On the Complexity of Computing Peer Agreements for Consistent Query Answering in Peer-to-Peer Data Integration Systems

Gianluigi Greco
Dip. di Matematica
Università della Calabria
87030 Rende - Italy
ggreco@mat.unical.it

Francesco Scarcello
DEIS
Università della Calabria
87030 Rende - Italy
scarcello@deis.unical.it

ABSTRACT

Peer-to-Peer (*P2P*) data integration systems have recently attracted significant attention for their ability to manage and share data dispersed over different peer sources. While integrating data for answering user queries, it often happens that inconsistencies arise, because some integrity constraints specified on peers' global schemas may be violated. In these cases, we may give semantics to the inconsistent system by suitably "repairing" the retrieved data, as typically done in the context of traditional data integration systems. However, some specific features of *P2P* systems, such as peer autonomy and peer preferences (e.g., different source trusting), should be properly addressed to make the whole approach effective. In this paper, we face these issues that were only marginally considered in the literature. We first present a formal framework for reasoning about autonomous peers that exploit individual preference criteria in repairing the data. The idea is that queries should be answered over the best possible database repairs with respect to the preferences of all peers, i.e., the states on which they are able to find an agreement. Then, we investigate the computational complexity of dealing with peer agreements and of answering queries in *P2P* data integration systems. It turns out that considering peer preferences makes these problems only mildly harder than in traditional data integration systems.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: systems—*Relational databases*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

General Terms

Theory, Management

CIKM'05, October 31–November 5, 2005, Bremen, Germany.
Copyright 2005 ACM 1-59593-140-6/05/0010 ...$5.00.

Keywords

Peer-to-Peer Systems, Data Integration Systems

1. INTRODUCTION

Peer-to-Peer (*P2P*) data integration systems are networks of *autonomous* peers that have recently emerged as an effective architecture for decentralized data sharing, integration, and querying. Indeed, *P2P* systems offer transparent access to the data stored at (the sources of) each peer p, by means of the global schema equipped with p for modeling its domain of interest; moreover, pair of peers with the same domain of interest one peer and the system is in charge of accessing each peer containing relevant data separately, and combining local results into a global answer by suitably exploiting the mapping rules.

P2P systems can be considered the natural evolution of traditional data integration systems, which have received considerable attention in the last few years, and which have already become a key technology for managing enormous amounts of information dispersed over many data sources.

In fact, *P2P* systems have attracted significant attention recently, both in the development of efficient distributed algorithms for the retrieval of relevant information and for answering user queries (see, e.g., [9, 21, 12, 13]), and in the investigation of its theoretical underpinnings (see, e.g., [16, 3, 20, 11, 9, 5]).

In this paper, we continue along this latter line of research, by investigating some important theoretical issues. In particular, we consider an expressive framework where *integrity constraints* are specified on peer schemas in order to enhance their expressiveness, so that each peer can be in fact considered a completely specified data integration system. In this scenario, it may happen that data at different peers are mutually *inconsistent*, i.e., some integrity constraints are violated after the integration is carried out; then, a "repair" for the *P2P* system has to be computed [5, 17]. Roughly speaking, repairs may be viewed as insertions or deletions of tuples at the peers that are able to lead the system to a consistent state.

Our aim is to deal with data integration in *P2P* systems, by extending some of the ideas described in previous studies on merging mutually inconsistent databases into a single consistent theory [2, 14] and on repairing individual data integration systems [8, 6, 4, 10].

Indeed, in order to be effective in this framework, the repair approach should consider the peculiarities of *P2P* systems and, specifically, the following two issues:

- In practical applications, peers often have an a-priori knowledge about the reliability of the sources that, in turn, determines their criteria for computing repairs. That is, peers will rarely delete tuples coming from highly reliable sources, and will try to solve conflicts by updating the less reliable sources only.
- Peers are autonomous and not benevolent: they rarely disregard their individual preferences in order to find an agreement with other peers on the way the repair should be carried out. Therefore, the presence of possibly contrasting interests of selfish peers should be accounted for, when answering user queries.

Despite the wide interest in this field, none of the approaches in the literature considered the issue of modeling the autonomy of the peers in providing a semantics for the system, and therefore they implicitly assume that all the peers act cooperatively in the network. Moreover, the possibility of modeling peer preferences has been rarely considered in previous studies, even though it has been widely recognized to be a central issue for the design of quality-aware integration systems (cf. [17]). Indeed, the first and almost isolated attempt is in [5], where the authors considered trust relationships among peers in a simplified setting in which the system does not transitively propagate information through peers. Actually, an extension to the case of transitive propagations is also argued, but peers autonomy is not considered, and query answering is undecidable in presence of loops.

In this paper, we face the above issues by introducing a formal framework for reasoning about autonomous peers that exploit individual preference criteria in repairing data. In summary, our contributions are the following:

▷ We preliminary introduce a framework for *P2P* data integration systems, where each peer is equipped with integrity constraints on its global schema. The model is simple yet very expressive, since each peer is assumed to be in turn a data integration system. The semantics of a *P2P* system is defined in terms of suitable databases for the peers, called *models*. We show that checking whether a system has a model can be done efficiently.

▷ We propose an approach to the repair of inconsistent *P2P* systems that focuses on data stored at the sources, rather than on the global schema (following the approach described by [15] for the standard data integration setting). This is particularly suited for dealing with peers, as their preferences are typically expressed over the sources. Indeed, if repairs were considered on the global schema, suitable reformulations and translation of the preferences would be required.

▷ We investigate the effect of considering individual preferences on the semantics of *P2P* database integration systems. The idea is that queries should be answered over the best possible database repairs with respect to the preferences of all peers, i.e., over the states on which they are able to find an agreement. Unfortunately, but not surprisingly, it turns out that considering autonomous peers gives rise to scenarios where they are not able to find any agreement on the way the integration should be done.

▷ The above result motivates the subsequent study of the complexity of dealing with peer agreements and of answering queries in such *P2P* data integration systems. We show that checking whether a given database is an agreed repair is a difficult task, since it is complete for the class co-NP. Moreover, the complexity of computing an agreement turns out to be complete for the functional class $\mathrm{FP}^{\mathrm{NP}}$. Finally, we study the complexity of computing consistent answers and show that this problem is Δ_2^{P}-complete. It follows that our approach for handling preferences in *P2P* systems is just mildly harder than the basic data integration framework, where in fact query answering lies at the first level of the polynomial hierarchy [8], as well.

The rest of the paper is organized as follows. In Section 2, we briefly present some preliminaries on relational databases. In Section 3, we introduce a simple formalization of *P2P* data integration systems and in the subsequent section we enrich it to take care of peers' preferences. The computational complexity of the concept of agreement in query answering is studied in Section 5. Finally, in Section 6 we draw our conclusions.

2. PRELIMINARIES ON RELATIONAL DATABASES

We recall the basic notions of the relational model with integrity constraints. For further background on relational database theory, we refer the reader to [1].

We assume a (possibly infinite) fixed database domain Γ whose elements can be referenced by constants $c_1, \ldots, c_n$ under the *unique name assumption*, i.e. different constants denote different objects. These elements are assumed to be shared by all the peers and are, in fact, the constants that can appear in the *P2P* system.

A *relational schema* (or simply *schema*) $\mathcal{RS}$ is a pair $\langle \Psi, \Sigma \rangle$, where: Ψ is a set of relation symbols, each with an associated arity that indicates the number of its attributes, and Σ is a set of *integrity constraints*, i.e., (first-order) assertions that have to be satisfied by each database instance. We deal with quantified constraints, i.e., first order formulas of the form:

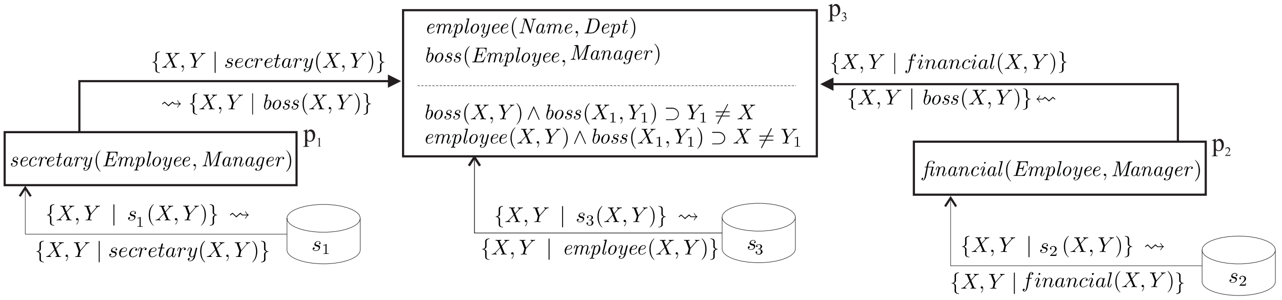
$$\forall \tilde{\mathbf{x}}.\ \bigwedge_{i=1}^{l} A_i \supset \exists \tilde{\mathbf{y}}.\ \bigvee_{j=1}^{m} B_j \vee \bigvee_{k=1}^{n} \phi_k, \qquad (1)$$

where $l+m > 0$, $n \geq 0$, $A_1, \ldots A_l$ and $B_1, \ldots B_m$ are positive literals, $\phi_1, \ldots \phi_n$ are built-in literals, and $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ are lists of distinct variables.

Actually, to keep things simple, we shall assume throughout the paper that $\tilde{\mathbf{y}}$ is empty, thereby dealing with *universally quantified* constraints. We recall here that this kind of constraint covers most of the classical constraints issued on a relational schema, such as keys, functional dependencies, and exclusion dependencies. A brief discussion on how to generalize the results in the paper to other classes of constraints is reported in Section 6.

A *database instance* (or simply *database*) $\mathcal{DB}$ for a schema $\mathcal{RS} = \langle \Psi, \Sigma \rangle$ is a set of facts of the form $r(t)$ where r is a relation of arity n in Ψ and t is an n-tuple of constants from Γ. We denote as $r^{\mathcal{DB}}$ the set $\{t \mid r(t) \in \mathcal{DB}\}$.

A database $\mathcal{DB}$ for a schema $\mathcal{RS}$ is said to be *consistent* with $\mathcal{RS}$ if it satisfies (in the first order logic sense) all constraints expressed on $\mathcal{RS}$.



$employee(Name, Dept)$
$boss(Employee, Manager)$
$boss(X,Y) \wedge boss(X_1,Y_1) \supset Y_1 \neq X$
$employee(X,Y) \wedge boss(X_1,Y_1) \supset X \neq Y_1$
p$_3$
$\{X,Y \mid secretary(X,Y)\}$
$\rightsquigarrow \{X,Y \mid boss(X,Y)\}$
$\{X,Y \mid financial(X,Y)\}$
$\{X,Y \mid boss(X,Y)\} \leftsquigarrow$
p$_1$
$secretary(Employee, Manager)$
p$_2$
$financial(Employee, Manager)$
$\{X,Y \mid s_1(X,Y)\} \rightsquigarrow$
$\{X,Y \mid secretary(X,Y)\}$
s_1
$\{X,Y \mid s_3(X,Y)\} \rightsquigarrow$
$\{X,Y \mid employee(X,Y)\}$
s_3
$\{X,Y \mid s_2(X,Y)\} \rightsquigarrow$
$\{X,Y \mid financial(X,Y)\}$
s_2

Figure 1: The *P2P* system $\mathcal{P}^r$ in Example 1.

A *relational query* (or simply *query*) over $\mathcal{RS}$ is a formula that is intended to extract tuples of elements from the underlying domain of constants Γ. We assume that queries over $\mathcal{RS} = \langle \Psi, \Sigma \rangle$ are *Unions of Conjunctive Queries* (UCQs), i.e., formulas of the form $\{\tilde{\mathbf{x}} \mid \exists \tilde{\mathbf{y}}_1 . conj_1(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}_1) \vee \cdots \vee \exists \tilde{\mathbf{y}}_m . conj_m(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}_m)\}$ where, for each $i \in \{1, \ldots, m\}$, $conj_i(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}_i)$ is a conjunction of atoms whose predicate symbols are in Ψ, and involve $\tilde{\mathbf{x}} = X_1, \ldots, X_n$ and $\tilde{\mathbf{y}}_i = Y_{i,1}, \ldots, Y_{i,n_i}$, where n is the arity of the query, and each X_k and each $Y_{i,\ell}$ is either a variable or a constant in Γ.

Given a database $\mathcal{DB}$ for $\mathcal{RS}$, the answer to a UCQ Q over $\mathcal{DB}$, denoted $Q^{\mathcal{DB}}$, is the set of n-tuples of constants $\langle c_1, \ldots, c_n \rangle$ such that, when substituting each X_i with c_i, the formula $\exists \tilde{\mathbf{y}}_1 . conj_1(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}_1) \vee \cdots \vee \exists \tilde{\mathbf{y}}_m . conj_m(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}_m)$ evaluates to true on $\mathcal{DB}$.

3. DATA INTEGRATION IN P2P SYSTEMS

In this section, we introduce a simple framework for dealing with *P2P* systems. The model is not meant to be a novel comprehensive formalization, since our aim here is to face the problem of finding agreement among peers rather than to investigate new syntactic modeling features.

Therefore, our approach takes basically the same perspective as [9, 11, 5, 17].

3.1 Basic Framework

A *P2P* system $\mathcal{P}$ is a tuple $\langle P, \mathcal{I}, \mathcal{N}, map \rangle$, where P is a non-empty set of distinct peers and $\mathcal{I}$, $\mathcal{N}$ and map are functions whose meaning will be explained below. First, each peer $p \in P$ is equipped with its own data integration system $\mathcal{I}(p)$, which is formalized as a triple $\langle \mathcal{G}_p, \mathcal{S}_p, \mathcal{M}_p \rangle$.

Basically, $\mathcal{S}_p$ is meant to denote the set of sources to which p is allowed to access and is in fact modeled as a relational schema of the form $\mathcal{S}_p = \langle \Psi'_p, \emptyset \rangle$, i.e., there are no integrity constraints on the sources. The structure of the global schema is, instead, represented by means of the schema $\mathcal{G}_p = \langle \Psi_p, \Sigma_p \rangle$, whereas the relationships between the sources and the global schema are specified by $\mathcal{M}_p$, which is a set of *local mapping assertions* between $\mathcal{G}_p$ and $\mathcal{S}_p$. We assume that each assertion is of the form $Q_{\mathcal{S}_p} \rightsquigarrow Q_{\mathcal{G}_p}$, where $Q_{\mathcal{S}_p}$ and $Q_{\mathcal{G}_p}$ are two conjunctive queries of the same arity over the source schema $\mathcal{S}_p$ and the peer schema $\mathcal{G}_p$, respectively.

Example 1 Let us introduce three peers, namely p_1, p_2, and p_3, that constitute the *P2P* scenario that will be used as a running example throughout this paper to illustrate technical definitions.

The global schema $\mathcal{G}_{p_1}$ of peer p_1 consists of the relation predicate $secretary(Employee, Manager)$ (without constraints), the source schema $\mathcal{S}_{p_1}$ consists of the relation symbol s_1, and the set $\mathcal{M}_{p_1}$ of the local mapping assertions is $\{X, Y \mid s_1(X,Y)\} \rightsquigarrow \{X, Y \mid secretary(X,Y)\}$.

As for peer p_2, the schema $\mathcal{G}_{p_2}$ consists of the relation $financial(Employee, Manager)$ (without constraints), the source schema consists of the relation symbol s_2, and $\mathcal{M}_{p_2} = \{X, Y \mid s_2(X,Y)\} \rightsquigarrow \{X, Y \mid financial(X,Y)\}$.

The schema $\mathcal{G}_{p_3}$ of peer p_3 consists of the relations $employee(Name, Dept)$ and $boss(Employee, Manager)$, whose set of constraints contains the assertions (quantifiers are omitted) $employee(X,Y) \wedge boss(X_1,Y_1) \supset X \neq Y_1$ and $boss(X,Y) \wedge boss(X_1,Y_1) \supset Y_1 \neq X$, stating that managers are never employees; the source schema $\mathcal{S}_{p_3}$ comprises the relation symbols s_3; and, the set of the local mapping assertions is $\{X, Y \mid s_3(X,Y)\} \rightsquigarrow \{X, Y \mid employee(X,Y)\}$. □

Each peer $p \in P$ in a *P2P* system $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, map \rangle$ is also equipped with the *neighborhood* function $\mathcal{N}$ providing a set of peers $\mathcal{N}(p) \subseteq P - \{p\}$ containing the peers (called neighbors) who potentially have some information of interest to p. Intuitively, the neighborhood relation determines the structure of a *P2P* system $\mathcal{P}$. Such a structure is better described by the *dependency graph* $\mathrm{G}(\mathcal{P})$ of $\mathcal{P}$, i.e., by a directed graph having P as its set of vertices and $\{(p,q) \mid q \in P \wedge p \in \mathcal{N}(q)\}$ as its set of edges.

In particular, a peer q is in $\mathcal{N}(p)$ iff p is interested in the data exported by q by means of its global schema, i.e., some of the global relations of p can be populated by means of the data coming from q besides the data coming from the sources of p itself. To this aim, $map(p)$ defines the set of *peer mapping assertions* of p.

Each assertion is an expression of the form $Q_q \rightsquigarrow Q_p$, where the peer $q \in \mathcal{N}(p)$ is a neighbor of p, and Q_q and Q_p are two conjunctive queries of the same arity over schemas $\mathcal{G}_q$ and $\mathcal{G}_p$, respectively.

Example 1 (contd.) Let $\mathcal{P}^r = \langle P^r, \mathcal{I}^r, \mathcal{N}^r, map^r \rangle$ be a *P2P* system, where P^r consists of three peers p_1, p_2 and p_3, such that $\mathcal{N}^r(p_1) = \mathcal{N}^r(p_2) = \emptyset$ and $\mathcal{N}^r(p_3) = \{p_1, p_2\}$.

Figure 1 summarizes the structure of the system $\mathcal{P}^r$ by showing, for each peer, its global schema, its source schema, and its local and peer mapping assertions. In particular, notice that the mapping assertions are such that: $map(p_1) = map(p_2) = \emptyset$, and $map(p_3) = \{X, Y \mid financial(X,Y))\} \rightsquigarrow \{X, Y \mid boss(X,Y)\} \cup \{X, Y \mid secretary(X,Y)\} \rightsquigarrow \{X, Y \mid boss(X,Y)\}$. □

A *source database* for a *P2P* system $\mathcal{P}$ is a function $\mathcal{D}$ assigning to each peer $p \in P$ such that $\mathcal{I}(p) = \langle \mathcal{G}_p, \mathcal{S}_p, \mathcal{M}_p \rangle$ a database instance $\mathcal{D}(p)$ for $\mathcal{S}_p$.

A *global database* for $\mathcal{P}$ is a function $\mathcal{B}$ assigning to each peer p a database instance $\mathcal{B}(p)$ for $\mathcal{G}_p$. Usually, we are interested in global databases that can be "retrieved" from a given source, as formalized below.

Given a source database $\mathcal{D}$ for $\mathcal{P}$, a *retrieved global database* for $\mathcal{D}$ is a global database $\mathcal{B}$ that satisfies the mapping assertions $\mathcal{M}_p$ of each peer p, i.e., $\mathcal{B}$ is such that: $\forall p \in P$ and $\forall (Q_{\mathcal{S}_p} \rightsquigarrow Q_{\mathcal{G}_p}) \in \mathcal{M}_p$, it is the case that $Q_{\mathcal{S}_p}^{\mathcal{D}(p)} \subseteq Q_{\mathcal{G}_p}^{\mathcal{B}(p)}$.

We denote by $\mathit{ret}(\mathcal{P}, \mathcal{D})$ the set of all the retrieved global databases for $\mathcal{D}$ in the system $\mathcal{P}$.

Notice that in the definition above we are considering *sound* mappings: data retrieved from the sources by the mapping views are assumed to be a subset of the data that satisfy the corresponding global relation. This is a classical assumption in data integration, where sources in general do not provide all the intended extensions of the global schema, hence extracted data are to be considered sound but not necessarily complete.

Example 1 (contd.) Let $\mathcal{D}^r$ be a source database for the *P2P* system $\mathcal{P}^r$ such that $\mathcal{D}^r(p_1)$ is $\{s_1(\mathit{Albert}, \mathit{Bill})\}$, $\mathcal{D}^r(p_2)$ consists of $\{s_2(\mathit{John}, \mathit{Mary}), s_2(\mathit{Mary}, \ \mathit{Tom})\}$, and $\mathcal{D}^r(p_3) = \{s_3(\mathit{Mary}, \mathit{D1})\}$. Consider also the global database $\mathcal{B}^r$ such that $\mathcal{B}^r(p_1) = \{\mathit{secretary}(\mathit{Albert}, \mathit{Bill})\}$, $\mathcal{B}^r(p_2) = \{\mathit{financial}(\mathit{John}, \mathit{Mary}), \ \mathit{financial}(\mathit{Mary}, \mathit{Tom})\}$ and $\mathcal{B}^r(p_3) = \{\mathit{employee}(\mathit{Mary}, \mathit{D1})\}$. Then, it is easy to see that $\mathcal{B}^r$ is a retrieved database for $\mathcal{D}^r$ in $\mathcal{P}^r$, i.e., $\mathcal{B}^r \in \mathit{ret}(\mathcal{P}^r, \mathcal{D}^r)$.

Note that a global database $\overline{\mathcal{B}}$ whose peer schema for some peer $p \in \{p_1, p_2, p_3\}$ is a superset of $\mathcal{B}^r(p)$ is in $\mathit{ret}(\mathcal{P}^r, \mathcal{D}^r)$ as well - we simply say that $\overline{\mathcal{B}}$ is a superset of $\mathcal{B}^r$. □

3.2 Models of Peer-to-Peer Systems

Given a source database $\mathcal{D}$, it is particular important to investigate whether it is possible to retrieve from $\mathcal{D}$ a database which satisfies the semantics of the network. Therefore, we next define a suitable notion of *model* for a *P2P* system. The approach has been inspired by the autoepistemic approach of [9]; in particular, we assume that peers propagate through mapping assertions only the values they really trust.

Definition 2 Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map} \rangle$ be a *P2P* system, $p \in P$ a peer with $\mathcal{I}(p) = \langle \mathcal{G}_p, \mathcal{S}_p, \mathcal{M}_p \rangle$ and $\mathcal{G}_p = \langle \Psi_p, \Sigma_p \rangle$, and $\mathcal{D}$ a source instance for $\mathcal{P}$. Then, a *p-model* for $\mathcal{P}$ w.r.t. $\mathcal{D}$ is a maximal nonempty set of global databases $\mathbb{M} \subseteq \mathit{ret}(\mathcal{P}, \mathcal{D})$ such that:

1. for each $\mathcal{B} \in \mathbb{M}$, $\mathcal{B}(p)$ satisfies the constraints in Σ_p, and
2. for each assertion $Q_q \rightsquigarrow Q_p \in \mathit{map}(p)$, it holds: $\bigcap_{\mathcal{B}' \in \ \mathbb{M}} Q_q^{\mathcal{B}'(q)} \subseteq \bigcap_{\mathcal{B}' \in \ \mathbb{M}} Q_p^{\mathcal{B}'(p)}$. □

Thus, according to Condition 1, any databases in the p-model satisfies all the integrity constraints issued over the global schema of p; moreover, Condition 2 guarantees that peers communicate only those values that belong to all models, i.e., a *cautious* approach to the propagation has been pursued. Finally we point out that, as for local mapping assertions, peer mapping assertions are assumed to be sound.

Now, given that each peer singles out its models, a notion of model for the whole system can be easily stated.

Definition 3 Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map} \rangle$ be a *P2P* system. A *model* for $\mathcal{P}$ w.r.t. $\mathcal{D}$ is a maximal nonempty set $\mathbb{M} \subseteq \mathit{ret}(\mathcal{P}, \mathcal{D})$ of global databases such that, for each $p \in P$, $\mathbb{M}$ is a p-model. If a model for $\mathcal{P}$ w.r.t. $\mathcal{D}$ exists, we say that $\mathcal{D}$ *satisfies* $\mathcal{P}$, denoted by $\mathcal{D} \models \mathcal{P}$. □

For instance, in our running example, $\mathcal{D}^r$ does not satisfy $\mathcal{P}^r$; indeed, the peer mapping assertions constrain the schema of p_3 to contain in every global database (retrieved from $\mathcal{D}^r$) the tuples $\mathit{boss}(\mathit{Albert}, \mathit{Bill})$, $\mathit{boss}(\mathit{John}, \mathit{Mary})$, $\mathit{boss}(\mathit{Mary}, \mathit{Tom})$, and $\mathit{employee}(\mathit{Mary}, \mathit{D1})$ that violate the integrity constraints over p_3, since *Mary* results to be both an employee and a manger. Therefore, retrieving data from $\mathcal{D}^r$ leads to an inconsistent scenario.

We conclude by noticing that deciding whether a *P2P* system admits a model can be done efficiently. The result can be proven by modifying the techniques in [9], in order to first evaluate all the mappings in the network and then check for the satisfaction of the integrity constraints over peer schemas.

Theorem 4 *Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map} \rangle$ be a* P2P *system, and $\mathcal{D}$ be a database instance for $\mathcal{P}$. Then, deciding whether there is a model for $\mathcal{P}$ w.r.t. $\mathcal{D}$, i.e., $\mathcal{D} \models \mathcal{P}$, is feasible in polynomial time.*

4. DEALING WITH AUTONOMOUS PEERS

As shown in our running example, in general data stored in local and autonomous sources are not required to satisfy constraints expressed on the global schema (for example when a key dependency on $\mathcal{G}$ is violated by data retrieved from the sources). Thus, a *P2P* system may be unsatisfiable w.r.t. a source database $\mathcal{D}$. In this section, we face the problem of solving inconsistencies in *P2P* systems. Specifically, we introduce a semantics for "repairing" a *P2P* system. To this aim, we first provide a model for peer preferences, and then show the impact of these individual preferences on the cost of reaching a global agreed repair.

4.1 Peer Preferences and Repairs

Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map} \rangle$ be a *P2P* system, and $\mathcal{D}$ be a source database instance for $\mathcal{P}$. Next, we define a repair weighting function $w^p_{(\mathcal{P},\mathcal{D})}$ for each peer p, encoding its preferences on candidate repairs of $\mathcal{D}$. Formally, $w^p_{(\mathcal{P},\mathcal{D})}$ is a polynomially-computable function assigning, to each source database instance $\overline{\mathcal{D}}$, a natural number that is a measure of the preference of p on having $\overline{\mathcal{D}}$ as a repair for $\mathcal{D}$ (the lower the number, the more preferred the repair).

As a quite simple, yet natural example of weighting function, we can consider the evaluation of the number of deletions performed to the peer's sources. In this case, we have that $w^p_{(\mathcal{P},\mathcal{D})}(\mathcal{D}') = |\mathcal{D}'(p) - \mathcal{D}(p)|$, which in fact corresponds to the size of the difference between $\mathcal{D}'$ and $\mathcal{D}$ restricted to tuples of peer p. This weighting function is called *cardinality-based* in the following.

Example 1 (contd.) Consider the source databases $\mathcal{D}^r_1$, $\mathcal{D}^r_2$, and $\mathcal{D}^r_3$ such that: $\mathcal{D}^r_1(p_1) = \mathcal{D}^r_2(p_1) = \mathcal{D}^r_3(p_1) = \mathcal{D}^r(p_1)$,

$\mathcal{D}_1^r(p_2) = \{s_2(\mathit{John}, \mathit{Mary})\}$, $\mathcal{D}_2^r(p_2) = \{s_2(\mathit{Mary}, \mathit{Tom})\}$, $\mathcal{D}_3^r(p_2) = \{\}$, $\mathcal{D}_1^r(p_3) = \{\}$, $\mathcal{D}_2^r(p_3) = \{s_3(\mathit{Mary}, \mathit{D1})\}$, and $\mathcal{D}_3^r(p_3) = \{s_3(\mathit{Mary}, \mathit{D1})\}$.

Assume that, for each peer p, $w^p_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}) = |\mathcal{D}(p) - \mathcal{D}^r(p)|$, i.e., she prefers source repairs where the minimum number of tuples is deleted from $\mathcal{D}^r(p)$. Then, $w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_1^r) = w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_2^r) = w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_3^r) = 0$; $w^{p_2}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_1^r) = w^{p_2}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_2^r) = 1$; $w^{p_2}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_3^r) = 2$; $w^{p_3}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_1^r) = 1$; $w^{p_3}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_2^r) = w^{p_3}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_3^r) = 0$. □

The problem of solving inconsistency in "classical" data integration systems has been traditionally faced by providing a semantics in terms of the *repairs* of the global databases that the mapping forces to be in the semantic of the system [4, 7, 6]. Repairs are obtained by means of addition and deletion of tuples according to some minimality criterion.

We next propose a generalization of these approaches to the *P2P* framework, which takes into account peers preferences. To this aim, we focus on finding the proper set of facts at the sources that imply as a consequence a global database satisfying all integrity constraints. Basically, such a way of proceeding allows us to easily take into account information on preferences when trying to solve inconsistency, since repairing is performed by directly focusing on those sources, whose integration has caused inconsistency.

Definition 5 (Repair) Let $\mathcal{P}$ be a *P2P* system, p a peer, and $\mathcal{D}$ and $\mathcal{D}'$ two source databases. We say that $\mathcal{D}'$ is *p-minimal* if $\mathcal{D}' \models \mathcal{P}$, and there exists no source database $\mathcal{D}''$ such that $w^p_{(\mathcal{P},\mathcal{D})}(\mathcal{D}'') < w^p_{(\mathcal{P},\mathcal{D})}(\mathcal{D}')$ and $\mathcal{D}'' \models \mathcal{P}$.

Then, $\mathcal{D}'$ is a *repair* for $\mathcal{P}$ w.r.t. $\mathcal{D}$ if $\mathcal{D}'$ is *p-minimal* for each peer p. □

Example 1 (contd.) It is easy to see that $\mathcal{D}_1^r$, $\mathcal{D}_2^r$, and $\mathcal{D}_3^r$ satisfy $\mathcal{P}^r$ and they are both p_1-minimal. Indeed, peer p_1 has no preferences among the three databases, since $w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_1^r) = w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_2^r) = w^{p_1}_{(\mathcal{P}^r,\mathcal{D}^r)}(\mathcal{D}_3^r) = 0$. Moreover, $\mathcal{D}_1^r$ and $\mathcal{D}_2^r$ are equally preferred by p_2, whereas $\mathcal{D}_2^r$ and $\mathcal{D}_3^r$ are equally preferred by p_3. Therefore, all peers agree on $\mathcal{D}_2^r$, which is thus a repair for $\mathcal{D}^r$ w.r.t. $\mathcal{P}^r$. However, neither $\mathcal{D}_3^r$ is p_2-minimal, nor $\mathcal{D}_1^r$ is p_3-minimal, and thus they are not repairs. □

We next define the semantics of a *P2P* system, in terms of models for those sources on which all the peers agree.

Definition 6 (Agreement) Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map}\rangle$ be a *P2P* system, and $\mathcal{D}$ be an instance for $\mathcal{P}$. The *agreement* for $\mathcal{P}$ w.r.t. $\mathcal{D}$ is the set of all of its models w.r.t. some repair, and will be denoted by $\mathit{Agr}(\mathcal{P}, \mathcal{D})$. □

Example 1 (contd.) $\mathcal{D}_2^r$ is p-minimal, for each peer p, and it is easy to see that the set $\mathit{Agr}(\mathcal{P}^r, \mathcal{D}^r)$ contains all databases belonging to some model for $\mathcal{P}^r$ w.r.t. $\mathcal{D}_2^r$. In particular, it contains the supersets (satisfying the constraints) of the database $\mathcal{B}_2^r$ such that $\mathcal{B}_2^r(p_1) = \{\mathit{secretary}(\mathit{Albert}, \mathit{Bill})\}$, $\mathcal{B}_2^r(p_2) = \{\mathit{financial}(\mathit{Mary}, \mathit{Tom})\}$ and $\mathcal{B}_2^r(p_3) = \{\mathit{boss}(\mathit{Albert}, \mathit{Bill})$, $\mathit{boss}(\mathit{Mary}, \mathit{Tom})$, $\mathit{employee}(\mathit{Mary}, \mathit{D1})\}$. Moreover, no other global database is in $\mathit{Agr}(\mathcal{P}^r, \mathcal{D}^r)$. □

We can finally characterize the answer to a user query in terms of the repairs for the system.

Definition 7 Let $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map}\rangle$ be a *P2P* system, let $\mathcal{D}$ be a source database for it, and let Q be a query over the schema of a peer p. Then, the *answer* to Q is the evaluation of the query over all the possible agreed databases: $\mathit{ans}(Q, p, \mathcal{P}, \mathcal{D}) = \bigcap_{\mathcal{B} \in \ \mathit{Agr}(\mathcal{P},\mathcal{D})} Q_p^{\mathcal{B}(p)}$. □

For instance, in our running example, the answer to the user query $\{X \mid \mathit{boss}(X, Y)\}$ posed over peer p_3, which asks for all employees that have a boss, is $\{\langle \mathit{Albert}\rangle, \langle \mathit{Mary}\rangle\}$, since this query is evaluated over the supersets of the database $\mathcal{B}_2^r$ retrieved from $\mathcal{D}_2^r$ only.

We conclude the section by noticing that $\mathit{Agr}(\mathcal{P}, \mathcal{D})$ is just a formal characterization of the semantics of a *P2P* system. Usually, we are not interested in computing such a set; and, in fact, for practical applications, suitable techniques and optimization algorithms should be investigated to handle inconsistency at query time (in the spirit of, e.g., [10]).

4.2 The Price of Autonomy

Given the framework presented so far, we are in the position of studying the effects of having autonomous peers repairing their source databases according to their own preferences. We next show that, in some cases, peers might not find an agreement on the way the repair has to be carried out. This is a somehow expected consequence of having selfish interested peers in the absence of a global coordination.

Proposition 8 *There exists a P2P system $\mathcal{P}$ and a source database $\mathcal{D}$ such that there is no agreement, i.e., $\mathit{Agr}(\mathcal{P}, \mathcal{D})$ is empty.*

Proof [**Sketch**]. Consider the *P2P* system $\mathcal{P} = \langle P, \mathcal{I}, \mathcal{N}, \mathit{map}\rangle$, where P consists of the peers *challenger* (short: c) and *duplicator* (short: d), that are mutually connected, i.e., $\mathcal{N}(c) = \{d\}$ and $\mathcal{N}(d) = \{c\}$.

Peer c is such that $\mathcal{I}(c) = \langle \mathcal{G}_c, \mathcal{S}_c, \mathcal{M}_c\rangle$, where the schema $\mathcal{G}_c$ consists of predicates $r_c(X)$ and $mr_d(X)$ with constraints $r_c(X) \wedge r_c(Y) \supset X \neq Y$ and $r_c(X) \wedge mr_d(Y) \supset X \neq Y$; the source schema consists of the relation symbol s_c; and $\mathcal{M}_c$ contains only the assertion $\{X \mid s_c(X)\} \rightsquigarrow \{X \mid r_c(X)\}$.

Peer d is such that $\mathcal{I}(d) = \langle \mathcal{G}_d, \mathcal{S}_d, \mathcal{M}_d\rangle$, where the schema $\mathcal{G}_d$ consists of predicates $r_d(X)$ and $mr_c(X)$ with constraints $r_d(X) \wedge r_d(Y) \supset X \neq Y$ and $r_d(X) \wedge mr_c(Y) \supset X = Y$; the source schema consists of the relation symbol s_d; and $\mathcal{M}_d$ contains only the assertion $\{X \mid s_d(X)\} \rightsquigarrow \{X \mid r_d(X)\}$.

Finally, $\mathit{map}(c)$ contains the assertion $\{X \mid r_c(X))\} \rightsquigarrow \{X \mid mr_c(X)\}$, while $\mathit{map}(d)$ contains the assertion $\{X \mid r_d(X))\} \rightsquigarrow \{X \mid mr_d(X)\}$.

Let $\mathcal{D}$ be a source database for $\mathcal{P}$ such that $\mathcal{D}(c) = \{s_c(0), s_c(1)\}$ and $\mathcal{D}(d) = \{s_d(0), s_d(1)\}$. We build four source databases, say $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$, that satisfy $\mathcal{P}$. They are such that: $\mathcal{D}_1(c) = \{\}$, $\mathcal{D}_1(d) = \{s_d(0)\}$; $\mathcal{D}_2(c) = \{\}$, $\mathcal{D}_2(d) = \{s_d(1)\}$; $\mathcal{D}_3(c) = \{s_c(0)\}$, $\mathcal{D}_3(d) = \{\}$; $\mathcal{D}_4(c) = \{s_c(1)\}$, $\mathcal{D}_4(d) = \{\}$. Notice that all the other databases satisfying $\mathcal{P}$ are proper subsets of these ones. Then, by assuming that each peer wants to minimize the number of deletions in $\mathcal{D}$, there exists no source database satisfying $\mathcal{P}$ that is both c-minimal and d-minimal. □

5. THE COMPLEXITY OF QUERY ANSWERING

In the light of Proposition 8, it is particulary relevant to investigate the complexity of dealing with peer agreements and query answering in such *P2P* data integration systems. In this section, we first present some basic problems arising in the proposed framework, and subsequently analyze their computational complexity. This analysis is a fundamental premise to devise effective and optimized implementations.

5.1 Problems

Given a *P2P* system $\mathcal{P}$ and a source database $\mathcal{D}$ for $\mathcal{P}$, we consider the following problems:

- `RepairChecking`: given a source instance $\mathcal{D}'$, is $\mathcal{D}'$ a repair for $\mathcal{P}$ w.r.t. $\mathcal{D}$?
- `AgreementExistence`: is $Agr(\mathcal{P}, \mathcal{D}) \neq \emptyset$?
- `AnyAgreementComputation`: compute a database $\mathcal{B}$ in the agreement $Agr(\mathcal{P}, \mathcal{D})$, if any.
- `QueryOutputTuple`: given a query Q over a peer schema $\mathcal{G}_p$ and a tuple t, is $t \in ans(Q, p, \mathcal{P}, \mathcal{D})$?

Intuitively, `RepairChecking` is the very basic problem of assessing whether a source instance at hand satisfies the data integration system. Then, `AgreementExistence` (and its corresponding computational version `AnyAgreementComputation`) asks for singling out scenarios where some agrement can be in fact computed. Finally, `QueryOutputTuple` represents the problem characterizing the intrinsic complexity of a query answering in the proposed framework; indeed, it is the problem of deciding the membership of a given tuple in the result of query evaluation.

5.2 Results

Our first result is that checking whether all the peers are satisfied by a given source database is a difficult task that is unlikely to be feasible in polynomial time.

Theorem 9 `RepairChecking` *is co-NP-complete. Hardness holds even for cardinality-based weighting functions.*

Proof [Sketch]. *Membership.* Consider the complementary problem of deciding whether there exists a peer p such that $\mathcal{D}'$ is not p-minimal. This problem is feasible in NP by guessing a source database $\mathcal{D}''$ and checking in that *1.* $\mathcal{D}'' \models P$, and *2.* there exists a peer p such that $w^p_{(\mathcal{P},\mathcal{D})}(\mathcal{D}'') < w^p_{(\mathcal{P},\mathcal{D})}(\mathcal{D}')$. In particular, *1.* is feasible in polynomial time because of Theorem 4, and *2.* is feasible in polynomial time because our weighting functions are polynomially computable.

Hardness. Recall that deciding whether a Boolean formula in conjunctive normal form $\Phi = C_1 \wedge \ldots \wedge C_m$ over the variables $X_1, \ldots, X_n$ is not satisfiable, i.e., deciding whether there exists no truth assignments to the variables making each clause C_j true, is a co-NP-hard problem.

We built a *P2P* system $\mathcal{P}^\Phi$ such that: $\mathcal{P}^\Phi$ contains a peer x_i for each variable X_i, a peer c_j for each clause C_j, and the distinguished peer e. The source schema of x_i (resp. c_j) consists of the unary relation s_{x_i} (resp. s_{c_j}), whereas the global schema consists of the unary relation r_{x_i} (resp. r_{c_j}). The source schema of e consists of the unary relations s_e and s_a, whereas its global schema consists of the unary relations r_e and r_a. For each source relation, say s_ℓ, $\mathcal{P}(\Phi)$ contains a local mapping assertion of the form $\{X \mid s_\ell(X)\} \rightsquigarrow \{X \mid r_\ell(X)\}$. Each global relation of the form r_{x_i} is equipped with the constraint $r_{x_i}(X_1) \wedge r_{x_i}(X_2) \supset X_1 = X_2$, stating that each relation must contain one atom at most. Each global relation of the form r_{c_j} is equipped with the constraint $r_{c_j}(tx_i) \wedge r_{c_j}(fx_i) \supset \perp$, where $\perp$ is the empty disjunction, stating that for each variable x_i, r_{c_j} cannot contain both tx_i and fx_i at the same time. Moreover, peer e has also the constraint $r_e(X_1) \wedge r_a(X_2) \supset X_1 = X_2$.

Consider the source database $\mathcal{D}^\Phi$ for $\mathcal{P}^\Phi$ such that: $\mathcal{D}^\Phi(x_i) = \{s_{x_i}(tx_i), s_{x_i}(fx_i)\}$; for each x_i occurring in c_j, $\mathcal{D}^\Phi(c_j) = \{s_{c_j}(tx_i), s_{c_j}(fx_i)\}$; and $\mathcal{D}^\Phi(e) = \{s_e(t), s_e(f), s_a(t)\}$. Notice that due to the constraints issued over peers schemas, any source database $\mathcal{D}'$, with $\mathcal{D}' \models \mathcal{P}^\Phi$, is such that $|\mathcal{D}'(x_i)| \leq 1$, for each x_i. Therefore, the restriction of $\mathcal{D}'$ to the peers of the form x_i is in one-to-one correspondence with a truth-value assignment for Φ, denoted by $\mu(\mathcal{D}')$. Intuitively, the atom $s_{x_i}(tx_i)$ (resp. $s_{x_i}(fx_i)$) means that variable X_i is set to true (resp. false), whereas the atom $s_{c_j}(tx_i)$ means that the clause C_j is true, witnessed by the assignment for the variable X_i occurring in c_j.

Finally, the peers mapping assertions in $\mathcal{P}^\Phi$ are defined as follows. For each variable X_i occurring positively (resp. negatively) in the clause C_j there are exactly two mappings of the form $\{r_{x_i}(tx_i)\} \rightsquigarrow \{r_{c_j}(tx_i)\}$ and $\{r_{x_i}(fx_i)\} \rightsquigarrow \{r_{c_j}(fx_i)\}$ (resp. $\{r_{x_i}(fx_i)\} \rightsquigarrow \{r_{c_j}(tx_i)\}$ and $\{r_{x_i}(tx_i)\} \rightsquigarrow \{r_{c_j}(fx_i)\}$); moreover, for each clause C_j containing variables $X_{j_1}, ..., X_{j_k}$, there exists a mapping $\{r_{c_j}(fx_{j_1}) \wedge \cdots \wedge r_{c_j}(fx_{j_k})\} \rightsquigarrow \{r_e(f)\}$.

Figure 2 shows on the upper part the dependency graph $\mathrm{G}(\mathcal{P}^\Phi)$ for the formula $\Phi = (X_1 \vee X_2) \wedge (X_3) \wedge (X_1 \vee X_3 \vee \neg X_4) \wedge (X_4) \wedge (\neg X_5 \vee \neg X_6 \vee X_7) \wedge (X_4 \vee X_6 \vee X_8)$.

Assume that each peer wants to minimize the number of deletions in $\mathcal{D}^\Phi$. Then, given a source database $\mathcal{D}'$ minimal w.r.t. each peer in $\mathcal{P}^\Phi$ but e, we can show that the above mappings encode an evaluation of the assignment $\mu(\mathcal{D}')$. In particular, it is easy to see that $\mu(\mathcal{D}')$ is a satisfying assignment for Φ if and only if $\mathcal{D}'(e)$ contains the facts $\{s_e(t), s_a(t)\}$, i.e., one fact is deleted from the source of e only. Assume, now, that $\mathcal{D}'$ is such that $\mathcal{D}'(e) = \{s_e(f)\}$, i.e., two facts are deleted from the source of e. Then, $\mathcal{D}'$ is also e-minimal if and only if Φ is not satisfiable. □

Given the above complexity result, one can easily see that `AnyAgreementComputation` is feasible in the functional version of Σ^P_2. Indeed, we can guess in NP a source instance $\mathcal{D}$, build in polynomial time a model $\mathcal{B}$ for $\mathcal{P}$ w.r.t. $\mathcal{D}$ (by construction in Theorem 4), and check in co-NP that $\mathcal{D}$ is minimal for each peer.

Actually, we can do much better. In fact, we next show that the problem is complete for the polynomial time closure of NP, and thus remains at the first level of the polynomial hierarchy.

Theorem 10 `AnyAgreementComputation` *is* FP^{NP}*-complete. Hardness holds even for cardinality-based weighting functions.*

Proof [Sketch]. *Membership.* The problem can be solved by processing peers in a sequential manner. For each peer in $\mathcal{P}$, we can find the minimum value of the associated preference function by means of a binary search, in which at each step we guess in NP a database instance and verify that such a preference holds. After having collected the minimum values for all peers, we conclude with a final guess to get a repair $\mathcal{D}$, and a subsequent check that actually each peer gets its minimum possible value for $\mathcal{P}$ w.r.t. $\mathcal{D}$.

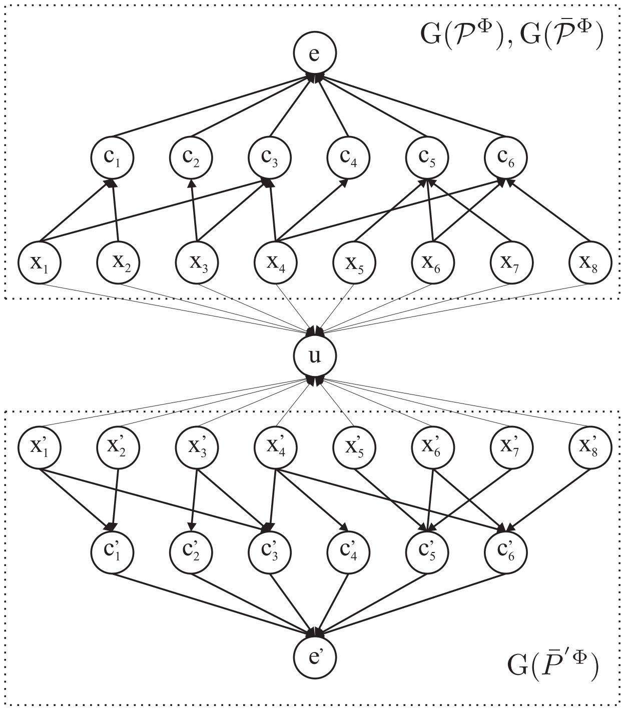


Figure 2: Constructions in Proofs of Complexity Results.

Finally, a model for $\mathcal{P}$ w.r.t. $\mathcal{D}$ can be build in polynomial time (again, by construction in Theorem 4).

Hardness. Let Φ be a boolean formula in conjunctive normal form $\Phi = C_1 \wedge \ldots \wedge C_m$ over the variables $X_1, \ldots, X_n$. Assume that each clause, say C_j, is equipped with a weight $\mathtt{w}_j$ (natural number). Let σ be an assignment for the variables in Φ. Its weight is the sum of the weights of all the clauses satisfied in σ. The problem of computing the maximum weight over any truth assignment, called $\mathtt{MAX}-\mathtt{WEIGHT}-\mathtt{SAT}$, is $\mathrm{FP}^{\mathrm{NP}}$-complete.

Consider again the construction in Theorem 9, and modify $\mathcal{P}^{\Phi}$ as follows. The source schema of peer e consists of the relation s_w, whereas its global schema consists of the relations r_w and r_v, and of the constraint $r_v(X) \wedge r_w(X,Y) \supset \perp$. The local mappings of e is $\{X, Y \mid s_w(X,Y)\} \rightsquigarrow \{X, Y \mid r_w(X,Y)\}$. Moreover, for each clause c_j over variables $X_{j_1}, ..., X_{j_k}$, $map(e)$ contains the assertion $\{r_{c_j}(fx_{j_1}) \wedge \cdots \wedge r_{c_j}(fx_{j_k})\} \rightsquigarrow \{r_v(fc_j)\}$. Let $\bar{\mathcal{P}}^{\Phi}$ be such a modified *P2P* system. Notice that $G(\bar{P}^{\Phi})$ coincides with $G(\mathcal{P}^{\Phi})$ (see again Figure 2).

Consider now the database instance $\bar{\mathcal{D}}^{\Phi}$ for $\bar{\mathcal{P}}^{\Phi}$ obtained by modifying $\mathcal{D}^{\Phi}$ such that $\bar{\mathcal{D}}^{\Phi}(e)$ contains the atoms $s_w(fc_j, 1)$, $s_w(fc_j, 2)$, $...s_w(fc_j, \mathtt{w}_j)$ for each clause c_j. Intuitively, peer e stores $\mathtt{w}_j$ distinct atoms for each clause c_j.

Let $\mathcal{D}''$ be a source instance that satisfies $\bar{\mathcal{P}}^{\Phi}$. As in Theorem 9, the restriction of $\mathcal{D}''$ over the variables is in one-to-one correspondence with a truth assignment for Φ, denoted by $\mu(\mathcal{D}'')$. Then, it is easy to see that peer e must delete in $\mathcal{D}''$ all the $\mathtt{w}_j$ distinct atoms corresponding to a clause C_j that is not satisfied by the assignment $\mu(\mathcal{D}'')$. Therefore, $|\mathcal{D}''(e)| = \sum_{i|\mathcal{C}_i \text{ is false in } \mu(\mathcal{D}'')} \mathtt{w}_i$. Hence, the result easily follows, since computing the source instance that is e-minimal, say $\overline{\mathcal{D}}$, determines the maximum weight over any assignment for Φ as $(\sum_i \mathtt{w}_i) - |\overline{\mathcal{D}}(e)|$. □

We next focus on the `AgreementExistence` problem. Note that membership of this problem in Δ_2^{P} is easy to proven, after the above theorem. However, the reduction for the hardness part we shall exploit here is rather different.

Theorem 11 `AgreementExistence` *is* Δ_2^{P}*-complete. Hardness holds even for cardinality-based weighting functions.*

Proof [Sketch]. Membership is shown with the same line of reasoning of Theorem 10. For the hardness, consider again $\mathtt{MAX}-\mathtt{WEIGHT}-\mathtt{SAT}$, and the Δ_2^P-complete problem of deciding whether it has a unique solution.

Let $\bar{\mathcal{P}}^{\Phi}$ be the *P2P* system built in Theorem 10, and let $\bar{\mathcal{P}}'^{\Phi}$ be a copy of it, obtained by replacing each element (both relations and peers) r in $\bar{\mathcal{P}}^{\Phi}$ by r'. Then, consider the system $\tilde{\mathcal{P}}^{\Phi}$ obtained as the union of $\bar{\mathcal{P}}^{\Phi}$, $\bar{\mathcal{P}}'^{\Phi}$ and a fresh peer u. Figure 2 shows the dependency graph $G(\tilde{\mathcal{P}}'^{\Phi})$.

The local schema of u is empty, while its global schema consists of the unary relation r_u with the constraint $\bigwedge_{i=1}^{n} r_u(bad_i) \supset \perp$. The mapping assertions are as follows. For each variable X_i in Φ, $map(u)$ contains $\{r_{x_i}(tx_i) \wedge r'_{x'_i}(tx_i)\} \rightsquigarrow \{r_u(bad_i)\}$ and $\{r_{x_i}(fx_i) \wedge r'_{x'_i}(fx_i)\} \rightsquigarrow \{r_u(bad_i)\}$. It is worthwhile noting that, for the sake of simplicity, the mapping assertions are slightly more general than those allowed in the usual definition of *P2P* systems, since they involve joins among different peers. However, this is only a syntactical facility, as such a mapping can be easily simulated by introducing a suitable dummy peer.

The idea of the reduction is that, if the same assignment that maximizes the weight of the satisfied clauses is selected for both $\bar{\mathcal{P}}^{\Phi}$ and $\bar{\mathcal{P}}'^{\Phi}$, then $r_u(bad_i)$ is pushed to u (for each i), thereby violating the constraint. Thus, there is a (non-empty) agreement in $\tilde{\mathcal{P}}^{\Phi}$ if and only if there are at least two such assignments. □

We conclude our investigation by observing that query answering is at least as hard as `AgreementExistence`. Indeed, intuitively, if peers are not able to find an agreement in an inconsistent *P2P* system, then the answer to any given query will be empty. Moreover, membership can be proven by the same line of reasoning of Theorem 10, and we thus get the following result.

Theorem 12 `QueryOutputTuple` *is* Δ_2^{P}*-complete. Hardness holds even for cardinality-based weighting functions.*

6. CONCLUSIONS

In this paper, we investigated some important theoretical issues in *P2P* data integration systems. Specifically, we introduced a setting in which peers take into account their own preferences over data sources, in order to integrate data if some inconsistency arise. This seems a natural setting for such kind of systems, which has not been previously investigated in the literature. It turns out that there are scenarios where peers do not find any agreement on the way the repair should be carried out, and where some kind of centralized coordination is required.

Actually, our results show that this coordination comes with a cost and some basic problems are unlikely to be tractable. However, the complexity of the problems studied in this paper are only mildly harder than the corresponding problems in traditional data integration systems.

This is an important feature of our approach, that paves the way for possible easy implementations, based on available systems.

In particular, the prototypical implementation appears viable with minor efforts if done on top of integration systems that exploit a declarative approach to data integration (e.g., [18], where logic programs serve as executable logic specifications for the repair computation). Indeed, our complexity results show that logic engines able to express all problems in the second level of the polynomial hierarchy, such as the DLV system [19], suffices for managing the framework, once we provide appropriate logic specifications.

A number of interesting research questions arise from this work. First, it is natural to ask whether the framework can be extended to the presence of existentially quantified constraints. This can be easily done for some special syntactic fragments, such as for non key-conflicting schemas, i.e., global schemas enriched with inclusion dependencies and keys, for which decidability in the context of data integration systems has been proven in [7]. To this aim, one has to modify the algorithm in [9] to propagate information in a *P2P* system by accounting for mapping assertion as well as for inclusion dependencies, and eventually check that after such propagation no key has been violated.

We conclude by noticing that an avenue of further research is to consider more sophisticated peer-agreement semantics, besides the Pareto-like approach described here. For instance, we may think of some applications where peers may form cooperating groups, or do not cooperate at all. Another line of research may lead to enrich the setting by further kinds of peer preferences criteria, by replacing or complementing the weighting functions proposed in this paper.

Acknowledgments

The work was partially supported by the European Commission under project IST-2001-33570 INFOMIX.

Francesco Scarcello's work was also supported by ICAR-CNR, Rende, Italy.

7. REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison Wesley Publ. Co., Reading, Massachussetts, 1995.

[2] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proc. of PODS'99*, pages 68–79, 1999.

[3] P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Workshop on the Web and Databases, WebDB*, 2002.

[4] Leopoldo Bertossi, Jan Chomicki, Alvaro Cortes, and Claudio Gutierrez. Consistent answers from integrated data sources. In *Proc. of FQAS'02*, pages 71–85, 2002.

[5] Leopoldo E. Bertossi and Loreto Bravo. Query answering in peer-to-peer data exchange systems. In *Proc. of EDBT Workshops 2004*, pages 476–485, 2004.

[6] Loreto Bravo and Leopoldo Bertossi. Logic programming for consistently querying data integration systems. In *Proc. of IJCAI'03*, pages 10–15, 2003.

[7] Andrea Calì, Domenico Lembo, and Riccardo Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *Proc. of PODS'03*, pages 260–271, 2003.

[8] Andrea Calì, Domenico Lembo, and Riccardo Rosati. Query rewriting and answering under constraints in data integration systems. In *Proc. of IJCAI'03*, pages 16–21, 2003.

[9] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Logical foundations of peer-to-peer data integration. In *Proc. of PODS'04*, pages 241–251, 2004.

[10] Thomas Eiter, Michael Fink, Gianluigi Greco, and Domenico Lembo. Efficient evaluation of logic programs for querying data integration systems. In *Proc. of ICLP'03*, pages 348–364, 2003.

[11] Enrico Franconi, Gabriel Kuper, Andrei Lopatenko, and Luciano Serafini. A robust logical and computational characterisation of peer-to-peer database systems. In *Proc. of DBISP2P'03*, pages 64–76, 2003.

[12] Enrico Franconi, Gabriel Kuper, Andrei Lopatenko, and Ilya Zaihrayeu. A distributed algorithm for robust data sharing and updates in p2p database networks. In *Proc. of P2P&DB'04*, pages 446–455, 2004.

[13] Enrico Franconi, Gabriel Kuper, Andrei Lopatenko, and Ilya Zaihrayeu. Queries and updates in the codb peer to peer database system. In *Proc. of VLDB'04*, pages 1277–1280, 2004.

[14] Gianluigi Greco, Sergio Greco, and Ester Zumpano. A logic programming approach to the integration, repairing and querying of inconsistent databases. In *Proc. of ICLP'01*, pages 348–364. Springer, 2001.

[15] Gianluigi Greco and Domenico Lembo. Data integration with prefernces among sources. In *Proc. of ER'04*, pages 231–244, 2004.

[16] Alon Y. Halevy, Zachary G. Ives, Peter Mork, and Igor Tatarinov. Piazza: data management infrastructure for semantic web applications. In *Proc. of WWW'03*, pages 556–567, 2003.

[17] Maurizio Lenzerini. Quality-aware peer-to-peer data integration. In *Proc. of IQIS'04*, 2004.

[18] Nicola Leone, Thomas Eiter, Wolfgang Faber, Michael Fink, Georg Gottlob, Gianluigi Greco, Giovambattista Ianni, Edyta Kalka, Domenico Lembo, Maurizio Lenzerini, Vincenzino Lio, Bartosz Nowicki, Riccardo Rosati, Marco Ruzzi, Witold Staniszkis, and Giorgio Terracina. The INFOMIX system for advanced integration of incomplete and inconsistent data. In *Proc. of SIGMOD'05*, pages 915–917, 2005.

[19] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The DLV System for Knowledge Representation and Reasoning. *ACM Transaction on Cumputational Logic.* To appear.

[20] Luciano Serafini, Fausto Giunchiglia, John Mylopoulos, and Philip A. Bernstein. Local relational model: A logical formalization of database coordination. In *Fourth International and Interdisciplinary Conference on Modeling and Using Context, CONTEXT 2003*, pages 286–299, 2003.

[21] Igor Tatarinov and Alon Halevy. Efficient query reformulation in peer data management systems. In *Proc. of SIGMOD'04*, pages 539–550, 2004.