary[C_T]$, $T_S.Summary[C_S])$ is also added to θ. The initial value of θ becomes $\{(N, NA),\ (P, JD), (x_3, x_1), (x_4, x_2)\}$. Gradually, T_φ and θ are modified for inclusion of each mapping table. $T_{\varphi'}$ is obtained by adding the column P to T_φ. Adding a row with tag mt to $T_{\varphi'}$ results in $T_{\varphi''}$. Shifting the summary variable from JD to P and adding variables to $T_{\varphi''}.Row[mt]$ gives $T_{\varphi'''}$. Making all the variables identical in column JD of $T_{\varphi'''}$ results in final T_S. θ is also updated when mt is added. The triple $< T_S, T_T, \theta >$, shown in the shaded box of Figure 7, now expresses the bi-level mapping $\mathcal{M}$.

```
Algorithm MergeTableau($T_1, T_2$)
Input: Two tableaux $T_1$ and $T_2$
Output: A tableau $T$ with all the rows and columns of
        $T_1$ and $T_2$
begin
    $T \leftarrow new\ Tableau()$
    for each $C \in T_1.Columns$ do
        \\ adds a new column $C$ to $T$;
        T.addNewCol(C)
        T.Summary[C]$\leftarrow T_1.Summary[C]$
        for each $R \in T_1.Tags$ do
        \\ adds a new row to T with tag R
            T.addNewRow(R)
            T.Rows[R][C] $\leftarrow T_1.Rows[R][C]$
        endfor
    endfor
    for each $C \in T_2.Columns$ do
        \\ adds a new column $C$ to $T$
        T.addNewCol(C)
        T.Summary[C]$\leftarrow T_2.Summary[C]$
        for each $R \in T_2.Tags$ do
        \\ adds a new row to T with tag R
            T.addNewRow(R)
            T.Rows[R][C]$\leftarrow T_2.Rows[R][C]$
        endfor
    endfor
    $T.Tags \leftarrow (T_1.Tags \cup T_1.Tags)$
    return $T$
end
```

Figure 9 Merge algorithm

4.2. Mapping Composition Process

Two mappings $M_{A\to B}$ and $M_{B\to C}$ can be composed when there exists a common relation between $Target(M_{A\to B})$ and $Source(M_{B\to C})$. In order to compose two mappings, first the mappings are converted into triples. Let the triple representation of the two mappings, $M_{A\to B}$ and $M_{B\to C}$ be $< T_{SAB}, T_{TAB}, \theta_{AB} >$ and $< T_{SBC}, T_{TBC}, \theta_{BC} >$, respectively. The source tableaux T_{SAB} and T_{SBC} are merged into a larger tableau T_{SAC}. T_{SAC} contains all the rows and columns of T_{SAB} and T_{SBC}. Using information of θ_{AB} and the common tag between T_{TAB} and T_{SBC}, a direct link is made between the mapping tables of B to the relations of A in T_{SAC}. Finally, the row and columns corresponding to the common relation is eliminated from T_{SAC}. Now, T_{SAC} becomes the source tableaux of the desired composed mapping $M_{A\to C}$.

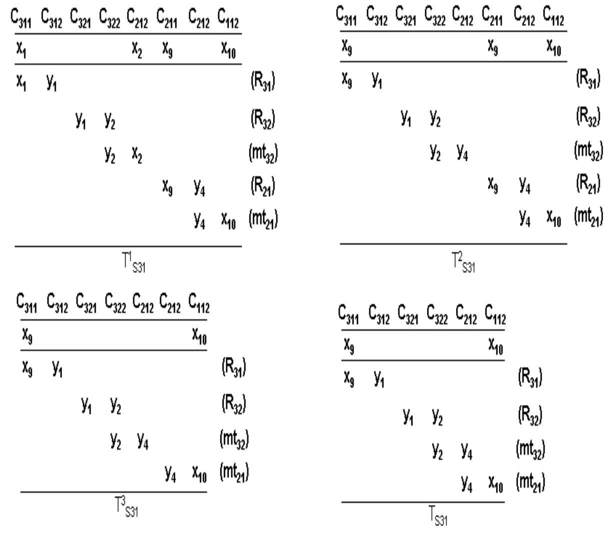
Since θ_{AB} and θ_{BC} are two functions and the domain of θ_{BC} and co-domain of θ_{AB} are the same, they can be composed. Let $\theta_{AC} = \theta_{AB} \circ \theta_{BC}$. θ_{AC} becomes the homomorphism for triple representation of $M_{A\to C}$. The target tableau T_{TAC} is computed from T_{TBC}, θ_{AC} and T_{SAC}. The composition process is explained below with the scenario of example 8. The tableau representation of the mappings of $M_{3\to 2}$ and $M_{2\to 1}$ in Example 8 is shown in Figure 8.

Consider the mappings and their triple representation of Figure 8. Let the composition of mapping $M_{3\to 2}$ and $M_{2\to 1}$ be $M_{3\to 1}$ which is equivalent to the triple $< T_{S31}, T_{T31}, \theta_{31} >$. In the following it is explained how to compute T_{S31},T_{T31} and θ_{31} from the given components of the given triple. The triple $< T_{S31}, T_{T31}, \theta_{31} >$ is then converted to $M_{3\to 1}$.

The steps for the construction of the source tableau, T_{S31} are described as follows:

1. First T_{S32} and T_{S21} is merged together to get T^1_{S31} (which is the initial version of T_{S31}) as shown in Figure 10. The algorithm to merge two tableaux is shown in Figure 9. During the time of merging all the Rows, Columns, Tags, and Summaries of T_{S32} and T_{S21} are simply copied in T^1_{S31}.
2. Next, a common relation between T_{T32} and T_{S21} is identified. In this example the relation is R_{21} of peer P_2.
3. For each column C of the common relation R_{21}, if $(C, C') \in \theta_{32}$ where C' is a column of a arbitrary relation R of P_3, the value of $T^1_{S31}.Rows[R][C']$ is updated by the value of $T^1_{S31}.Rows[R_{21}][C]$. This step creates a link between the mapping tables of P_2 to the relations of P_3. If $T^1_{S31}.Rows[R_{21}][C]$ be a distinguished variable then $T^1_{S31}.Summary[C']$ is also updated by the value $T^1_{S31}.Rows[R_{21}][C]$. In the example, column C_{211} of R_{21} is mapped to C_{311} of R_{31} by θ_{32}, therefore, $T^1_{S31}.Rows[R_{31}][C_{311}]$ is updated by x_9 (i.e. the value of $T^1_{S31}.Rows[R_{21}][C_{211}]$). Since x_9 is a distinguished variable, $T^1_{S31}.Summary[C_{311}]$ is also updated by the value x_9. After taking similar action for every columns of R_{21} in T^1_{S31}, T^1_{S31} takes the shape of T^2_{S31} depicted in Figure 10.
4. Now, the common relation R_{21} is eliminated from T^2_{S31}. At first, the columns having non-null values only in the row with R_{21} are deleted. Deleting a column from a tableau T means deleting it from T.Columns, T.Rows and T.Summary. e.g. the column C_{211} is a candidate for deletion. Since the column C_{212} has a variable in the row with tag m_{21}, it is not a candidate for deletion. After deleting the column C_{211}, the row with tag R_{21} is deleted from T^2_{S31}. After eliminating the R_{21}, T^2_{S31} becomes T^3_{S31} as shown in Figure 10.
5. Finally, the redundant columns of T^3_{S31} (i.e. the columns with the same name) are removed. There are two columns in T^3_{S31} with the name C_{212}. So they are merged into a single column to get the final T_{S31} as shown in Figure 10.

Steps for computing of target tableau, T_{T31}, and the homomorphism, θ_{31} :



C_{311}	C_{312}	C_{321}	C_{322}	C_{212}	C_{211}	C_{212}	C_{112}	
x_1				x_2	x_9		x_{10}	
x_1	y_1							(R_{31})
		y_1	y_2					(R_{32})
			y_2	x_2				(mt_{32})
					x_9	y_4		(R_{21})
						y_4	x_{10}	(mt_{21})

T^1_{S31}

C_{311}	C_{312}	C_{321}	C_{322}	C_{212}	C_{211}	C_{212}	C_{112}	
x_9					x_9		x_{10}	
x_9	y_1							(R_{31})
		y_1	y_2					(R_{32})
			y_2	y_4				(mt_{32})
					x_9	y_4		(R_{21})
						y_4	x_{10}	(mt_{21})

T^2_{S31}

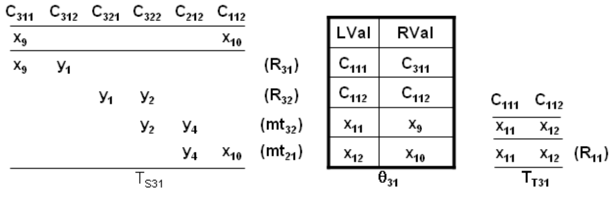
C_{311}	C_{312}	C_{321}	C_{322}	C_{212}	C_{212}	C_{112}	
x_9						x_{10}	
x_9	y_1						(R_{31})
		y_1	y_2				(R_{32})
			y_2	y_4			(mt_{32})
					y_4	x_{10}	(mt_{21})

T^3_{S31}

C_{311}	C_{312}	C_{321}	C_{322}	C_{212}	C_{112}	
x_9					x_{10}	
x_9	y_1					(R_{31})
		y_1	y_2			(R_{32})
			y_2	y_4		(mt_{32})
				y_4	x_{10}	(mt_{21})

T_{S31}

Figure 10 Different phases for the construction of T_{S31}



C_{311}	C_{312}	C_{321}	C_{322}	C_{212}	C_{112}	
x_9					x_{10}	
x_9	y_1					(R_{31})
		y_1	y_2			(R_{32})
			y_2	y_4		(mt_{32})
				y_4	x_{10}	(mt_{21})

T_{S31}

LVal	RVal
C_{111}	C_{311}
C_{112}	C_{112}
x_{11}	x_9
x_{12}	x_{10}

θ_{31}

C_{111}	C_{112}	
x_{11}	x_{12}	
x_{11}	x_{12}	(R_{11})

T_{T31}

Figure 11 Triple $< T_{S31}, T_{T31}, \theta_{31} >$

1. First, a tableau T_{T31} is created and is initialized with the values of T_{T21}. An empty homomorphism θ_{31} is also created.
2. For each column C of T_{T21}, the following steps are taken
   - A column B that is mapped to C by the homomorphism θ_{21} is identified.
   - If B has an entry in $T_{S31}.Summary$ then (C, B) and $(T_{T21}.Summary[C]$ $, T_{S31}.Summary[B])$ pairs are added to θ_{31}. e.g. when C is C_{112}, B is also C_{112}. $T_{S31}.Summary[C_{112}]$ is nonempty and its value is x_{10}. So, (C_{112}, C_{112}) and (x_{12}, x_{10}) are added to θ_{31}.
   - If B column is Null in $T_{S31}.Summary$ then a column A is identified that is mapped to B by the homomorphism θ_{32}. If A column has an entry in $T_{S31}.Summary$ then (C, A) and $(T_{T21}.Summary[C], T_{S31}.Summary[A])$ pairs are added to θ_{31}. e.g. when C is C_{111} then B is C_{211}. But there is no entry in $T_{S31}.Summary$ under the column C_{211}. So, it is further checked which column is mapped to C_{211} by θ_{32}. It is found as C_{311}. $T_{S31}.Summary[C_{311}]$ is nonempty and its value is x_9. So, (C_{111}, C_{311}) and (x_{11}, x_9) are added to θ_{31}.
   - If neither B nor A can be found in $T_{S31}.Summary$ for the column C, C is deleted from T_{T31}.

Final θ_{31} and T_{T31} are shown in Figure 11. T_{S31}, T_{T31} and θ_{31} forms a triple and are then converted as normal bi-level mapping expression. In our example, when the triple $< T_{S31}, T_{T31}, \theta_{31} >$ is converted to normal expression, the mapping becomes as follows:

$$M_{3\to1} : \pi_{C_{311},C_{112}}(R_{31} \bowtie_{C_{312}=C_{321}} R_{32}) \underset{R_{11}}{\overset{\{mt_{32},mt_{21}\}}{\rightsquigarrow}}$$

The algorithm for composing two bi-level mappings is presented in Figure 12.

5. Chain Composition of Mappings

Consider a list $P_1, P_2, \ldots, P_n$ of peers organized in a way that peer P_i is connected to peer P_{i+1} by bi-level mappings, where i ranges from 1 to $n-1$. The list is said to form a chain, say ξ, between P_1 and P_n. A direct mapping between P_1 and P_n is said to be equivalent to the ordered set of mappings along the chain ξ if the two have the same effect in terms of query translation between P_1 and P_n.

5.1. Algorithm for chain composition

Usually an algorithm is designed considering the inputs for it to be locally available. However, in a P2P system each peer stores only the mappings that involve itself and its immediate acquaintances. If a small network is considered with a small number of mappings, it may be reasonable to send all the mappings to a single peer and perform all the necessary computation there. But in reality there may be tens or even hundreds of peers in the network. Size of the mapping tables (which is a part of the bi-level mappings) may be huge in some cases. Our algorithm distributes its computation among the peers on a given chain. In addition to that it delivers the results in a streaming fashion.

For the description of the algorithm, a simple running example is used that involves a chain $\xi = P_1, P_2, \ldots, P_6$ as shown in Figure 13.a. A dashed arrow from peer P_i $(i \in [1,5])$ to peer P_{i+1} indicates that P_i stores mappings that maps P_i to P_{i+1}. The label on the dashed arrow denotes the set of mapping among the acquainted peers. The algorithm runs in each of the peer in a chain.

The algorithm is initiated by having even positioned peers to send their mappings to their respective previous peer in the chain except the last peer. The solid arrows in Figure 13.b denotes the

```
Algorithm Compose($m_{A\to B}, m_{B\to C}$)
Input: Two bi-level mappings $m_{A\to B}$ and $m_{B\to C}$
Output: A Tableau that is the source tableau of a Triple $T$ where
        $T$ is the tableau representation of the mapping obtained
        from the composition of $m_{A\to B}$ and $m_{B\to C}$
begin
    $< T_{SAB}, T_{TAB}, \theta_{AB} > \leftarrow$ ConvertMappingToTriple($m_{A\to B}$)
    $< T_{SBC}, T_{TBC}, \theta_{BC} > \leftarrow$ ConvertMappingToTriple($m_{B\to C}$)

    // Creation of $T_{SAC}$
    $T_{SAC} \leftarrow$ MergeTableau($T_{SAB}, T_{SBC}$)
    $CommTag \leftarrow T_{TAB}.Tags \cap T_{SBC}.Tags$
    for each $C \in T_{TAB}.Columns$ do
        $CommTagV \leftarrow T_{SAC}.Rows[CommTag][C]$
        $MapColInA \leftarrow \theta_{AB}(C)$
        for each $R \in T_{SAB}.Tags$ do
            if $T_{SAC}.Rows[R][MapColInA] \neq Null$
                $T_{SAC}.Rows[R][MapColInA] \leftarrow CommTagV$
                if $T_{SAC}.Rows[CommTag][C] \in XVar$
                    $T_{SAC}.Summary[MapColInA] \leftarrow CommTagV$
                else
                    $T_{SAC}.Summary[MapColInA] \leftarrow Null$
                endif
            endif
        endfor
        $CandidateColumnForDelete \leftarrow True$
        for each $R \in T_{SBC}.Tags$ do
            if $R \neq CommTag$ AND $T_{SBC}.Rows[R][C] \neq Null$
                $CandidateColumnForDelete \leftarrow False$
            endif
        endfor
        if $CandidateColumnForDelete = True$
            $T_{SAC}.DeleteColumn(C)$
        endif
    endfor
    $T_{SAC}.DeleteRow(CommTag)$
    $T_{SAC}.RemoveRedundantCol()$

    // Creation of $\theta_{AC}$ and $T_{TAC}$
    $T_{TAC} \leftarrow new\ Tableau()$
    $T_{TAC} \leftarrow T_{TBC}$
    $\theta_{AC} \leftarrow$ new $Homomorphism()$
    for each $Col \in T_{TBC}.Columns$ do
        $foundMapped \leftarrow False$
        $mapColInB \leftarrow \theta_{BC}.MappedTo(Col)$
        if $T_{SAC}.Summary[mapColInB] \neq Null$
            $foundMapped \leftarrow True$
            $mappedCol \leftarrow mapColInB$
        else
            $mapColInA \leftarrow \theta_{BC}.MappedTo(mapColInB)$
            if $T_{SAC}.Summary[mapColInA] \neq Null$
                $foundMapped \leftarrow True$
                $mappedCol \leftarrow mappedColInA$
            endif
        endif
        if $foundMapped = True$
            $lVar \leftarrow T_{TBC}.Summary[Col]$
            $rVar \leftarrow T_{SAC}.Summary[mappedCol]$
            $\theta_{AC}.Add(Col, mappedCol)$
            $\theta_{AC}.Add(lVar, rVar)$
        else
            $T_{TAC}.DeleteColumn(Col)$
        endif
    endfor

    // Synthesis and conversion of $T_{SAC}$, $\theta_{AC}$ and $T_{TAC}$
    $m_{A\to C} \leftarrow ConvertTripleToMap(< T_{SAC}, T_{TAC}, \theta_{AC} >)$
    return $m_{A\to C}$
end
```

Figure 12 Compose Algorithm

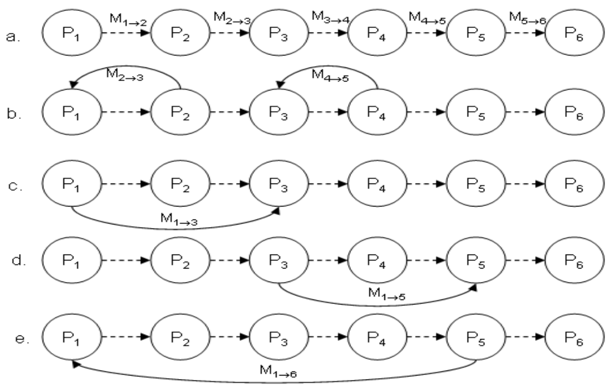


Figure 13 Steps of ChainCompose Algorithm

sending of the mappings. P_2 sends $M_{2\to3}$ to P_1 and in parallel P_4 sends $M_{4\to5}$ to P_3. After receiving $M_{2\to3}$, P_1 composes it with its own $M_{1\to2}$ to produce $M_{1\to3}$. In parallel, P_3 also computes $M_{3\to5}$. P_3 then waits for $M_{1\to3}$ from P_1. After receiving $M_{1\to3}$ from P_1, P_3 composes it with $M_{3\to5}$ to produce $M_{1\to5}$ and sends $M_{1\to5}$ to P_5. P_5 composes $M_{1\to5}$ and $M_{5\to6}$ to generate $M_{1\to6}$. Finally, P_5 sends $M_{1\to6}$ to P_1. P_1 can now use $M_{1\to6}$ as a direct mapping between itself and P_6.

6. Related Work

In the following, some related works regarding the settings of peer database management systems are discussed.

Hyperion system [5] addresses the problem of mapping data in P2P systems where different peers may use different values to identify or describe the same data. Hyperion relies on mapping tables that list pairs of corresponding values for search domains that are used in different peers. Mapping tables provide the foundation for exchanging information between peers. In terms of query answering, Hyperion offers a mechanism [6] that rewrite queries between peers by considering the mapping tables.

The Piazza system [4] provides a solution to peer data management system where the single logical schema of data integration systems is replaced by a set of mediator schemas that are interlinked to define semantic mappings between the peer schemas. Piazza uses two data integration formalisms local-as-view (LAV) and global-as-view (GAV) for peer mappings. GAV is used to define relations of the mediator's schema over the relations in the sources and LAV is used to define relations in the sources over the mediated schema. In Piazza, a reformulation algorithm for query processing is presented that addresses both

GAV and LAV mappings. However, the Piazza system considers only the schema-level heterogeneity among the peers.

The authors of [7] give FOL (First Order Logic) based semantics for interpreting p2p systems. They also present an alternative approach based on epistemic logic and compare these two approaches. The proposed semantics of the bi-level mappings follows the logical formalism from [7].

The SASMINT system [19] provides a solution for supporting interoperability infrastructures that enables sharing and exchange of data among diverse sources. The system mainly finds and resolves syntactic, semantic, and structural conflicts among schemas and matches schemas automatically. This system can be very useful to discover mappings between peers automatically, which we assume in this paper that mappings are exist. The system mainly discovers mappings between two peers which are directly connected. However, in this paper the mappings are discovered between two peers which are connected through other peers in the path.

Authors in [20] proposed a continuous query processing mechanism from heterogeneous data streams. There are two different types of data streams: a relational stream and an XML. The authors mainly assumes that data in different sources use the same vocabularies. In this paper, we consider that schemas as well as data in sources may differ.

Authors in [21] proposed schema mapping and query translation mechanism in heterogeneous P2P databases. The peers use XML databases. In order to translate queries across peers, each peer provides correspondences between its schema. Authors also develop an algorithm for inferring mapping rules from informal schema correspondence. Note that the mappings only consider schema mappings. We consider both schema level and data level mappings between peers. This paper also shows the composition mappings considering the bi-level mappings.

The work of Kementsietsidis et al [5] shows how data level heterogeneity in P2P system can be handled using mapping tables. Mapping tables contains a set of data associations between data values in two peer databases. In that paper, semantic and algorithmic issues related to the use of mapping tables are addressed. The authors describe how the mapping tables can be treated as mapping constraints on the exchange of information between peers. They produce a technique for composing new mapping tables from the existing one.

Given semantic mappings between data sources A and B, and between B and C, it is always desirable to generate a direct mapping between A and C. This type of operation is named Mapping composition. In [17] the authors investigates the theoretical aspects of mapping composition. They also provide an algorithm that compose two GLAV mappings. They dealt with only schema-level mappings. Our mapping composition approach considers both schema and data level mappings at the same time. Definability and computational complexity of the composition of two schema mappings is discussed in [18].

Considering the sharing of data based on semantic meaning, authors in [22] describe the semantic annotation process for integrating university libraries. The system mainly annotates the documents published and distributed throughout the Web. However, we considered the semantic meanings of the data stored in the databases with different schemas and data vocabularies. Authors in [23] focuses on an efficient data aggregation approach from various sensors data sources for a decision modeling. However, the system assumes that the data sources are homogeneous with respect to data and schema.

7. Conclusion

This paper presented a mapping semantics, called bi-level mapping, combining the schema-level and the data-level mappings necessary for resolving heterogeneity among peers in a peer data management system. This bi-level mappings allow peers efficient data sharing facilities that peers miss considering only schema or data level mappings. The bi-level mapping is based on the tableau representation. The paper also provided an algorithm for the the composition of two bi-level mappings. Moreover, a distributed algorithm is proposed for composing the bi-level mappings between two end peers in a chain of peers that are connected through the bi-level mappings.

A future goal is to investigate the composition of mappings considering the dynamic behavior of peers. Further, we are interested to evaluate the whole process in a large peer data management system.

References

[1] J. D. Ullman. Information Integration Using Logical Views. *Proc. of the Int'l Conf. on Database Theory (ICDT)*, pp. 19-40, 1997.

[2] P. Ziegler and K. R. Dittrich. Three Decades of Data Integration - All Problems Solved? *Proc. of the World Computer Congress (WCC)*, pp. 3-12, 2004.

[3] A. V. Aho, Y. Sagiv, and J. D. Ullman. Efficient optimization of a class of relational expressions. In *ACM Transactions on Database Systems (TODS)*, Vol. 4, No. 4, pp. 435-454, 1979.

[4] A. Y. Halevy, Z. G. Ives, J. Madhavan, P. Mork, D. Suciu, and I. Tatarinov. The Piazza Peer Data Management System. In *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 7, pp. 787-798, 2004.

[5] A. Kemensietsidis, M. Arenas, and R. J. Miller. Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues. Proc. *ACM SIGMOD international conference on Management of data*, pp. 325-336, 2003.
[6] A. Kementsietsidis and M. Arenas. Data sharing through query translation in autonomous sources. Proc. *International Conference on Very Large Data Bases*, Vol 40, pp. 468-479, 2004.
[7] D. Calvanese, G. D. Giacomo, M. Lenzerinin, and R. Rosati. Logical Foundations of Peer-To-Peer Data Integration. Proc. *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 241-251, 2004.
[8] M. Lenzerini. Data Integration: A Theoretical Prespective. Proc. *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 233-246, 2002.
[9] J.D. Ullman. Information Integration Using Logical Views. In *Theoretical Computer Dcience*, Vol. 239, No. 2, pp. 189-210, 2000.
[10] A. Levy, A. Rajaraman, and J. Ordille. Querying Heterogeneous Information Sources using Source Descriptions. Proc. *22th International Conference on Very Large Data Bases*, pp. 251-262, 1996.
[11] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. Proc. *Information Processing Society of Japan*, pp. 7-18, 1994.
[12] J. D. Ullman. Information Integration Using Logical Views. In *Theoretical Computer Science*, Vol. 239, No. 2, pp. 189-210, 2000.
[13] M. Boyd, S. Kittivoravitkul, C. Lazanitis, P.J. McBrien and N. Rizopoulos. AutoMed: A BAV Data Integration System for Heterogeneous Data Sources. In *Lecture Notes in Computer Science*, Vol. 3084, pp. 82-97, 2004.
[14] R. J. Miller, M. Hernndez, L. M. Haas, L. Yan, C. T. Howard Ho, R. Fagin, L. Popa. The Clio Project: Managing Heterogeneity. In *SIGMOD Record*, Vol. 30, No. 1, pp. 78-83, 2001.
[15] M. A. Rahman, M. M. Masud, I. Kiringa, and A. El Saddik. Bi-Level Mapping: Combining Schema and Data Level Heterogeneity in Peer Data Sharing Systems. *Proc. Alberto Mendelzon Workshop on Foundations of Data Management*, Vol. 450, 2009.
[16] M. A. Rahman, M. M. Masud, I. Kiringa, and A. El Saddik. A Peer Data Sharing System Combining Schema and Data Level Mappings. In *International Journal of Semantic Computing*, Vol 3, No. 1, pp 105-129, 2009.
[17] J. Madhavan and A. Y. Halevy. Composing mappings among data sources. *Proc. 29th International conference on Very large data bases*, Vol. 29, pp. 572-583, 2003.
[18] R. Fagin, G. Kolaitis, L. Popa, and W. C. Tan. Composing schema mappings: Second-order dependencies to the rescue. In *Journal of ACM Transactions on Database Systems*, Vol. 30, No. 4, pp. 994-1055, 2005.
[19] O. Unal and H. Afsarmanesh. Semi-automated schema integration with SASMINT. In *Journal of Knowledge and Information System Journal*, Vol. 23, No. 1, pp. 99-128, 2010.
[20] H. Lee and W. Lee. Consistent collective evaluation of multiple continuous queries for filtering heterogeneous data streams In *Journal of Knowledge and Information System*, Vol. 22, No. 2, pp. 185-210, 2010.
[21] A. Bonifati, E. Chang, T. Ho, L. V. S. Lakshmanan, R. Pottinger, and Y. Chung. Schema mapping and query translation in heterogeneous P2P XML databases. In *International Journal on Very Large Data Bases*, Vol. 19, No. 2, pp. 231-256, 2010.
[22] H. S. AL-Obaidy and A. Al Heela. Annotation: An Approach for Building Semantic Web Library. Applied Mathematics and Information Sciences, Vol. 6 No. 1, pp. 133-143, 2012.
[23] W. Sung and M. Tsai. Multi-Sensor Wireless Signal Aggregation for Environmental Monitoring System via Multi-bit Data Fusion. Applied Mathematics and Information Sciences, Vol. 5, No. 3, pp. 589-603, 2011.

Md. Anisur Rahman received his PhD degree in Computer Science at the University of Ottawa, Canada. He completed his Masters from Asian Institute of Technology, Thailand, in 2000. He is also a faculty at Computer Science Department, Khulna University, Bangladesh. His research interests are data integration, schema mappings, model management, He has published several research papers at international journals and conference proceedings.

Mehedi Masud received his PhD in Computer Science from the University of Ottawa, Canada. He is an Assistant Professor at the Department of Computer Science, Taif University, KSA. His research interests include issues related to P2P and networked data management, query processing and optimization, eHealth, and information security. He has published several research papers at international journals and conferences. He has served as a member of the technical committees of several international conferences and workshops. He is on the editorial board of some journals including Journal of Internet and Information Systems (JIIS), Journal of Engineering and Computer Innovations, and Journal of Software (JWS). He served as a guest editor for Journal of Computer Science and Information Science (ComSIS).