

THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

Determining the Currency of Data

Citation for published version:
Fan, W, Geerts, F & Wijsen, J 2012, 'Determining the Currency of Data', *ACM Transactions on Database Systems*, vol. 37, no. 4, 25. https://doi.org/10.1145/2389241.2389244

Digital Object Identifier (DOI):
10.1145/2389241.2389244

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Early version, also known as pre-print

Published In:
ACM Transactions on Database Systems

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.



Download date: 29. Mar. 2024

Determining the Currency of Data

WENFEI FAN, University of Edinburgh and Harbin Institute of Technology
FLORIS GEERTS, University of Antwerp
JEF WIJSEN, Université de Mons

Data in real-life databases become obsolete rapidly. One often finds that multiple values of the same entity reside in a database. While all of these values were once correct, most of them may have become stale and inaccurate. Worse still, the values often do not carry reliable timestamps. With this comes the need for studying data currency, to identify the current value of an entity in a database and to answer queries with the current values, in the absence of reliable timestamps.

This paper investigates the currency of data. (1) We propose a model that specifies partial currency orders in terms of simple constraints. The model also allows us to express what values are copied from other data sources, bearing currency orders in those sources, in terms of copy functions defined on correlated attributes. (2) We study fundamental problems for data currency, to determine whether a specification is consistent, whether a value is more current than another, and whether a query answer is certain no matter how partial currency orders are completed. (3) Moreover, we identify several problems associated with copy functions, to decide whether a copy function imports sufficient current data to answer a query, whether a copy function can be extended to import necessary current data for a query while respecting the constraints, and whether it suffices to copy data of a bounded size. (4) We establish upper and lower bounds of these problems, all matching, for combined complexity and data complexity, and for a variety of query languages. We also identify special cases that warrant lower complexity.

Categories and Subject Descriptors: H.2.3 [**Information Systems**]: Database Management—*Languages*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Computational Logic*

General Terms: Languages, Theory, Design

Additional Key Words and Phrases: Currency, Data quality

1. INTRODUCTION

The quality of data in a real-life database quickly degenerates over time. Indeed, it is estimated that "2% of records in a customer file become obsolete in one month" [Eckerson 2002]. That is, in a database of 500 000 customer records, 10 000 records may go stale per month, 120 000 records per year, and within two years about 50% of all the

	FN	LN	address	salary	status
s_1:	Mary	Smith	2 Small St	50k	single
s_2:	Mary	Dupont	10 Elm Ave	50k	married
s_3:	Mary	Dupont	6 Main St	80k	married
s_4:	Bob	Luth	8 Cowan St	80k	married
s_5:	Robert	Luth	8 Drum St	55k	married

(a) Relation Emp

	dname	mgrFN	mrgLN	mgrAddr	budget
t_1:	R&D	Mary	Smith	2 Small St	6500k
t_2:	R&D	Mary	Smith	2 Small St	7000k
t_3:	R&D	Mary	Dupont	6 Main St	6000k
t_4:	R&D	Ed	Luth	8 Cowan St	6000k

(b) Relation Dept

Fig. 1. A company database.

records may be obsolete. In light of this, we often find that multiple values of the same entity reside in a database, which were *once correct*, *i.e.,* they were true values of the entity at some time. However, most of them have become *obsolete* and *inaccurate*. As an example from daily life, when one moves to a new address, a bank may retain her old address, and worse still, credit card bills may still be sent to this old address for quite some time (see, *e.g.,* [Knowledge Integrity 2003] for more examples). Stale data is one of the central problems to data quality. It is known that dirty data costs US businesses 600 billion USD each year [Eckerson 2002], and stale data accounts for a large part of the losses.

This highlights the need for studying the *currency of data*, which aims to identify the current values of entities in a database, and to answer queries using the most current values only.

The question of data currency would be trivial if all data values carried valid timestamps. In practice, however, one often finds that timestamps are unavailable or imprecise [Zhang et al. 2010]. Add to this the complication that data values are often copied or imported from other sources [Berti-Equille et al. 2009; Dong et al. 2010; Dong et al. 2009], which may not support a uniform scheme of timestamps. These make it challenging to identify the current values.

Not all is lost. It is often possible to deduce currency orders from the semantics of the data. Moreover, data copied from other sources inherit currency orders from those sources. Taken together, these may provide sufficient current values of the data to answer certain queries, as illustrated below.

Example 1.1. Consider two relations of a company shown in Fig. 1. Each Emp tuple is an employee record with name, address, salary and marital status. A Dept tuple specifies the name, manager and budget of a department. Records in these relations may be stale, and do not carry timestamps. By entity identification techniques (see, *e.g.,* [Elmagarmid et al. 2007]), we may know that tuples s_1, s_2 and s_3 refer to the same employee Mary, but s_4 and s_5 represent a person distinct from Mary. Consider the following queries posed on these relations.

(1) Query Q_1 is to find Mary's current salary. No timestamps are available for us to tell which of 50k or 80k is more current. However, we may know that the salary of each employee in the company does *not* decrease, as commonly found in the real world. This yields currency orders $s_1 \prec_{\mathsf{salary}} s_3$ and $s_2 \prec_{\mathsf{salary}} s_3$, *i.e.,* $s_3[\mathsf{salary}]$ is more current than both $s_1[\mathsf{salary}]$ and $s_2[\mathsf{salary}]$. Hence the answer to Q_1 is 80k.

(2) Query Q_2 is to find Mary's current last name. We can no longer answer Q_2 as above. Nonetheless, we may know the following: (a) the marital status can only change from single to married and from married to divorced; but not from married to single; and (b) Emp tuples with the most current marital status also contain the most current last name. Therefore, $s_1 \prec_{\mathsf{LN}} s_2$ and $s_1 \prec_{\mathsf{LN}} s_3$, and the answer to Q_2 is Dupont.

(3) Query Q_3 is to find Mary's current address. We may know that Emp tuples with the most current status or salary contain the most current address. Putting this and (1) above together, we know that the answer to Q_3 is "6 Main St".

(4) Finally, query Q_4 is to find the current budget of department R&D. Again no timestamps are available for us to evaluate the query. However, we may know the following: (a) Dept tuples t_1 and t_2 have copied their mgrAddr values from $s_1[\mathsf{address}]$ in Emp; similarly, t_3 has copied from s_3, and t_4 from s_4; and (b) in Dept, tuples with the most current address also have the most current budget. Taken together, these tell us that $t_1 \prec_{\mathsf{budget}} t_3$ and $t_2 \prec_{\mathsf{budget}} t_3$. Observe that we do not know which budget in t_3 or t_4 is more current. Nevertheless, in either case the most current budget is 6000k, and hence it is the answer to Q_4. $\diamond$

These suggest that we give a full treatment of data currency, and answer the following questions. How should we specify currency orders on data values in the absence of timestamps but in the presence of copy relationships? When currency orders are only partly available, can we decide whether an attribute value is more up-to-date than another? How can we answer a query with only current data in a database? To answer a query, do we need to import current data from another source, and if so, what to copy? The ability to answer these questions may provide guidance for practitioners to decide, *e.g.,* whether the answer to a query is corrupted by stale data, or what copy functions are needed, among other things.

A model for data currency. To answer these questions, we approach data currency based on the following.

(1) For each *attribute* A of a relation D, we assume an (implicit) currency order $\prec_A$ on its tuples such that for tuples t_1 and t_2 in D that represent the same real-world entity, $t_1 \prec_A t_2$ indicates that t_2 is more up-to-date than t_1 in the A attribute value. Here $\prec_A$ is not a total order since in practice, currency information is only partially available. Note that for distinct attributes A and B, we may have $t_1 \prec_A t_2$ and $t_2 \prec_B t_1$, *i.e.,* there may be no single tuple that is most up-to-date in all attribute values.

(2) We express additional currency relationships as denial constraints [Bertossi 2006; Chomicki 2007], which are simple universally quantified FO sentences that have been used to improve the consistency of data. We show that the same class of constraints also suffices to express currency semantics commonly found in practice. For instance, all the currency relations we have seen in Example 1.1 can be expressed as denial constraints.

(3) We define a copy relationship from relation D_j to D_k in terms of a partial mapping, referred to as a *copy function*. It specifies what attribute values in D_j have been copied from D_k along with their currency orders in D_k. It also assures that correlated attributes are copied together. As observed in [Berti-Equille et al. 2009; Dong et al. 2010; Dong et al. 2009], copy functions are common in the real world, and can be automatically discovered.

Putting these together, we consider $\mathbf{D} = (D_1, \ldots, D_n)$, a collection of relations such that (a) each D_j has currency orders *partially defined* on its tuples for each attribute, indicating *available* currency information; (b) each D_j satisfies a set Σ_j of denial constraints, which expresses currency orders derived from the semantics of the data; and

(c) for each pair (D_j, D_k) of relations, there are possibly copy functions defined on them, which import values from one to another.

We study *consistent completions* D_j^c of D_j, which extend $\prec_A$ in D_j to a total order on all tuples *pertaining to the same entity*, such that D_j^c satisfies Σ_j and those constraints imposed by the copy functions. One can construct from D_j^c the *current tuple* for each entity *w.r.t.* $\prec_A$, which contains the entity's most current A value for each attribute A. This yields the *current instance* of D_j^c consisting of only the current tuples of the entities in D_j, from which currency orders are removed. We evaluate a query Q on current instances of relations in $\mathbf{D}$, without worrying about currency orders. We study *certain current answers* to Q in $\mathbf{D}$, *i.e.,* tuples that are the answers to Q in all consistent completions of $\mathbf{D}$.

Reasoning about data currency. We study fundamental problems for data currency. (a) The *consistency problem* is to determine, given denial constraints Σ_j imposed on each D_j and copy functions between these relations, whether there exist consistent completions of every D_j, *i.e.,* whether the specification makes sense. (b) The *certain ordering problem* is to decide whether a currency order is contained in all consistent completions. (c) The *deterministic current instance problem* is to determine whether the current instance of each relation remains unchanged for all consistent completions. The ability to answer these questions allows us to determine whether an attribute value is certainly more current than another, and to identify the current value of an entity. (d) The *certain current query answering problem* is to decide whether a tuple t is a certain current answer to a query Q, *i.e.,* it is certainly computed using current data.

Currency preserving copy functions. It is natural to ask what values should be copied from one data source to another in order to answer a query. To characterize this intuition we introduce a notion of currency preservation. Consider data sources $\mathbf{D} = (D_1, \ldots, D_p)$ and $\mathbf{D}' = (D'_1, \ldots, D'_q)$, each consisting of a collection of relations with denial constraints imposed on them. Consider copy functions $\overline{\rho}$ from relations in $\mathbf{D}'$ to those in $\mathbf{D}$. For a query Q posed on $\mathbf{D}$, we say that $\overline{\rho}$ is *currency preserving* if no matter how we extend $\overline{\rho}$ by copying from $\mathbf{D}'$ more values of those entities in $\mathbf{D}$, the certain current answers to Q in $\mathbf{D}$ remain unchanged. In other words, $\overline{\rho}$ has already imported all data values needed for computing certain current answers to Q.

We identify several problems associated with currency-preserving copy functions. (a) The *currency preservation problem* is to determine, given Q, $\overline{\rho}$, $\mathbf{D}$, $\mathbf{D}'$ and their denial constraints, whether $\overline{\rho}$ is currency preserving for Q. Intuitively, we want to know whether we need to extend $\overline{\rho}$ in order to answer Q. (b) The *existence problem* is to determine whether $\overline{\rho}$ can be extended to be currency preserving for Q. (c) Moreover, the *bounded copying* problem is to decide whether there exists such an extension that imports additional data of a bounded size. Intuitively, we want to find currency-preserving copy functions that import as few data values as possible.

Complexity results. We provide *combined complexity* and *data complexity* of all the problems stated above. For the combined complexity of the problems that involve queries, we investigate the impact of various query languages, including conjunctive queries (CQ), unions of conjunctive queries (UCQ), positive existential first-order logic ($\exists$FO$^+$) and first-order logic (FO). We establish upper and lower bounds of all these problems, *all matching*, ranging over $O(1)$, NP, coNP, Π_2^p, Σ_2^p, Π_3^p, Σ_3^p, Σ_4^p and PSPACE. We find that most of the problems are intractable. In light of this, we also identify special practical cases with lower complexity, some in PTIME. We also study the impact of denial constraints. For example, in the absence of denial constraints,

the certain current query answering problem is in PTIME for SP queries (CQ queries without "join"), but it becomes intractable when denial constraints are present, even when the constraints are fixed.

This work is a first step towards a systematic study of data currency in the absence of reliable timestamps but in the presence of copy relationships. The results may help practitioners decide how to specify data currency, analyze query answers and design copy functions. We also provide a complete picture of complexity bounds for important problems associated with data currency and copy functions, which are proved by using a variety of reductions and by providing (PTIME) algorithms.

Related work. There has been a host of work on temporal databases (see, *e.g.,* [Chomicki and Toman 2005; Snodgrass 1999] for surveys). Temporal databases provide support for valid time, transaction time, or both. They assume the availability of timestamps, and refer to "now" by means of current-time variables [Clifford et al. 1997; Dyreson et al. 2009]. Dynamic and temporal integrity constraints allow us to restrict the set of legal database evolutions. Our currency model differs from temporal data models in several respects. We do not assume explicit timestamps. Nevertheless, if such timestamps are present, they can be related to currency by means of denial constraints or partial currency orders. Unlike temporal databases that timestamp entire tuples, our model allows that different values within the same tuple have distinct currencies. That is, the same tuple can contain an up-to-date value for one attribute, and an outdated value for another attribute.

Since currency orders are different from temporal orders used in temporal databases, our currency (denial) constraints differ from traditional temporal constraints. Currency constraints can sometimes be derived from temporal constraints, however. For example, when salaries are constrained to be non-decreasing, we can express that the highest salary is the most current one. Also, our copy functions can require certain attributes to be copied together when these attributes cannot change independently, as for example expressed by the dynamic functional dependencies in [Vianu 1987].

Closer to this work are [van der Meyden 1997; Koubarakis 1994; 1997; Grohe and Schwandtner 2009] on querying indefinite data. In [van der Meyden 1997], the evaluation of CQ queries is studied on data that is linearly ordered but only provides a partial order. The problem studied there is similar to (yet different from) certain current query answering. An extension of conditional tables [Grahne 1991; Imieliński and Lipski 1984] is proposed in [Koubarakis 1994] to incorporate indefinite temporal information, and in that setting, the complexity bounds for FO query evaluation are provided in [Koubarakis 1997]. Recently the non-emptiness problem for datalog on linear orders is investigated in [Grohe and Schwandtner 2009]. However, none of these considers copying data from external sources, or the analyses of certain ordering and currency-preserving copy functions. In addition, we answer queries using current instances of relations, which are normal relations without (currency) ordering. This semantics is quite different from its counterparts in previous work. We also consider denial constraints and copy functions, which are not expressible in CQ or datalog studied in [van der Meyden 1997; Grohe and Schwandtner 2009]. In contrast to our work, [Koubarakis 1994; 1997] assume explicit timestamps, while we use denial constraints to specify data currency. To encode denial constraints in extended conditional tables of [Koubarakis 1994; 1997], an exponential blowup is inevitable. Because of these reasons, the results of [van der Meyden 1997; Koubarakis 1994; 1997; Grohe and Schwandtner 2009] cannot carry over to our setting, and vice versa.

There has also been a large body of work on the temporal constraint satisfaction problem (TCSP), which is to find a valuation of temporal variables that satisfies a set

of temporal constraints (see, *e.g.,* [Bodirsky and Kára 2010; Schwalb and Vila 1998]). It differs from our consistency problem in that it considers neither completions of currency orders that satisfy denial constraints, nor copy relationships. Hence the results for TCSP are not directly applicable to our consistency problem, and vice versa.

Copy relationships between data sources have recently been studied in [Berti-Equille et al. 2009; Dong et al. 2010; Dong et al. 2009]. The previous work has focused on automatic discovery of copying dependencies and functions. Copy relationships are also related to data provenance, which studies propagation of annotations in data transformations and updates (see [Buneman et al. 2008; Cheney et al. 2009] for recent surveys on data provenance). However, to the best of our knowledge, no previous work has studied currency-preserving copy functions and their associated problems.

Denial constraints have proved useful in data repairing (see, *e.g.,* [Bertossi 2006; Chomicki 2007]). We adopt the same class of constraints to specify the currency of data, so that data currency and consistency could be treated in a uniform logical framework. Denial constraints can also be automatically discovered, along the same lines as data dependency profiling (see, *e.g.,* [Fan et al. 2011]).

The study of data currency is also related to research on incomplete information (see [van der Meyden 1998] for a survey), when missing data concerns data currency. In contrast to that line of work, we investigate how to decide whether a value is more current than another, and study the properties of copy functions. We use denial constraints to specify data currency, which are, as remarked earlier, more succinct than, *e.g.,* C-tables and V-tables for representing incomplete information [Grahne 1991; Imieliński and Lipski 1984]. In addition, we evaluate queries using current instances, a departure from the study of incomplete information.

Certain query answers have been studied in data integration and exchange. In data integration, for a query Q posed on a global database D_G, it is to find the certain answers to Q over all data sources that are consistent with D_G *w.r.t.* view definitions (see *e.g.,* [Lenzerini 2002]). In data exchange, it is to find the certain answers to a query over all target databases generated from data sources via schema mapping (see [Kolaitis 2005]). By contrast, we consider certain answers to a query over all completions of currency orders, which satisfy denial constraints and constraints from copy functions. Certain current query answering is also different from consistent query answering (see, *e.g.,* [Bertossi 2006; Chomicki 2007]), which is to find certain answers to a query over all *repairs* of a database and does not distinguish between stale and current data in the repairs. Finally, whereas it may be possible to model our setting as a data exchange scenario with built-in constraints [Deutsch et al. 2008], our complexity results do not follow gratuitously and a careful analysis of the chase is required in this setting.

This paper is an extension of earlier work [Fan et al. 2011] by including (a) proofs for all the theorems; some of the proofs are nontrivial and the techniques are interesting in their own right; (b) new proofs for stronger lower bounds of the certain ordering problem and the deterministic current instance problem, in a practical setting when input specifications are assumed consistent (Theorem 3.4). The previous proofs in [Fan et al. 2011] heavily relied on the availability of inconsistent input specifications.

Organization. Section 2 presents the data currency model. Section 3 states its related problems and establishes their complexity bounds. Section 4 introduces the notion of currency preservation and its fundamental problems, followed by their complexity analysis in Section 5. Some tractable cases of the problems in connection with data currency and currency preservation are identified in Section 6. Section 7 summarizes the main results of the paper.

2. DATA CURRENCY

We introduce a model for specifying data currency. A specification consists of (a) partial currency orders, (b) denial constraints, and (c) copy functions. We first present these notions, and then study consistent completions of currency orders. Finally, we show how queries are answered on current instances derived from these completions.

Data with partial currency orders. A relation schema is specified as $R = (\mathsf{EID}, A_1, \ldots, A_n)$, where EID denotes entity id that identifies tuples pertaining to the same entity, as introduced by Codd [Codd 1979]. Such EID values can be obtained using entity identification techniques (*a.k.a.* record linkage, record matching and data deduplication; see, *e.g.,* [Elmagarmid et al. 2007] for a survey). A finite instance D of R is referred to as a *normal instance of* R.

A *temporal instance* D_t *of* R is given as $(D, \prec_{A_1}, \ldots, \prec_{A_n})$, where each $\prec_{A_i}$ is a strict partial order defined on D such that for tuples t_1 and t_2 in D, $t_1 \prec_{A_i} t_2$ implies $t_1[\mathsf{EID}] = t_2[\mathsf{EID}]$. We call $\prec_{A_i}$ the *currency order for attribute* A_i. Recall that a strict partial order is irreflexive and transitive, and therefore asymmetric. Intuitively, if $t_1 \prec_{A_i} t_2$, then t_1 and t_2 refer to *the same entity*, and t_2 contains a *more current* A_i-value for that entity than t_1. In other words, t_2 is more current than t_1 *in attribute* A_i. A currency order $\prec_{A_i}$ is empty when no currency information is known for A_i.

A *completion* of $D_t = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ is a temporal instance $D_t^c = (D, \prec^c_{A_1}, \ldots, \prec^c_{A_n})$ of R, such that for each $i \in [1, n]$, (1) $\prec_{A_i} \subseteq \prec^c_{A_i}$, and (2) for all $t_1, t_2 \in D$, t_1 and t_2 are comparable under $\prec^c_{A_i}$ iff $t_1[\mathsf{EID}] = t_2[\mathsf{EID}]$. The latter condition implies that $\prec^c_{A_i}$ induces a total order on tuples that refer to the same entity, while tuples representing distinct entities are not comparable under $\prec^c_{A_i}$. We call $\prec^c_{A_i}$ a *completed currency order*. Note that tuples that bear the same EID and carry the same value v in an attribute A_i are also comparable via $\prec^c_{A_i}$ in a completion. When considering the most current value of A_i (see below), however, it is irrelevant which of these tuples contributes if v is taken as the latest value of A_i, since these tuples share the same A_i value.

Denial constraints. We use denial constraints [Bertossi 2006; Chomicki 2007] to specify additional currency information derived from the semantics of data. A *denial constraint* φ for R is a universally quantified FO sentence of the form:

$$\forall t_1, \ldots, t_k : R \big(\bigwedge_{j \in [1,k]} (t_1[\mathsf{EID}] = t_j[\mathsf{EID}] \wedge \psi) \rightarrow t_u \prec_{A_i} t_v \big),$$

where $u, v \in [1, k]$, each t_j is a tuple variable denoting a tuple of R, and ψ is a conjunction of predicates of the form (1) $t_j \prec_{A_l} t_h$, *i.e.,* t_h is more current than t_j in attribute A_l; (2) $t_j[A_l] = t_h[A_l]$ (resp. $t_j[A_l] \neq t_h[A_l]$), *i.e.,* $t_j[A_l]$ and $t_h[A_l]$ are identical (resp. distinct) values; (3) $t_j[A_l] = c$ (resp. $t_j[A_l] \neq c$), where c is a constant; and (4) possibly other built-in predicates defined on particular domains. These constraints enrich $\prec_{A_i}$.

The constraint is interpreted over completions D_t^c of temporal instances of R. We say that D_t^c *satisfies* φ, denoted by $D_t^c \models \varphi$, if for all tuples $t_1, \ldots, t_k$ in D that have the same EID value, if these tuples satisfy the predicates in ψ following the standard semantics of FO, then $t_u \prec^c_{A_i} t_v$. The use of EID in φ enforces that φ is imposed on tuples that refer to *the same entity*. We say that D_t^c satisfies a set Σ of denial constraints, denoted by $D_t^c \models \Sigma$, if $D_t^c \models \varphi$ for all $\varphi \in \Sigma$.

Example 2.1. Recall relations Emp and Dept given in Fig. 1. Denial constraints on these relations include:

φ_1: $\forall s, t : \mathsf{Emp}\big((s[\mathsf{EID}] = t[\mathsf{EID}] \wedge s[\mathsf{salary}] > t[\mathsf{salary}]) \rightarrow t \prec_{\mathsf{salary}} s\big)$
φ_2: $\forall s, t : \mathsf{Emp}\big((s[\mathsf{EID}] = t[\mathsf{EID}] \wedge s[\mathsf{status}] = \text{“married”} \wedge t[\mathsf{status}] = \text{“single”}) \rightarrow t \prec_{\mathsf{LN}} s\big)$
φ_3: $\forall s, t : \mathsf{Emp}\big((s[\mathsf{EID}] = t[\mathsf{EID}] \wedge t \prec_{\mathsf{salary}} s) \rightarrow t \prec_{\mathsf{address}} s\big)$

φ_4: $\forall s, t : \mathsf{Dept}\big((s[\mathsf{EID}] = t[\mathsf{EID}] \ \wedge \ t \prec_{\mathsf{mgrAddr}} s) \ \rightarrow \ t \prec_{\mathsf{budget}} s\big)$

Here φ_1 states that when Emp tuples s and t refer to the same employee, if s[salary] $>$ t[salary], then s is more current than t in attribute salary. Note that '$>$' denotes the built-in predicate "greater-than" in the numeric domain of salary, whereas $\prec_{\mathsf{salary}}$ is the currency order for salary. Constraint φ_2 asserts that if s[status] is married and t[status] is single, then s is more current than t in LN. Constraint φ_3 states that if s is more current than t in salary, then s is also more current than t in address; similarly for φ_4. $\diamond$

Copy functions. Consider two temporal instances $D_{(t,1)} = (D_1, \prec_{A_1}, \ldots, \prec_{A_p})$ and $D_{(t,2)} = (D_2, \prec_{B_1}, \ldots, \prec_{B_q})$ of (possibly distinct) relation schemas R_1 and R_2, respectively. A *copy function* ρ of signature $R_1[\vec{A}] \Leftarrow R_2[\vec{B}]$ is a partial mapping from D_1 to D_2, where $\vec{A} = (A_1, \ldots, A_l)$ and $\vec{B}$ = $(B_1, \ldots, B_l)$ denote attributes in R_1 and R_2, respectively. Here ρ is required to satisfy *the copying condition*: for each tuple t in D_1, if $\rho(t) = s$, then $t[A_i] = s[B_i]$ for all $i \in [1, l]$.

Intuitively, for tuples $t \in D_1$ and $s \in D_2$, $\rho(t) = s$ indicates that the values of the $\vec{A}$ attributes of t have been imported from the $\vec{B}$ attributes of tuple s in D_2. Here $\vec{A}$ specifies a list of *correlated attributes* that should be copied together.

The copy function ρ is called $\prec$*-compatible* relative to the currency orders found in $D_{(t,1)}$ and $D_{(t,2)}$ if for all $t_1, t_2 \in D_1$, for each $i \in [1, l]$, if $\rho(t_1) = s_1$, $\rho(t_2) = s_2$, t_1[EID] $= t_2$[EID] and s_1[EID] $= s_2$[EID], then $s_1 \prec_{B_i} s_2$ implies $t_1 \prec_{A_i} t_2$. Intuitively, $\prec$*-compatibility* requires that copy functions preserve currency orders. In other words, when attribute values are imported from D_2 to D_1 the currency orders on corresponding tuples defined in $D_{(t,2)}$ are inherited by $D_{(t,1)}$.

Example 2.2. Consider relations Emp and Dept shown in Fig. 1. A copy function ρ of signature Dept[mgrAddr] $\Leftarrow$ Emp[address], depicted in Fig. 1 by arrows, is given as follows: $\rho(t_1) = s_1$, $\rho(t_2) = s_1$, $\rho(t_3) = s_3$ and $\rho(t_4) = s_4$. That is, the mgrAddr values of t_1 and t_2 have both been imported from s_1[address], while t_3[mgrAddr] and t_4[mgrAddr] are copied from s_3[address] and s_4[address], respectively. The function satisfies the copying condition, since t_1[mgrAddr] $= t_2$[mgrAddr] $= s_1$[address], t_3[mgrAddr] $= s_3$[address], and t_4[mgrAddr] $= s_4$[address].

Suppose that $\prec_A$ is empty for each attribute A in Emp or Dept. Then the copy function ρ is $\prec$-compatible *w.r.t.* these temporal instances of Emp and Dept. By contrast, assume that partial currency orders $s_1 \prec_{\mathsf{address}} s_3$ on Emp and $t_3 \prec_{\mathsf{mgrAddr}} t_1$ are given. Then ρ is not $\prec$-compatible. Indeed, since s_1, s_3 pertain to the same person Mary, and t_1, t_3 to the same department R&D, the relation $s_1 \prec_{\mathsf{address}} s_3$ should carry over into $t_1 \prec_{\mathsf{mgrAddr}} t_3$, as $\rho(t_1) = s_1$ and $\rho(t_3) = s_3$. Clearly, $t_3 \prec_{\mathsf{mgrAddr}} t_1$ and $t_1 \prec_{\mathsf{mgrAddr}} t_3$ are contradictory. $\diamond$

Consistent completions of temporal orders. A *specification* $\mathbf{S}$ of data currency consists of (1) a collection of temporal instances $D_{(t,i)}$ of schema R_i for $i \in [1, s]$, (2) a set Σ_i of denial constraints imposed on each $D_{(t,i)}$, and (3) a (possibly empty) copy function $\rho_{(i,j)}$ that imports data from $D_{(t,i)}$ to $D_{(t,j)}$ for $i, j \in [1, s]$. It specifies data values and entities (by normal instances embedded in $D_{(t,i)}$), partial currency orders known for each relation (by $D_{(t,i)}$), additional currency information derived from the semantics of the data (Σ_i), and data that has been copied from one source to another ($\rho_{(i,j)}$). These $D_{(t,i)}$'s may denote different data sources, *i.e.,* they may not necessarily be in the same database.

A *consistent completion* $\mathbf{D}^c$ of $\mathbf{S}$ consists of temporal instances $D^c_{(t,i)}$ of R_i such that for all $i, j \in [1, s]$,

(1) $D^c_{(t,i)}$ is a completion of $D_{(t,i)}$,
(2) $D^c_{(t,i)} \models \Sigma_i$, and
(3) $\rho_{(i,j)}$ is $\prec$-compatible *w.r.t.* the completed currency orders found in $D^c_{(t,i)}$ and $D^c_{(t,j)}$.

We use $\mathsf{Mod}(\mathbf{S})$ to denote the set of all consistent completions of $\mathbf{S}$. We say that $\mathbf{S}$ is *consistent* if $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$, *i.e.,* there exists at least one consistent completion of $\mathbf{S}$.

Intuitively, if $D_{(t,i)} = (D_i, \prec_{A_1}, \ldots, \prec_{A_n})$ is part of a specification and $D^c_{(t,i)} = (D_i, \prec^c_{A_1}, \ldots, \prec^c_{A_n})$ is part of a consistent completion of that specification, then each $\prec^c_{A_j}$ extends $\prec_{A_j}$ to a completed currency order, and the completed orders satisfy the denial constraints Σ_i and the constraints imposed by copy functions. Observe that the copying condition and $\prec$-compatibility impose constraints on consistent completions. This is particularly evident when a data source imports data from multiple sources, and when two data sources copy from each other, directly or indirectly. In addition, these constraints interact with denial constraints.

Example 2.3. Consider a specification $\mathbf{S}_0$ consisting of Emp and Dept of Fig. 1, the denial constraints φ_1–φ_4 given in Example 2.1, and the copy function ρ defined in Example 2.2. Assume that no currency orders are known for Emp and Dept initially. A consistent completion $\mathbf{D}^c_0$ of $\mathbf{S}_0$ defines (1) $s_1 \prec_A s_2 \prec_A s_3$ when A ranges over FN, LN, address, salary and status for Emp tuples, and (2) $t_1 \prec_B t_2 \prec_B t_4 \prec_B t_3$ when B ranges over mgrFN, mgrLN, mgrAddr and budget for Dept tuples (assuming that dname is the EID attribute of Dept). One can verify that $\mathbf{D}^c_0$ satisfies the denial constraints and the constraints imposed by ρ, and hence, $\mathbf{D}^c_0 \in \mathsf{Mod}(\mathbf{S}_0)$. No currency order is defined between any of s_1, s_2, s_3 and any of s_4, s_5, since they represent different entities.

As another example, suppose that there is a copy function ρ_1 that imports budget attribute values of t_1 and t_3 from the budget attributes of s'_1 and s'_3 in another source D_1, respectively, where $s'_1 = t_1$ and $s'_3 = t_3$, but in D_1, $s'_3 \prec_{\mathsf{budget}} s'_1$. Then there is no consistent completion in this setting either. Indeed, all completed currency orders of $\prec_{\mathsf{budget}}$ in Dept have to satisfy denial constraints φ_1, φ_3 and φ_4, which enforce $t_1 \prec_{\mathsf{budget}} t_3$, but ρ_1 is not $\prec$-compatible with this currency order. This shows the interaction between denial constraints and currency constraints of copy functions. $\diamond$

Current instances. In a temporal instance $D_t = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ of R, let $E = \{t[\mathsf{EID}] \mid t \in D\}$, and for each entity $e \in E$, let $I_e = \{t \in D \mid t[\mathsf{EID}] = e\}$. That is, E contains all EID values in D, and I_e is the set of tuples pertaining to the entity with $\mathsf{EID} = e$. In a completion D^c_t of D_t, for each attribute A of R, the *current A value* for entity $e \in E$ is $t[A]$, where t is the greatest (*i.e.,* most current) tuple in the totally ordered set $(I_e, \prec^c_A)$. The *current tuple* for entity $e \in E$, denoted by $\mathsf{LST}(e, D^c_t)$, is the tuple t_e such that for each attribute A of R, $t_e[A]$ is the current A value for entity e. Here LST stands for "last" since current tuples are formed by collecting the last values in totally ordered sets in a completion. Note that $\mathsf{LST}(e, D^c_t)$ is most current *relative to* the information available in E for entity e; we defer the discussion of incomplete information to Section 7, when tuples or values may be missing from E.

We use $\mathsf{LST}(D^c_t)$ to denote $\big\{\mathsf{LST}(e, D^c_t) \mid e \in E\big\}$, referred to as the *current instance* of D^c_t. Observe that $\mathsf{LST}(D^c_t)$ is a *normal instance* of R, carrying no currency orders. For any $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$, we define $\mathsf{LST}(\mathbf{D}^c) = \big\{\mathsf{LST}(D^c_{(t,i)}) \mid D^c_{(t,i)} \in \mathbf{D}^c\big\}$, the set of all current instances.

Example 2.4. Recall the completion $\mathbf{D}^c_0$ of $\mathbf{S}_0$ from Example 2.3. We have that $\mathsf{LST}(\mathbf{D}^c_0) = \big\{\mathsf{LST}(\mathsf{Emp}), \mathsf{LST}(\mathsf{Dept})\big\}$, where $\mathsf{LST}(\mathsf{Emp}) = \{s_3, s_4, s_5\}$, and $\mathsf{LST}(\mathsf{Dept}) = \{t_3\}$. Note that $\mathsf{LST}(\mathsf{Emp})$ and $\mathsf{LST}(\mathsf{Dept})$ are normal instances.

Table I. A summary of notations

D	a normal instance of a relation schema R
D_t	a temporal instance of R with partial currency orders
D_t^c	a completion of partial currency orders in D_t
$\mathbf{S}$	a specification of data currency
$\mathbf{D}^c$	a consistent completion of a specification $\mathbf{S}$
$\mathsf{LST}(\mathbf{D}^c)$	the current instance of $\mathbf{D}^c$
$\bar{\rho}$	a collection of copy functions in $\mathbf{S}$
$\bar{\rho}^e$	an extension of copy functions $\bar{\rho}$
$\mathbf{S}^e$	an extension of specification $\mathbf{S}$ by $\bar{\rho}^e$
$\mathbf{D}^e$	an extension of temporal instances by $\bar{\rho}^e$

As another example, suppose that s_4 and s_5 refer to the same person. Consider an extension of the currency orders given in $\mathbf{D}_0^c$ by adding $s_4 \prec_A s_5$ and $s_5 \prec_B s_4$, where A ranges over FN, LN, address and status while B is salary. Then the current tuple of this person is (Robert, Luth, 8 Drum St, 80k, married), in which the first four attributes are taken from s_5 while its salary attribute is taken from s_4. $\diamond$

Certain current answers. Consider a query Q posed on normal instances of $(R_1, \ldots, R_l)$, which does not refer to currency orders, where R_i is in specification $\mathbf{S}$ for $i \in [1, l]$. We say that a tuple t is a *certain current answer* to Q *w.r.t.* $\mathbf{S}$ if t is in

$$\bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} Q\big(\mathsf{LST}(\mathbf{D}^c)\big).$$

That is, t is guaranteed to belong to the answer computed from the current values no matter how the partial currency orders in $\mathbf{S}$ are completed, as long as the denial constraints and constraints imposed by the copy functions of $\mathbf{S}$ are satisfied.

Example 2.5. Recall queries Q_1, Q_2, Q_3 and Q_4 from Example 1.1, and specification $\mathbf{S}_0$ from Example 2.3. One can verify that answers to the queries given in Example 1.1 are certain current answers *w.r.t.* $\mathbf{S}_0$, *i.e.,* the answers remain unchanged in $\mathsf{LST}(\mathbf{D}^c)$ for all $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S}_0)$. $\diamond$

We summarize notations in Table I, including those given in this section and notations to be introduced in Section 4.

3. REASONING ABOUT THE CURRENCY OF DATA

We study four problems associated with data currency, and establish their data complexity and combined complexity. For the data complexity, we fix denial constraints and queries (for CCQA), and study the complexity in terms of varying size of data sources and copy functions. For the combined complexity we also allow denial constraints and queries to vary (see, *e.g.,* [Abiteboul et al. 1995] for data and combined complexity).

The consistency of specifications. The first problem is to decide whether a given specification $\mathbf{S}$ makes sense, *i.e.,* whether there exists a consistent completion of $\mathbf{S}$. As shown in Example 2.3, there exist specifications $\mathbf{S}$ such that $\mathsf{Mod}(\mathbf{S})$ is empty, because of the interaction between denial constraints and copy functions, among other things.

CPS:	The *consistency problem for specifications*.
INPUT:	A specification $\mathbf{S}$ of data currency.
QUESTION:	Is $\mathsf{Mod}(\mathbf{S})$ nonempty?

The result below tells us the following. (1) The problem is nontrivial: it is Σ_2^p-complete. It remains intractable even when denial constraints are fixed (data complexity).

(2) Denial constraints are a major factor that makes the problem hard. Indeed, the complexity bounds are not affected even when no copy functions are defined in $\mathbf{S}$.

THEOREM 3.1. *For* CPS*, (1) the combined complexity is Σ_2^p-complete, and (2) the data complexity is NP-complete. The upper bounds and lower bounds remain unchanged even in the absence of copy functions.* □

PROOF. We show that the combined complexity of deciding whether $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$ for a specification $\mathbf{S}$ is Σ_2^p-complete and its data complexity is NP-complete.

Combined complexity CPS*:* The Σ_2^p-hardness of CPS is shown by reduction from the $\exists^*\forall^*$3DNF problem, which is known to be Σ_2^p-complete [Stockmeyer 1976]. The $\exists^*\forall^*$3DNF problem is to determine, given a sentence $\varphi = \exists X \forall Y \psi(X, Y)$, whether φ is true. Here $X = \{ x_1, \ldots, x_m \}$, $Y = \{ y_1, \ldots, y_n \}$, and ψ is a formula $C_1 \vee \cdots \vee C_r$ such that for each $i \in [1, r]$, clause C_i is of the form $\ell_1^i \wedge \ell_2^i \wedge \ell_3^i$, where for each $j \in [1, 3]$, ℓ_j^i is either a variable or the negation of a variable in $X \cup Y$.

Given an instance φ of $\exists^*\forall^*$3DNF, we define a specification $\mathbf{S}$ consisting of a single fixed schema, a corresponding temporal instance and a single denial constraint. No copy functions are defined. We then show that φ is true iff $\mathsf{Mod}(\mathbf{S})$ is non-empty. More specifically, the specification $\mathbf{S}$ is defined as follows.

(1) Temporal instance. A (fixed) relation schema $R_V(\mathsf{EID}, V, v, A_1, A_2, A_3, B)$ is defined in $\mathbf{S}$. Its temporal instance I_V consists of three parts: (a) $I_X = \{ t_i, t'_i \mid i \in [1, m] \}$, where $t_i = (\mathsf{eid}, x_i, 1, \#, \#, \#, \#)$, $\#$ is a distinct symbol (a placeholder), and t'_i is the same as t_i except $t'_i[v] = 0$; (b) $I_Y = \{ s_j, s'_j \mid j \in [1, n] \}$, where $s_j = (\mathsf{eid}, y_j, 1, \#, \#, \#, \#)$, and similarly for s'_j except $s'_j[v] = 0$; and (c) $I_\vee$ consists of 8 tuples c_l encoding disjunction such that $c_l[\mathsf{EID}] = \mathsf{eid}$, $c_l[A_p]$ ranges over 0 and 1 for $p \in [1, 3]$, $c_l[B] = c_l[A_1] \vee c_l[A_2] \vee c_l[A_3]$, and $c_l[V] = c_l[v] = \#$. The currency order for V is such that for any tuples $t, s \in I_V$, $t \prec_V s$ if (a) $t[V] = x_i$, $s[V] = x_j$ and $i < j$, (b) $t[V] = y_i$, $s[V] = y_j$ and $i < j$, (c) $t[V] = x_i$ and $s[V] = y_j$ when $i \in [1, m]$ and $j \in [1, n]$, or (d) $t[V] = \#$ but $s[v] \neq \#$. The initial partial currency orders for the other attributes are empty. Intuitively, completions I_V^c of I_V are (a) to encode truth assignments μ for X such that for any $t_i, t'_i \in I_X^c$, if $t_i[V] = t'_i[V] = x_i$ and $t'_i[v] \prec_v t_i[v]$, then $\mu(x_i)$ takes the value of $t_i[v]$ (either 0 or 1); (b) to enumerate all truth assignments of Y, *i.e.,* for each $j \in [1, n]$, s_j and s'_j denote the truth values of y_j (both 0 and 1); and (c) to conduct the disjunction computation with the A_p and B attributes of the tuples in $I_\vee$, which, as will be seen shortly, is needed to encode ψ.

(2) Denial constraints. We define a denial constraint ϕ to encode $\varphi = \exists X \forall Y \psi(X, Y)$ (the equality of EID is omitted as all tuples of I_V^c refer to the same entity):

$$\phi = \forall t_1, t'_1, \ldots, t_m, t'_m,\ \forall s_1, \ldots, s_n,\ \forall c_1, \ldots, c_r \big((\bigwedge_{i \in [1,m]} \xi_i \wedge \bigwedge_{j \in [1,n]} \chi_j \wedge \bigwedge_{l \in [1,r]} \omega_l) \rightarrow t_1 \prec_V t_1 \big)$$

Here for each $i \in [1, m]$, ξ_i is $t_i[V] = t'_i[V] = x_i \wedge t'_i \prec_v t_i$, *i.e.,* $t_i[v]$ indicates the truth value of x_i in I_X^c. For each $j \in [1, n]$, χ_j is $s_j[V] = y_j$, indicating a truth value of y_i. For each $l \in [1, r]$, ω_l encodes the negation $\neg C_l$ of the conjunctive clause $C_l = \ell_1^l \wedge \ell_2^l \wedge \ell_3^l$. More specifically, ω_l is of the form $(c_l[B] = 1) \wedge \bigwedge_{p \in [1,3]} \eta_p^l$, where η_p^l is one of the following four cases: (a) $c_l[A_p] \neq t_i[v]$ if ℓ_p^l is x_i, (b) $c_l[A_p] = t_i[v]$ if ℓ_p^l is $\bar{x}_i$, (c) $c_l[A_p] \neq s_j[v]$ if ℓ_p^l is y_j, and (d) $c_l[A_p] = s_j[v]$ if ℓ_p^l is $\bar{y}_j$. Obviously, ψ is encoded in ϕ as $(\bigwedge_{i \in [1,m]} \xi_i \wedge \bigwedge_{j \in [1,n]} \chi_j \rightarrow$

$\bigvee_{l\in[1,r]} C_l$). While ξ_i picks the latest value of x_i in I^c_X as its truth value, χ_j checks all possible truth values of y_j as it imposes no constraints on $s_j[v]$.

We next verify that $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$ iff φ is true.

$\boxed{\Rightarrow}$ Suppose that $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$. Then there exists an $I^c_V \in \mathsf{Mod}(\mathbf{S})$ with a total order $\prec^c_v$. Define a truth assignment μ_X for X such that for each $i \in [1, m]$, $\mu_X(x_i) = 1$ if $t_i[V] = t'_i[V] = x_i$, $t_i[v] = 1$, $t'_i[v] = 0$ and $t'_i \prec^c_v t_i$; and $\mu_X(x_i) = 0$ if $t_i \prec^c_v t'_i$. Since $I^c_V \models \phi$, one can verify that μ_X satisfies $\forall Y \psi$, and hence, φ is true.

$\boxed{\Leftarrow}$ Conversely, suppose that φ is true. Let μ_X be a satisfying truth assignment for X. We define I^c_V such that for each $i \in [1, m]$ and $t_i, t'_i \in I^c_V$ with $t_i[V] = t'_i[V] = x_i$, $t_i[v] = 1$ and $t'_i[v] = 0$, if $\mu_X(x_i) = 1$ then $t'_i \prec^c_v t_i$, and $t_i \prec^c_v t'_i$ otherwise. The other currency orders can be completed arbitrarily. One can verify that $I^c_V \models \phi$ since φ is true. As a result, I^c_V is in $\mathsf{Mod}(\mathbf{S})$. □

Data complexity CPS*:* We show that CPS is NP-hard by reduction from the Betweenness problem, which is known to be NP-complete (cf. [Garey and Johnson 1979]). The Betweenness problem is to decide whether for given sets A and $B = \{(a_i, a_j, a_k) \mid a_i, a_j, a_k \in A^3\}$, there exists a bijection $\pi : A \to \{1, \ldots, |A|\}$ such that for each $(a_i, a_j, a_k) \in B$, either $\pi(a_i) < \pi(a_j) < \pi(a_k)$ or $\pi(a_k) < \pi(a_j) < \pi(a_i)$ holds.

We show that CPS is already NP-hard when $\mathbf{S}$ consists of a single temporal database instance of a fixed schema equipped with a fixed set of denial constraints. No copy functions are specified in $\mathbf{S}$.

(1) Temporal instance. The specification $\mathbf{S}$ consists of a single 5-ary relation $R(\mathsf{EID}, \mathsf{TID}, A, P, O)$. The corresponding temporal instance I is used to encode the set of triples B, as follows. For each $(a_i, a_j, a_k) \in B$ we add six tuples tuples to I: $(\mathsf{eid}, \mathsf{tid}, a_i, 1, 1)$, $(\mathsf{eid}, \mathsf{tid}, a_j, 2, 1)$ and $(\mathsf{eid}, \mathsf{tid}, a_k, 3, 1)$, and similarly, $(\mathsf{eid}, \mathsf{tid}, a_i, 3, 2)$, $(\mathsf{eid}, \mathsf{tid}, a_j, 2, 2)$ and $(\mathsf{eid}, \mathsf{tid}, a_k, 1, 2)$. Note that all tuples in I pertain to the same entity. By contrast, tid serves as a unique identifier for the triples in B. Furthermore, the O-attribute value of these tuples distinguishes between the two allowed orderings of a_i, a_j and a_k. That is, the three tuples with O-attribute set to 1 correspond to $a_i < a_j < a_k$, whereas the tuples with O-attribute set to 2 correspond to $a_k < a_j < a_i$. Finally, the attribute P indicates the position of the elements in a triple within these orderings (*i.e.,* position 1, 2, or 3). We further add an additional tuple $t_\# = (\mathsf{eid}, \#, \#, \#, \#)$ to I, where $\#$ is a symbol not used anywhere else. Intuitively, this special tuple serves as a separator between the two alternative orderings of triples in B. More specifically, in a completion I^c of I, tuples that are more recent than $t_\#$ relative to $\prec^c_A$ represent the chosen ordering of triples. Note that I consists of at most $O(|A|^3)$ tuples. The initial partial currency orders in I are empty.

(2) Denial constraints. We define a fixed number of denial constraints, which together assure that in any completion I^c of I, only one of the two alternative orderings for each triple in B is selected. More specifically, we include the following denial constraints (omitting the condition that all the involved tuples refer to same entity):

$$\sigma_1 = \forall t_1, t_2, s:\ R\big((t_1[\mathsf{TID}] = t_2[\mathsf{TID}] \wedge t_1[O] = t_2[O] \wedge s[A] = \# \wedge t_1 \prec_A s \prec_A t_2) \rightarrow t_1 \prec_A t_1\big).$$

That is, σ_1 enforces that in a completion I^c of I, all three tuples corresponding to the same alternative ordering of a triple in B are either more or less current than $t_\#$ relative to $\prec^c_A$. We further enforce that in I^c, only one of the two alternative orderings is more current than $t_\#$ relative to $\prec^c_A$. This is achieved by including two constraints, denoted by σ_2 and σ_3, that express that no pair of tuples t_1 and t_2 can exist such that

(i) t_1 and t_2 refer to the same triple but correspond to different orderings (*i.e.,* different O-attribute value); and (ii) both t_1 and t_2 are more (resp. less) current than $t_\#$ relative to $\prec^c_A$. In addition, using a constraint σ_4, we enforce that for those tuples that belong to the same triple and the same ordering, if they are more recent than $t_\#$ then they also must be ordered correctly, *i.e.,* if $t_1[P] < t_2[P]$ then also $t_1 \prec^c_A t_2$ in completions of I. Finally, we include a constraint σ_5 that enforces tuples in the selected ordering of triples (*i.e.,* those that are more current than $t_\#$) and that refer to the same value (element) in A, to be ordered consecutively in completions of I. It is readily verified that the constraints σ_2, σ_3, σ_4 and σ_5 can be expressed as denial constraints, similar to σ_1. This concludes the definition of $\mathbf{S}$.

We claim that $\mathsf{Mod}(\mathbf{S})$ is non-empty iff there exists a bijection $\pi : A \to \{1, \ldots, |A|\}$ such that for each $(a_i, a_j, a_k) \in B$, either $\pi(a_i) < \pi(a_j) < \pi(a_k)$ or $\pi(a_k) < \pi(a_j) < \pi(a_i)$.

$\boxed{\Rightarrow}$ Suppose that $\mathsf{Mod}(\mathbf{S})$ is non-empty. Let I^c be a consistent completion of I with $\prec^c_A$ the total order on tuples in I relative to the A-attribute. Let $a \in A$ and let $I(a)$ be the set of tuples t in I such that (i) $t[A] = a$; and (ii) $t_\# \prec^c_A t$. Observe that by σ_5 all such tuples appear consecutively in $\prec^c_A$. As a consequence, we can order the sets $I(a)$ such that $I(a) < I(a')$ for $a, a' \in A$ iff all tuples in $I(a)$ come before all tuples in $I(a')$ relative to $\prec^c_A$. We say that $I(a)$ is the ith block in I if it is the ith element in this order. We define the bijection $\pi : A \to \{1, \ldots, |A|\}$ by letting $\pi(a) = i$, where $I(a)$ is the ith block. Since $I^c \in \mathsf{Mod}(\mathbf{S})$ it follows from σ_1–σ_4 that π satisfies the desired condition.

$\boxed{\Leftarrow}$ Suppose that there exists a valid bijection $\pi : A \to \{1, \ldots, |A|\}$. We define a completion I^c of I in which the total order $\prec^c_A$ is defined as follows. For each $(a_i, a_j, a_k) \in B$, we first identify which of the two options is selected by π. That is, if $\pi(a_i) < \pi(a_j) < \pi(a_k)$ then $\prec^c_A$ puts after $t_\#$ all three tuples that correspond to the triple (a_i, a_j, a_k) (identified by tid) and have their O-attribute set to 1. The other three tuples that correspond to (a_i, a_j, a_k) and have O-attribute set 2 are ordered before $t_\#$ by $\prec^c_A$. As a consequence, I^c already satisfies σ_1–σ_3. Furthermore, $\prec^c_A$ also orders the three tuples in the appropriate order (using their P-attribute values) in order to satisfy σ_4. Finally, $\prec^c_A$ groups tuples that belong to same block $I(a)$ (as previously defined) consecutively in some arbitrary order, hereby ensuring that I^c also satisfies σ_5. We complete $\prec^c_A$ in an arbitrary way on the remaining tuples in I to get a total order. Clearly, I^c satisfies all constraints and hence $\mathsf{Mod}(\mathbf{S})$ is non-empty. □

Upper bound CPS*:* We next describe a decision algorithm for CPS that is in Σ^p_2 (combined complexity) and in NP (data complexity). Let $\mathbf{S}$ be a specification that consists of a collection of temporal instances $D_{(t,i)}$ of schema R_i for $i \in [1, s]$, with (1) a set Σ_i of denial constraints imposed on each $D_{(t,i)}$, and (2) a copy function $\rho_{(i,j)}$ from $D_{(t,j)}$ to $D_{(t,i)}$ for each pair of $i, j \in [1, s]$. The algorithm simply guesses a completion $\mathbf{D}^c$ and verifies whether it belongs to $\mathsf{Mod}(\mathbf{S})$, as follows:

(1) For each temporal instance $D_{(t,i)}$ and attributes A_j in R_i, guess a binary relation $\prec^c_{(i,j)}$ over the tuples in $D_{(t,i)}$.
(2) For each $i \in [1, s]$:
   (a) check whether $\prec_{i,j} \subseteq \prec^c_{(i,j)}$ and whether for each entity eid that occurs in $D_{(t,i)}$, the binary relation $\prec^c_{(i,j)}$ is a total order on all tuples $t \in D_{(t,i)}$ with $t[\mathsf{EID}] = \mathsf{eid}$. if not, reject the guess; otherwise continue;
   (b) check whether $D^c_{(t,i)} \models \Sigma_i$; if not, reject the guess; otherwise continue.
(3) For each $i, j \in [1, s]$:
   (a) check whether $\rho_{(i,j)}$ is compatible with $(D^c_{(t,i)}, \Sigma_i)$ and $(D^c_{(t,j)}, \Sigma_j)$. If not, reject the guess; otherwise return "yes".

Based on the algorithm, we present an analysis of the complexity of CPS. We start with the combined complexity. It suffices to show that steps 2 and 3 can be done using an NP or coNP oracle. Clearly, step 2(a) is in PTIME: it simply verifies whether the guessed binary relations $\prec^c_{(i,j)}$ are total orders that extend the initial partial orders. By contrast, step 2(b) consists of the validation of the denial constraints on each competed temporal instance, *i.e.,* it checks whether $D^c_{(t,i)} \models \Sigma_i$ for each $i \in [1, s]$. It is readily verified that this can done in coNP by checking whether one of the conjunctive queries, obtained by negating the denial constraints, is satisfied. Finally, step 3 is also in PTIME. Indeed, it simply verifies whether the copy functions are compatible with respect to guessed completions. Hence, the combined complexity of the algorithm is $\Sigma^p_2 = \text{NP}^{\text{NP}}$. For the data complexity, it is readily verified that step 2(b) is in PTIME. Indeed, the data complexity of evaluating denial constraints is in PTIME. Hence, the data complexity of the algorithm is in NP. □

Certain currency orders. The next question studies whether a given currency order is contained in all consistent completions of a specification. Given two temporal instances $D_{(t,1)} = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ and $D_{(t,2)} = (D, \prec'_{A_1}, \ldots, \prec'_{A_n})$ of the same schema R, we say that $D_{(t,1)}$ is *contained in* $D_{(t,2)}$, denoted by $D_{(t,1)} \subseteq D_{(t,2)}$, if $\prec_{A_j} \subseteq \prec'_{A_j}$ for all $j \in [1, n]$.

Consider a specification $\mathbf{S}$ in which there is a temporal instance $D_t = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ of schema R. A *currency order* for D_t is a temporal instance $O_t = (D, \prec'_{A_1}, \ldots, \prec'_{A_n})$ of R. Observe that O_t does not necessarily contain D_t.

COP:	The *certain ordering problem*.
INPUT:	A specification $\mathbf{S}$ in which D_t is a temporal instance, and a currency order O_t for D_t.
QUESTION:	Is it the case that $O_t \subseteq D^c_t$ for all $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$? Here D^c_t is the completion of D_t in $\mathbf{D}^c$.

Example 3.2. Consider specification $\mathbf{S}_0$ of Example 2.3. We want to know whether $s_1 \prec_{\mathsf{salary}} s_3$ is assured by every completion $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S}_0)$. To this end we construct a currency order O_t = (Emp, $\prec_{\mathsf{FN}}$, $\prec_{\mathsf{LN}}$, $\prec_{\mathsf{address}}$, $\prec_{\mathsf{salary}}$, $\prec_{\mathsf{status}}$), in which $s_1 \prec_{\mathsf{salary}} s_3$ is in $\prec_{\mathsf{salary}}$, but the partial orders for all other attributes are empty. One can verify that O_t is indeed a certain currency order, as assured by denial constraint φ_1. Similarly, one can define a currency order O'_t to check whether $t_3 \prec_{\mathsf{mgrFN}} t_4$ is entailed by all $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S}_0)$. One can readily verify that it is not the case. Indeed, there exists a $\mathbf{D}^c_1 \in \mathsf{Mod}(\mathbf{S}_0)$, such that $t_4 \prec_{\mathsf{mgrFN}} t_3$ is given in $\mathbf{D}^c_1$. ◇

We study COP together with the certain current instance problem stated below.

Certain current instances. Given a specification $\mathbf{S}$ of data currency, one naturally wants to know whether every consistent completion of $\mathbf{S}$ yields the same current instance. We say that a specification $\mathbf{S}$ of data currency is *deterministic for current instances* if for all consistent completions $\mathbf{D}^c_1, \mathbf{D}^c_2 \in \mathsf{Mod}(\mathbf{S})$, $\mathsf{LST}(\mathbf{D}^c_1) = \mathsf{LST}(\mathbf{D}^c_2)$. This definition naturally carries over to a particular relation schema R: specification $\mathbf{S}$ is said to be *deterministic for current R instances* if for all consistent completions $\mathbf{D}^c_1$, $\mathbf{D}^c_2 \in \mathsf{Mod}(\mathbf{S})$, the instance of R in $\mathsf{LST}(\mathbf{D}^c_1)$ is equal to the instance of R in $\mathsf{LST}(\mathbf{D}^c_2)$.

DCIP:	The *deterministic current instance problem*
INPUT:	A specification $\mathbf{S}$ and a relation schema R defined in $\mathbf{S}$.
QUESTION:	Is $\mathbf{S}$ deterministic for current R instances?

Example 3.3. The specification $\mathbf{S}_0$ of Example 2.3 is deterministic for current Emp instances. Indeed, for all $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S}_0)$, if D^c_{Emp} is the completion of the Emp instance in $\mathbf{D}^c$, then $\mathsf{LST}(D^c_{\mathsf{Emp}}) = \{s_3, s_4, s_5\}$. $\diamond$

Observe that when the input specification $\mathbf{S}$ in COP and DCIP is inconsistent, the conditions stated in these problems are trivially satisfied since $\mathsf{Mod}(\mathbf{S}) = \emptyset$. The following result tells us that both COP and DCIP are beyond reach in practice.

THEOREM 3.4. *For both* COP *and* DCIP*, (1) the combined complexity is Π^p_2-complete, and (2) the data complexity is coNP-complete. The complexity bounds remain unchanged when no copy functions are present. In addition, the lower bounds hold even when the input specification is assumed to be consistent.* $\square$

PROOF. It suffices to show that COP and DCIP are Π^p_2-hard (combined complexity) and coNP-hard (data complexity) when the given specification is consistent and when copy functions are absent, and that they are in Π^p_2 (combined complexity) and coNP (data complexity) when the given specification is not necessarily consistent and when copy functions may be present.

Lower bounds COP *(combined complexity):* We show that COP is Π^p_2-hard by reduction from the complement of the $\exists^*\forall^*$3DNF problem. Given an instance $\varphi = \exists X \forall Y \psi(X, Y)$ of the $\exists^*\forall^*$3DNF problem, we define a consistent specification $\mathbf{S}$ consisting of a fixed schema R_V, a corresponding temporal instance D_V, a single denial constraint but no copy functions, as well as a currency order O_t for D_V. We show that φ is false iff $O_t \subseteq D^c_V$ for every $D^c_V \in \mathsf{Mod}(\mathbf{S})$. We assume *w.l.o.g.* that $X = \{x_1, \ldots, x_m\}$, $Y = \{y_1, \ldots, y_n\}$, and ψ is a formula $C_1 \vee \cdots \vee C_r$, as in the proof of Theorem 3.1.

More specifically, we define $\mathbf{S}$ and O_t as follows.

(1) Temporal instance. The specification $\mathbf{S}$ contains the same relation schema $R_V(\mathsf{EID}, V, v, A_1, A_2, A_3, B)$ as defined in the proof of Theorem 3.1. Its temporal instance D_V is the same as I_V defined there, except that it includes an additional tuple $t_\$ = (\mathsf{eid}, \$, \$, \$, \$, \$, \$)$, where $\$$ is a distinct symbol. Intuitively, I_X, I_Y and $I_\vee$ are used to encode truth assignments μ_X for X, enumerate all truth assignments for Y, and to code disjunction, respectively, as in the proof of Theorem 3.1. The tuple $t_\$$ is used to define the currency order O_t.

(2) Currency order O_t. We define O_t such that for attribute C ranging over V, v, A_1, A_2, A_3, B and for each tuple $t \in I_V$, $t \prec_C t_\$$. That is, it requires that $t_\$$ is the current instance $\mathsf{LST}(D^c_V)$ for all consistent completions D^c_V of D_V.

(3) Denial constraints. We define a denial constraint σ to encode $\varphi = \exists X \forall Y \psi(X, Y)$ (omitting the EID attributes):

$$\sigma = \forall t_1, t'_1, \ldots, t_m, t'_m,\ s_1, \ldots, s_n,\ c_1, \ldots, c_r \big(\tau \wedge \big(\bigwedge_{i\in[1,m]} \xi_i \wedge \bigwedge_{j\in[1,n]} \chi_j \wedge \bigwedge_{l\in[1,r]} \omega_l\big) \rightarrow t_1 \prec_V t_1\big)$$

Here σ is an extension of its counterpart ϕ defined in the proof of Theorem 3.1. It encodes ψ as $\tau \rightarrow \big(\bigwedge_{i\in[1,m]} \xi_i \wedge \bigwedge_{j\in[1,n]} \chi_j \rightarrow \bigvee_{l\in[1,r]} \neg\omega_l\big)$, where ξ_i, χ_j and ω_l are the same as defined for ϕ. The additional precondition τ simply checks whether the new tuple $t_\$$ is *not* more current than all tuples t_i, t'_i, s_j, c_l in all attributes. That is, $t_\$ \prec_C t$ for some t of t_i, t'_i, s_j or c_l and for attribute C of V, v, A_1, A_2, A_3 and B, where $i \in [1, m]$, $j \in [1, m]$, and $l \in [1, r]$. This can be expressed as a conjunction of atomic formulas by leveraging $I_\vee$, which encodes disjunction, and by introducing additional universally quantified variables ranging over tuples in $I_\vee$.

One can readily verify that $\mathbf{S}$ is consistent. Indeed, for any completion D^c_V such that $\mathsf{LST}(D^c_V) = t_\$$, we have that $D^c_V \models \sigma$. Also observe that no copy functions are defined.

We next verify that φ is false iff $O_t \subseteq D^C_V$ for all $D^c_V \in \mathsf{Mod}(\mathbf{S})$.

$\boxed{\Rightarrow}$ Suppose that φ is false. As argued in the proof of Theorem 3.1, for any completion I^c_V of I_V, $I^c_V \not\models \phi$, where $I_V \subset D_V$. Hence no consistent completion D^C_V of D_V can satisfy τ, since otherwise $D^c_V \not\models \sigma$. Because τ encodes the converse of O_t, $O_t \subset D^C_V$ for all $D^c_V \models \sigma$.

$\boxed{\Leftarrow}$ Conversely, suppose that φ is true. Then one can define a consistent completion D^c_V of D_V such that $\mathsf{LST}(D^c_V) \neq t_\$$ and moreover, $D^c_V \models \sigma$. Indeed, in such a D^c_V, σ is equivalent to ϕ defined in the proof of Theorem 3.1, and as argued there, $D^c_V \models \phi$ when φ is true. Hence there exists a $D^c_V \in \mathsf{Mod}(\mathbf{S})$ such that $O_t \not\subset D^C_V$. □

Lower bounds DCIP *(combined complexity):* From the proof for COP above it follows immediately that DCIP is Π^p_2-hard for consistent specifications, in the absence of copy functions. Indeed, the currency order O_t given there defines the current instance of R_V, and hence, the proof carries over to DCIP. □

Lower bounds COP *(data complexity):* We show that COP is coNP-hard by reduction from the complement of the 3SAT problem, which is known to be NP-complete. An instance of 3SAT is a logic formula $\psi = C_1 \wedge \cdots \wedge C_r$ defined on propositional variables $X = \{x_1, \ldots, x_m\}$, where for each $i \in [1, r]$, clause C_i is of the form $\ell^i_1 \vee \ell^i_2 \vee \ell^i_3$, and for each $j \in [1, 3]$, literal ℓ^i_j is either a variable or the negation of a variable in X. The problem is to decide whether there exists a truth assignment for X that satisfies ψ. It is known that 3SAT is NP-complete (cf. [Papadimitriou 1994]). We define a consistent specification $\mathbf{S}$ consisting of a fixed schema R_C, a corresponding temporal instance I_C, a set Σ of fixed denial constraints but no copy functions, as well as a currency order O_t for I_C. We show that φ is false iff $O_t \subseteq I^c_C$ for every $I^c_C \in \mathsf{Mod}(\mathbf{S})$.

More specifically, we define $\mathbf{S}$ and O_t as follows.

(1) Temporal instance. The specification $\mathbf{S}$ contains a single relation schema $R_C(\mathsf{EID}, C, L, S, V)$. Its temporal instance I_C consists of the following tuples: For each $i \in [1, r]$, $j \in [1, 3]$, I_C contains the tuple $(\mathsf{eid}, i, j, +, x_i)$ if x_i is the jth literal in C_i; and $(\mathsf{eid}, i, j, -, x_i)$ if $\bar{x}_i$ is the jth literal in C_i. Furthermore, I_C contains a special tuple $t_\# = (\mathsf{eid}, \#, \#, \#, \#)$, where $\#$ is a special symbol not appearing anywhere else in I_C.

(2) Denial constraints. We define a set Σ of fixed denial constraints that together imply (a) tuples that are more current in one attribute are more current in all attributes; (b) if there exists a tuple t such that $t_\# \prec_C t$, then for every $i \in [1, r]$ there exists a tuple $t_i = (\mathsf{eid}, i, \pm, x)$ such that $t_\# \prec_C t_i$, where $\pm \in \{+, -\}$ and $x \in X$; and finally, (c) only one of the tuples $(\mathsf{eid}, i, j, +, x_i)$ or $(\mathsf{eid}, i, j, -, x_i)$ can be more current than $t_\#$. Intuitively, these constraints imply that the most current tuple is a tuple from I_C and that either $t_\#$ is the current tuple, or if not, every clause has at least one of its literals appear after $t_\#$ *and* in these tuples either x or $\bar{x}$ appears, for $x \in X$, but not both. No copy functions are specified.

(3) Currency order O_t. We define O_t such that $t_\#$ is more recent than any other tuple in I_C.

One can readily verify that $\mathbf{S}$ is consistent. Indeed, it suffices to take a completion that makes $t_\#$ the most current tuple. We next verify that φ is false iff $O_t \subseteq I^c_C$ for all $I^c_C \in \mathsf{Mod}(\mathbf{S})$.

$\boxed{\Rightarrow}$ Suppose that φ is true and let μ be a satisfying truth assignment for φ. Define $\prec^c_C$ such that for each $j \in [1, r]$ exactly one tuple $(\mathsf{eid}, j, \pm, x)$ comes after $t_\#$ in which the

literal is encoded by $\pm$ and x evaluates to true under μ. The currency orders on the other attributes are the same as $\prec^c_C$. This clearly results in a consistent completion in $\mathsf{Mod}(\mathbf{S})$ such that its most current tuple is different from $t_\#$. Hence, $O_t \not\subseteq I^c_C$ for all $I^c_C \in \mathsf{Mod}(\mathbf{S})$.

$\boxed{\Leftarrow}$ Conversely, suppose that $\mathbf{S}$ has a consistent completion I^c_C in which $t_\#$ is not the more current tuple. This implies that some tuples corresponding to the clauses and literals in φ are more current than $t_\#$. In particular, the denial constraints imply that the following mapping μ is well-defined: $\mu(x_i) = 1$ in case $(\mathsf{eid}, j, i, +, x_i)$ is more current than $t_\#$ for some $j \in C$, and $\mu(x_i) = 0$ in case $(\mathsf{eid}, j, i, -, x_i)$ is more current than $t_\#$ for some $j \in C$; furthermore, since all clauses contribute such a tuple, μ can be extended to a satisfying truth assignment for φ. □

Lower bounds DCIP *(data complexity):* The coNP-hardness is established in precisely way as in the previous proof. Indeed, observe that a unique current instance exists if $O_t \subseteq I^c_C$ for all $I^c_C \in \mathsf{Mod}(\mathbf{S})$, where O_t and $\mathbf{S}$ are as in the previous proof. □

Upper bounds COP*:* We provide a decision algorithm for COP that is in Π^p_2 (combined complexity) and in coNP (data complexity). In fact, we provide an Σ^p_2 (resp. NP) algorithm for the complement problem: Given a specification $\mathbf{S}$ and currency order O_t for D_t, it checks whether there exists a $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$ such that $O_t \not\subseteq (D^c_t, \prec^c_1, \ldots, \prec^c_n)$, where the latter is the consistent completion for D_t in $\mathbf{D}^c$. This problem can be decided by a minor variation of the algorithm for CPS given in the proof of Theorem 3.1. Indeed, an additional PTIME step is required that checks whether the guessed completion in Step 1 does not contain the given currency order O_t. Recall also that COP is trivially true when $\mathsf{Mod}(\mathbf{S}) = \emptyset$. It is readily verified that in this case, the algorithm will never return "yes". The algorithm thus works correctly even when $\mathbf{S}$ is not consistent. The upper bounds thus follow from the analysis of the algorithm given in the proof of Theorem 3.1. □

Upper bounds DCIP*:* We provide a Σ^p_2 (combined complexity) and NP (data complexity) algorithm that decides the complement problem. Let $\mathbf{S}$ be the given specification, the algorithm returns "yes" if $\mathbf{S}$ is not deterministic for current instances. The algorithm is as follows:

(1) Guess two completions $\mathbf{D}^c_1$ and $\mathbf{D}^c_2$ of $\mathbf{S}$.
(2) Verify the following:
   (a) $\mathbf{D}^c_1$ and $\mathbf{D}^c_2$ are both in $\mathsf{Mod}(\mathbf{S})$; and
   (b) $\mathsf{LST}(\mathbf{D}^c_1) \neq \mathsf{LST}(\mathbf{D}^2_c)$.

The correctness of the algorithm is clear, and for the same reason as in the algorithm for COP, it works even when $\mathbf{S}$ is not consistent. Observe that for combined complexity, Step 2(a) is in NP since it involves verifying denial constraints, and Step 2(b) is in PTIME. Hence, the overall combined complexity of the algorithm is in Σ^p_2. For data complexity, Step 2(a) only requires PTIME, bringing the data complexity to NP. Since the algorithm decides the complement of DCIP, the Π^p_2 upper bounds follow. □

Query answering. Given a query Q, we want to know whether a tuple t is in $Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ for all $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$.

CCQA($\mathcal{L}_Q$):	The *certain current query answering* problem.
INPUT:	A specification $\mathbf{S}$, a tuple t and a query $Q \in \mathcal{L}_Q$.
QUESTION:	Is t a certain current answer to Q *w.r.t.* $\mathbf{S}$?

We note that, similarly as for COP and DCIP, the certain current query answer problem is vacuously true when inconsistent specifications $\mathbf{S}$ are given as input.

We study CCQA($\mathcal{L}_Q$) when $\mathcal{L}_Q$ ranges over the following query languages (see, *e.g.,* [Abiteboul et al. 1995] for the details):

— CQ, the class of conjunctive queries built up from relation atoms and equality ($=$), by closing under conjunction $\wedge$ and existential quantification $\exists$;
— UCQ, unions of conjunctive queries of the form $Q_1 \cup \cdots \cup Q_k$, where for each $i \in [1, k]$, Q_i is in CQ;
— $\exists$FO$^+$, first-order logic (FO) queries built from atomic formulas, by closing under $\wedge$, *disjunction* $\vee$ and $\exists$; and
— FO queries built from atomic formulas using $\wedge$, $\vee$, *negation* $\neg$, $\exists$ and universal quantification $\forall$.

While different query languages have no impact on the data complexity of CCQA($\mathcal{L}_Q$), we next show the following: (1) disjunctions in UCQ and $\exists$FO$^+$do not incur extra complexity to CCQA (indeed, CCQA has the same complexity for CQ as for UCQ and $\exists$FO$^+$); (2) the presence of negation in FO complicates the analysis; and (3) copy functions have no impact on the complexity bounds.

THEOREM 3.5. *The combined complexity of* CCQA($\mathcal{L}_Q$) *is*
(*1*) Π_2^p*-complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*, and*
(*2*) *PSPACE-complete when* $\mathcal{L}_Q$ *is FO.*

The data complexity is coNP-complete when $\mathcal{L}_Q \in \{$*CQ*, *UCQ*, $\exists$*FO*$^+$, *FO*$\}$*. These complexity bounds remain unchanged in the absence of copy functions. In addition, the lower bounds hold even when the input specification is assumed to be consistent.* □

PROOF. It suffices to show the lower bounds when copy functions are absent from the specification and when the specification is consistent. For the upper bounds, however, copy functions may be present and specifications may be inconsistent.

Combined complexity CCQA *for CQ, UCQ and* $\exists$*FO*$^+$*:* We show that CCQA($\mathcal{L}_Q$) is Π_2^p-hard by reduction from the $\forall^*\exists^*$3CNF problem, which is known to be Π_2^p-complete [Stockmeyer 1976]. The $\forall^*\exists^*$3CNF problem is to decide, given a sentence $\varphi = \forall X \exists Y \psi$, whether φ is true. Here $X = \{ x_1, \ldots, x_m \}$, $Y = \{ y_1, \ldots, y_n \}$, and ψ is an instance $C_1 \wedge \cdots \wedge C_r$ of 3SAT, as in the proof of Theorem 3.4.

Given φ, we define a specification $\mathbf{S}$ of data currency, a query Q in CQ and a tuple t. We show that φ is true iff for each consistent completion $\mathbf{D}^c$ of $\mathbf{S}$, $t \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$. In $\mathbf{S}$, neither denial constraints nor copy functions are defined.

(1) Temporal instances. The specification $\mathbf{S}$ consists of six relation schemas $R_X(\mathsf{EID}, A_x)$, $R_\vee = (\mathsf{EID}, A, A_1, A_2)$, $R_\wedge(\mathsf{EID}, A, A_1, A_2)$, $R_\neg = (\mathsf{EID}, A, \bar{A})$, $R_{01} = (\mathsf{EID}, A)$, and $R_b = (\mathsf{EID}, B)$. The corresponding instances I_X, $I_\vee$, $I_\wedge$, $I_\neg$, I_{01} and I_b are as shown in Fig. 2. Here I_X is a temporal instance of R_X consisting of two tuples $(i, 0)$ and $(i, 1)$ for each $i \in [1, m]$. Intuitively, each consistent completion I_X^c of I_X encodes a truth assignment μ_X for X such that for each $x_i \in X$, $\mu_X(x_i) = 1$ iff $(i, 0) \prec^c_{A_x} (i, 1)$ for tuples $(i, 0)$ and $(i, 1)$ in I_X, where $\prec^c_{A_x}$ is the completion of the currency order $\prec_{A_x}$ in I_X^c. The instances $I_\vee$, $I_\wedge$ and $I_\neg$ encode disjunction, conjunction and negation, respectively. The instance I_{01} encodes the Boolean domain and I_b keeps a flag indicating whether φ is satisfiable. The initial partial currency orders in all these instances are empty. Also note that in all, except for I_X, each entity has a single tuple associated with it. Hence, completions of these instances coincide with the instances themselves and so do their corresponding current instances.

$$I_X = \begin{array}{|cc|}\hline \mathsf{EID} & A_x \\ \hline 1 & 1 \\ 1 & 0 \\ \vdots & \vdots \\ m & 1 \\ m & 0 \\ \hline \end{array} \qquad I_\vee = \begin{array}{|cccc|}\hline \mathsf{EID} & A & A_1 & A_2 \\ \hline \mathsf{eid}_1 & 0 & 0 & 0 \\ \mathsf{eid}_2 & 1 & 0 & 1 \\ \mathsf{eid}_3 & 1 & 1 & 0 \\ \mathsf{eid}_4 & 1 & 1 & 1 \\ \hline \end{array}$$

$$I_\neg = \begin{array}{|ccc|}\hline \mathsf{EID} & A & \bar{A} \\ \hline \mathsf{eid}_5 & 0 & 1 \\ \mathsf{eid}_6 & 1 & 0 \\ \hline \end{array} \qquad I_{01} = \begin{array}{|cc|}\hline \mathsf{EID} & A \\ \hline \mathsf{eid}_7 & 1 \\ \mathsf{eid}_8 & 0 \\ \hline \end{array} \qquad I_b = \begin{array}{|cc|}\hline \mathsf{EID} & A \\ \hline \mathsf{eid}_9 & 1 \\ \hline \end{array}$$

$$I_\wedge = \begin{array}{|cccc|}\hline \mathsf{EID} & A & A_1 & A_2 \\ \hline \mathsf{eid}_{10} & 0 & 0 & 0 \\ \mathsf{eid}_{11} & 0 & 0 & 1 \\ \mathsf{eid}_{12} & 0 & 1 & 0 \\ \mathsf{eid}_{13} & 1 & 1 & 1 \\ \hline \end{array}$$

Fig. 2. Temporal instances used in the lower bound proof of Theorem 3.5(1).

(2) Query. We define a CQ query Q as follows (omitting the EID-attributes):

$$Q(w) \;=\; \exists \vec{x}\, \vec{y}\, \big(Q_X(\vec{x}) \;\wedge Q_Y(\vec{y}) \;\wedge\; Q_\psi(\vec{x}, \vec{y}, w) \;\wedge\; R_b(w)\big).$$

Here $Q_X(\vec{x})$ is $\bigwedge_{i\in[1,m]} R_X(i, x_i)$ and it selects truth assignments μ_X from completions I_X^c of I_X. The sub-query $Q_Y(\vec{y})$ simply generates all truth assignments μ_Y for Y using n Cartesian products of I_{01}. Finally, $Q_\psi(\vec{x}, \vec{y}, w)$ is a CQ query that encodes the truth value of $\psi(X, Y)$ for a given truth assignment μ_X for X and μ_Y for Y, such that $w = 1$ if ψ is satisfied by μ_X and μ_Y, and $w = 0$ otherwise. The query Q_ψ can be expressed in CQ in terms of $R_\vee$, $R_\wedge$ and $R_\neg$. We illustrate the construction of Q_ψ by means of the following example. Consider the formula $\psi = C_1 \wedge C_2$, where $C_1 = x_1 \vee y_1 \vee \bar{y}_2$ and $C_2 = x_2 \vee \bar{x}_3 \vee y_3$. Then, for the clause C_1 we consider the CQ query (omitting the EID attributes) $Q_1(x_1, y_1, y_2, w_1) = \exists w_1', y_2'\ \big(I_\vee(w_1', x_1, y_1) \wedge I_\vee(w_1, w_1', y_2') \wedge I_\neg(y_2, y_2')\big)$. The query $Q_2(x_2, x_3, y_3, w_2)$ for C_2 is constructed similarly. The query Q_ψ is then given by $Q_\psi(x_1, x_2, x_3, y_1, y_2, y_3, w) = \exists w_1, w_2\ \big(Q_1(x_1, y_1, y_2, w_1) \wedge Q_2(x_2, x_3, y_3, w_2) \wedge I_\wedge(w, w_1, w_2)\big)$. It can be readily verified that Q_ψ has the desired semantics.

Hence, given a consistent specification $\mathbf{D}^c$ of $\mathbf{S}$, query Q returns $\{(1)\}$ iff for the truth assignments μ_X for X encoded by the completion I_X^c in $\mathbf{D}^c$, there exists a truth assignment μ_Y for Y such that ψ is satisfied.

(3) Tuple t. We simply define t to be (1), the constant value in I_b.

We next verify the correctness of the reduction, *i.e.,* we show that φ is true iff for all consistent completions $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$, $t \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$.

$\boxed{\Rightarrow}$ Suppose that φ is true. Then for each truth assignment μ_X for X, there exists a truth assignment μ_Y for Y such that ψ is satisfied. Hence for each consistent completion of $\mathbf{D}^c$ in $\mathsf{Mod}(\mathbf{S})$, *i.e.,* for each truth assignment μ_X for X encoded by the completion I_X^c of I_X in $\mathbf{D}^c$, there exists $\vec{y}$ such that $Q(w)$ returns $\{(1)\}$. In other words, $t \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$, *i.e.,* t is a certain current answer to Q.

$\boxed{\Leftarrow}$ Suppose that φ is false. Then there exists a truth assignment μ_X for X such that for all truth assignments for Y, ψ is not satisfied. Define a completion $\mathbf{D}^c$ of $\mathbf{S}$ such that for each $x_i \in X$, $(i, 0) \prec^c_{A_x} (i, 1)$ iff $\mu_X(x_i) = 1$ for tuples $(i, 0)$ and $(i, 1)$ in I_X^c, where I_X^c is the completion of I_X in $\mathbf{D}^c$. Clearly, $\mathbf{D}^c$ is a consistent completion of $\mathbf{S}$, and moreover, that $Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ is empty, *i.e.,* $t \notin Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$. That is, t is not a certain current answer to Q. □

Combined complexity CCQA *for FO:*. We next show that $\mathsf{CCQA}(\mathcal{L}_Q)$ is PSPACE-hard by reduction from Q3SAT, which is known to be PSPACE-complete (cf. [Papadimitriou 1994]). Given a sentence $\varphi = P_1 X_1 \ \ldots P_m X_m \psi$, Q3SAT is to decide whether φ is true, where P_i is either $\exists$ or $\forall$, and ψ is an instance of 3SAT, as in the proof of Theorem 3.4.

Given an instance φ of Q3SAT, we construct a specification $\mathbf{S}$, a query Q and a tuple t. We show that φ is true iff t is a certain current answer to Q. The reduction uses neither denial constraints nor copy functions.

(1) Temporal instances. The specification $\mathbf{S}$ consists of two relation schemas $R_c(\mathsf{EID}, C)$ and $R_b(\mathsf{EID}, B)$ with corresponding instances I_c and I_b, respectively. Here I_c includes two tuples $(1, 0)$ and $(2, 1)$ with distinct EID's, while I_b contains a single tuple $(1, 1)$. Intuitively, I_c encodes Boolean values and I_b encodes a constant 1.

(2) Query. We define the query Q in FO as follows:

$$Q(c) = P_1\vec{x}_1, \ldots, P_m\vec{x}_m\ \exists\, e\ \big(Q_{X_1}(\vec{x}_1) \wedge \cdots \wedge Q_{X_m}(\vec{x}_m) \wedge Q_\psi(\vec{x}_1, \ldots, \vec{x}_m) \wedge R_b(e, c)\big).$$

Here Q_{X_i} is $\bigwedge_{y_j \in \vec{x}_i}(\exists e(R_b(e, y_j))$, *i.e.,* it assigns Boolean values to variables in $\vec{x}_i$, and Q_ψ is the same as ψ.

(3) We let the tuple t be the constant tuple (1).

One can easily verify the following: the only consistent completion $\mathbf{D}^c$ of $\mathbf{S}$ is $\mathbf{D} = (I_c, I_b)$ itself; and (2) when posed on $\mathbf{D}$, query Q returns $\{(1)\}$ iff φ is true. Therefore, t is a certain current answer to Q iff φ is true. □

Data complexity CCQA *for CQ, UCQ, $\exists$FO$^+$ and FO:*. It suffices to show that $\mathsf{CCQA}(\mathcal{L}_Q)$ is coNP-hard when $\mathcal{L}_Q$ is CQ, query Q is fixed, and when neither denial constraints nor copy functions are defined. We prove this by reduction from the complement of 3SAT.

Given an instance $\psi = C_1 \wedge \cdots \wedge C_r$ of 3SAT, as in the proof of Theorem 3.4, we construct a specification $\mathbf{S}$ consisting of two temporal instances of fixed relation schemas, with neither denial constraints nor copy functions. We also define a fixed query Q and a tuple t. We show that ψ is *not* satisfiable iff t is a certain current answer to Q.

(1) Temporal instances. The specification $\mathbf{S}$ consists of two relational schemas $R_X = (\mathsf{EID}_x, A_x)$ and $R_{\neg\psi} = (\mathsf{EID}, \mathsf{id}_C, P_x, \mathsf{EID}_x, B_x, w)$. The corresponding instances I_X and $I_{\neg\psi}$ are defined as follows:

— The temporal instance I_X of R_X encodes truth assignments for X and consists of $\{\,(x_i, 0), (x_i, 1) \mid i \in [1, k]\,\}$. Each completion I_X^c corresponds to a truth assignment μ_X for X such that $\mu_X(x_i) = 1$ iff $(x_i, 0) \prec_x^c (x_i, 1)$.

— The temporal instance $I_{\neg\psi}$ of $R_{\neg\psi}$ encodes the negation of clauses in ψ. For each $j \in [1, r]$, clause $C_j = \ell_1^j \vee \ell_2^j \vee \ell_3^j$ is encoded with three tuples in $I_{\neg\psi}$: $(\mathsf{eid}, j, i, x_{l_i}, v_i, 1)$ for each $i \in [1, 3]$, where $x_{l_1}, x_{l_2}, x_{l_3}$ are variables in $\ell_1^j, \ell_2^j, \ell_3^j$, respectively, such that $v_i = 0$ if ℓ_i^j is x_{l_i}, and $v_i = 1$ if ℓ_i^j is $\overline{x}_{l_i}$. Here eid is a unique value for each tuple.

(2) Query Q. The query is used to check whether a truth assignment for X satisfies ψ. It is defined as:

$$\begin{aligned} Q(w) \;=\; \exists j, x_1, x_2, x_3, v_1, v_2, v_3, \mathsf{eid}_1, \mathsf{eid}_2, \mathsf{eid}_3\ \big(&R_X(x_1, v_1) \wedge R_X(x_2, v_2) \wedge R_X(x_3, v_3) \wedge \\ &R_{\neg\psi}(\mathsf{eid}_1, j, 1, x_1, v_1, w) \wedge R_{\neg\psi}(\mathsf{eid}_2, j, 2, x_2, v_2, w) \wedge R_{\neg\psi}(\mathsf{eid}_3, j, 3, x_3, v_3, w)\big). \end{aligned}$$

Given a consistent completion $\mathbf{D}^c$ of $\mathbf{S}$, the query Q returns a singleton $\{(1)\}$ iff there exists a clause C_j such that the truth assignment encoded in I_X^c of $\mathbf{D}^c$ does not satisfy any literal in C_j. Indeed, the completed currency order $\prec_x^c$ in I_X^c of $\mathbf{D}^c$ ensures that for each $x_i \in X$, a unique truth value is selected for each variable in the current instance $\mathsf{LST}(I_X^c)$. The query Q is evaluated on $\mathsf{LST}(I_X^c)$ and $\mathsf{LST}(I_{\neg\psi}^c)$, and it checks whether each and every literal in C_j is false, *i.e.,* the clause is not satisfied by the truth assignment. It returns $\{(1)\}$ iff there exists at least one C_j that is not satisfied.

(3) Tuple t. The tuple t is simply defined to be (1).

We next verify the correctness of the reduction. That is, we show that ψ is not satisfiable iff for all consistent completions $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$, $t \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$.

$\boxed{\Rightarrow}$ Suppose that ψ is not satisfiable. Then for each truth assignment for X, ψ is not satisfied. From the discussion above it follows that for each consistent completion of $\mathbf{D}^c$ in $\mathsf{Mod}(\mathbf{S})$, $t \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$, *i.e.,* t is a certain current answer to Q.

$\boxed{\Leftarrow}$ Conversely, assume that ψ is satisfiable. Then there exists a truth assignment μ_X for X that satisfies ψ. Define a completion $\mathbf{D}^c$ of $\mathbf{S}$ such that for each $x_i \in X$, $(x_i, 0) \prec^c_x (x_i, 1)$ iff $\mu(x_i) = 1$ for tuples $(x_i, 0)$ and $(x_i, 1)$ in I^c_X, where I^c_X is the completion of I_X in $\mathbf{D}$. It is readily verified that $\mathbf{D}^c$ is a consistent completion of $\mathbf{S}$ and furthermore, $Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ is empty. In other words, t is not a certain current answer to Q. □

Upper bounds CCQA:. We establish the matching upper bounds by providing a nondeterministic algorithm for the complement problem. The algorithm checks, given a specification $\mathbf{S}$, a query Q and a tuple t as input, whether t is *not* a certain current answer to Q, as follows.

(1) Guess a completion $\mathbf{D}^c$ of $\mathbf{S}$.
(2) Check whether $\mathbf{D}^c$ is in $\mathsf{Mod}(S)$; if not, reject the guess; otherwise continue.
(3) Check whether $t \notin Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$, and return "no" if it is the case.

The correctness of the algorithm is clear. We next analyze its complexity. For the combined complexity, one can verify the following. (a) Step 2 is in NP; and (b) Step 3 is in coNP when $\mathcal{L}_Q$ is $\exists$FO$^+$, and is in PSPACE when $\mathcal{L}_Q$ is FO. Putting these together, the combined complexity of $\mathsf{CCQA}(\mathcal{L}_Q)$ is in Π^p_2 when $\mathcal{L}_Q$ is CQ, UCQ or $\exists$FO$^+$, and it is in NPSPACE = PSPACE when $\mathcal{L}_Q$ is FO. For the data complexity, observe that Step 2 is in PTIME when denial constraints are fixed, and that Step 3 is in PTIME when Q is fixed no matter what query language Q is in. Hence for CQ, UCQ, $\exists$FO$^+$and FO, the data complexity of $\mathsf{CCQA}(\mathcal{L}_Q)$ is in coNP. □

Special cases. Worse still, the absence of denial constraints does not make our lives easier when it comes to CCQA. Indeed, in the proof of Theorem 3.5, the lower bounds of CCQA are verified using neither denial constraints nor copy functions. This is in contrast to Theorem 6.1 to be seen shortly, which tells us that when denial constraints are absent, CPS, COP and DCIP all become easier.

COROLLARY 3.6. *In the absence of denial constraints,* $\mathsf{CCQA}(\mathcal{L}_Q)$ *remains coNP-hard (data complexity) and* Π^p_2*-hard (combined complexity) even for CQ, and PSPACE-hard (combined complexity) for FO.* □

PROOF. These results follow immediately from the proof of Theorem 3.5. Indeed, a close examination of the proofs of that theorem reveals that no denial constraints are used in its reductions for verifying the lower bounds. □

Theorem 3.5 shows that the complexity of CCQA for CQ is rather robust: adding disjunctions does not increase the complexity. We next investigate the impact of removing Cartesian product from CQ on the complexity of CCQA. We consider SP queries, which are CQ queries of the form

$$Q(\vec{x}) = \exists e\, \vec{y}\, \big(R(e, \vec{x}, \vec{y}) \ \wedge \ \psi\big),$$

where ψ is a conjunction of equality atoms and $\vec{x}$ and $\vec{y}$ are disjoint sequences of variables in which no variable appears twice. In other words, SP queries support projection and selection only. For instance, the queries $Q_1 - Q_4$ of Example 1.1 are SP queries. SP queries in which ψ is a tautology are referred to as *identity queries*.

Unfortunately, the following result tells us that in the presence of denial constraints, CCQA is no easier for identity queries than for $\exists$FO$^+$.

COROLLARY 3.7. *For SP queries,* CCQA*(SP) is coNP-complete (data complexity) and* Π_2^p*-complete (combined complexity) in the presence of denial constraints, even for identity queries.* □

PROOF. It has been verified in the proof of Theorem 3.5 that CCQA($\mathcal{L}_Q$) is in Π_2^p (combined complexity) and coNP (data complexity) for CQ queries, which include identity queries. Hence it suffices to show that CCQA is coNP-hard (data complexity) and is Π_2^p-hard (combined complexity) for identity queries.

These lower bounds are verified by showing the Σ_2^p-hardness (resp. NP-hardness) of the complement problem, which is to decide whether a given tuple does *not* belong to the certain current answers. We prove this by reduction from the CPS problem, which is NP-complete (data complexity) and Σ_2^p-complete (combined complexity) by Theorem 3.1.

Given a specification $\mathbf{S}$, we define a specification $\mathbf{S}'$ that extends $\mathbf{S}$ by adding a binary temporal instance $R_N(\mathsf{EID}, A)$, of which the temporal instance I_N consists of two tuples s and t that refer to the same entity. Neither denial constraints nor copy functions are defined on R_N. Let Q be the identity query on R_N, *i.e.,* $Q(x, y) = R_N(x, y)$. Then t (or equivalently, s) is not in the certain current answer of Q *w.r.t.* $\mathbf{S}'$ if CPS($\mathbf{S}$) is true.

$\boxed{\Rightarrow}$ Suppose that CPS($\mathbf{S}$) is true. Then there exists a consistent completion $\mathbf{D}^c \in$ Mod($\mathbf{S}$). In addition, $\mathbf{D}^c$ can be extended to two distinct completions $(\mathbf{D}^c)'$ of $\mathbf{S}$ such that in one completion s is the current answer to Q, whereas in the other completion t is the current answer. Since these two tuples are distinct, the certain current answer to Q is empty and hence t is not a certain answer.

$\boxed{\Leftarrow}$ Suppose that t is not a certain current answer to the identity query on R_N. Then there exists a consistent completion $(\mathbf{D}')^c$ of $\mathbf{S}'$ such that t does not belong to the current instance of I_N^c, where I_N^c is the completion of I_N in $(\mathbf{D}')^c$. It can be readily verified that when restricted to instances in $\mathbf{S}$, $(\mathbf{D}')^c$ induces a consistent completion $\mathbf{D}^c$ for $\mathbf{S}$. Hence, CPS($\mathbf{S}$) is true. □

In Section 6 we shall identify tractable cases for CPP, COP, DCIP and CCQA(SP) in the absence of denial constraints.

4. CURRENCY PRESERVATION IN DATA COPYING

As we have seen earlier, copy functions tell us what data values in a relation have been imported from other data sources. Naturally we want to leverage the imported values to improve query answers. This gives rise to the following questions: do the copy functions import sufficient current values for answering a query Q? If not, how do we extend the copy functions such that Q can be answered with more up-to-date data? To answer these questions we introduce a notion of currency-preserving copy functions.

We consider a specification $\mathbf{S}$ of data currency consisting of two collections of temporal instances (data sources) $\mathbf{D} = (D_1, \ldots, D_p)$ and $\mathbf{D}' = (D'_1, \ldots, D'_q)$, with (1) a set Σ_i (resp. Σ'_j) of denial constraints on D_i for each $i \in [1, p]$ (resp. D'_j for $j \in [1, q]$), and (2) a collection $\overline{\rho}$ of copy functions $\rho_{(j,i)}$ that imports tuples from D'_j to D_i, for $i \in [1, p]$ and $j \in [1, q]$, *i.e.,* all the functions of $\overline{\rho}$ import data from $\mathbf{D}'$ to $\mathbf{D}$.

Extensions. To formalize currency preservation, we first present the following notions. Assume that $D_i = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ and $D'_j = (D', \prec_{B_1}, \ldots, \prec_{B_m})$ are temporal instances of relation schemas $R_i = (\mathsf{EID}, A_1, \ldots, A_n)$ and $R'_j = (\mathsf{EID}, B_1, \ldots, B_m)$, respectively. Assume that $n \leq m$. An *extension* of D_i is a temporal instance $D_i^e =$

	FN	LN	address	salary	status
s'_1:	Mary	Dupont	6 Main St	60k	married
s'_2:	Mary	Dupont	6 Main St	80k	married
s'_3:	Mary	Smith	2 Small St	80k	divorced

Fig. 3. Relation Mgr

$(D^e, \prec^e_{A_1}, \ldots, \prec^e_{A_n})$ of R_i such that (1) $D \subseteq D^e$, (2) $\prec_{A_h} \subseteq \prec^e_{A_h}$ for all $h \in [1, n]$, and (3) $\pi_{\mathsf{EID}}(D^e) = \pi_{\mathsf{EID}}(D)$. Intuitively, D^e_i extends D_i by adding new tuples for those entities that are already in D_i. It does not introduce new entities.

Consider two copy functions: $\rho_{(j,i)}$ imports tuples from D'_j to D_i, and $\rho^e_{(j,i)}$ from D'_j to D^e_i, both of signature $R_i[\vec{A}] \Leftarrow R'_j[\vec{B}]$, where $\vec{A} = (A_1, \ldots, A_n)$ and $\vec{B}$ is a sequence of n attributes in R'_j. We say that $\rho^e_{(j,i)}$ *extends* $\rho_{(j,i)}$ if

(1) D^e_i is an extension of D_i;
(2) for each tuple t in D_i, if $\rho_{(j,i)}(t)$ is defined, then so is $\rho^e_{(j,i)}(t)$ and moreover, $\rho^e_{(j,i)}(t) = \rho_{(j,i)}(t)$;
(3) for each tuple t in $D^e_i \setminus D_i$, there exists a tuple s in D'_j such that $\rho^e_{(j,i)}(t) = s$.

We refer to D^e_i as the *extension of* D_i *by* $\rho^e_{(j,i)}$.

Observe that D^e_i is not allowed to expand arbitrarily: (a) each new tuple t in D^e_i is copied from a tuple s in D'_j; and (b) no new entity is introduced. Note that only copy functions that cover all attributes but EID of R_i can be extended. This assures that all the attributes of a new tuple are in place.

An *extension* $\overline{\rho}^e$ of $\overline{\rho}$ is a collection of copy functions $\rho^e_{(j,i)}$ such that $\overline{\rho}^e \neq \overline{\rho}$ and moreover, for all $i \in [1, p]$ and $j \in [1, q]$, either $\rho^e_{(j,i)}$ is an extension of $\rho_{(j,i)}$, or $\rho^e_{(j,i)} = \rho_{(j,i)}$. We denote the set of all extensions of $\overline{\rho}$ as $\mathsf{Ext}(\overline{\rho})$.

For each $\overline{\rho}^e$ in $\mathsf{Ext}(\overline{\rho})$, we denote by $\mathbf{S}^e$ the *extension of* $\mathbf{S}$ *by* $\overline{\rho}^e$, which consists of the same $\mathbf{D}'$ and denial constraints as in $\mathbf{S}$, but has copy functions $\overline{\rho}^e$ and $\mathbf{D}^e = (D^e_1, \ldots, D^e_p)$, where D^e_i is the union of extensions of D_i, one for each $\rho^e_{(j,i)}$, for $j \in [1, q]$.

Currency preservation. We are now ready to define currency preservation. Consider a collection $\overline{\rho}$ of copy functions in a specification $\mathbf{S}$. We say that $\overline{\rho}$ is *currency preserving* for a query Q *w.r.t.* $\mathbf{S}$ if (a) $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$, and moreover, (b) for all $\overline{\rho}^e \in \mathsf{Ext}(\overline{\rho})$ such that $\mathsf{Mod}(\mathbf{S}^e) \neq \emptyset$, we have that

$$\bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} Q\big(\mathsf{LST}(\mathbf{D}^c)\big) \;=\; \bigcap_{\mathbf{D}^c_e \in \mathsf{Mod}(\mathbf{S}^e)} Q\big(\mathsf{LST}(\mathbf{D}^c_e)\big).$$

Intuitively, $\overline{\rho}$ is currency preserving if (1) $\overline{\rho}$ is meaningful; and (2) for each extension $\overline{\rho}^e$ of $\overline{\rho}$ that makes sense, the certain current answers to Q are not improved by $\overline{\rho}^e$, *i.e.,* no matter what additional tuples are imported for those entities in $\mathbf{D}$, the certain current answers to Q remain unchanged.

Example 4.1. As shown in Fig. 3, relation Mgr collects manager records. Consider a specification $\mathbf{S}_1$ consisting of the following: (a) temporal instances Mgr of Fig. 3 and Emp of Fig. 1, in which partial currency orders are empty for all attributes; (b) denial constraints φ_1–φ_3 of Example 2.1 and

$$\varphi_5\colon \forall s, t : \mathsf{Mgr}\ \big((s[\mathsf{EID}] = t[\mathsf{EID}] \ \wedge\ s[\mathsf{status}] = \text{“divorced”} \wedge t[\mathsf{status}] = \text{“married”}) \ \rightarrow\ t \prec_{\mathsf{LN}} s\big),$$

i.e., if s[status] is divorced and t[status] is married, then s is more current than t in LN; and (c) a copy function ρ with signature $\mathsf{Emp}[\vec{A}] \Leftarrow \mathsf{Mgr}[\vec{A}]$, where $\vec{A}$ is (FN, LN, address,

salary, status), such that $\rho(s_3) = s'_2$, *i.e.,* s_3 of Emp is copied from s'_2 of Mgr. Obviously $\mathbf{S}_1$ is consistent.

Recall query Q_2 of Example 1.1, which is to find Mary's current last name. For Q_2, ρ is not currency preserving. Indeed, there is an extension ρ_1 of ρ by copying s'_3 to Emp. In all consistent completions of the extension Emp_1 of Emp by ρ_1, the answer to Q_2 is Smith. However, the answer to Q_2 in all consistent completions of Emp is Dupont (see Examples 1.1 and 2.5). In contrast, ρ_1 is currency preserving for Q_2: copying more tuples from Mgr (*i.e.,* tuple s'_1) to Emp does not change the answer to Q_2 in Emp_1. $\diamond$

Deciding currency preservation. There are several decision problems associated with currency-preserving copy functions, which we shall investigate in the next section. The first problem is to decide whether the given copy functions have imported all necessary current data for answering a query. In practice, one often repeatedly issues a (fixed) load of queries on a database $\mathbf{D}_t$ that imports data from multiple sources. Each time before the queries are executed, CPP is to ensure that the current values needed for answering the queries have been imported and updated from the data sources. The need for the check is evident since the data sources are typically *dynamic* in the real world, *i.e.,* incrementally updated by including new information. CPP aims to keep $\mathbf{D}_t$ up-to-date *w.r.t.* the dynamic data sources, and to extend copy functions by importing current information from those data sources that was overlooked.

CPP($\mathcal{L}_Q$):	The *currency preservation problem*.
INPUT:	A query Q in $\mathcal{L}_Q$, and a specification $\mathbf{S}$ of data currency with copy functions $\overline{\rho}$.
QUESTION:	Is $\overline{\rho}$ currency preserving for Q?

Extending copy functions. Consider a consistent specification $\mathbf{S}$ in which $\overline{\rho}$ is not currency preserving for a query Q. The next problem is to decide whether $\overline{\rho}$ in $\mathbf{S}$ can be extended to be currency preserving for Q at all. Here we consider consistent specifications $\mathbf{S}$ only, since when $\mathbf{S}$ is inconsistent, one cannot extend it and make it currency preserving (see more detailed discussions in Section 5).

ECP($\mathcal{L}_Q$):	The *existence problem*.
INPUT:	A query Q in $\mathcal{L}_Q$, and a consistent specification $\mathbf{S}$ with non-currency-preserving $\overline{\rho}$
QUESTION:	Does there exist $\overline{\rho}^e$ in Ext($\overline{\rho}$) that is currency preserving for Q?

Bounded extension. We also want to know whether it suffices to extend $\overline{\rho}$ by copying additional data of a bounded size, and make it currency preserving and up-to-date. That is, whether $\overline{\rho}$ can be made currency preserving with bounded cost.

BCP($\mathcal{L}_Q$):	The *bounded copying problem*.
INPUT:	$\mathbf{S}$, $\overline{\rho}$ and Q as in CPP, and a positive number k.
QUESTION:	Does there exist $\overline{\rho}^e \in$ Ext($\overline{\rho}$) such that $\overline{\rho}^e$ is currency preserving for Q and $\|\overline{\rho}^e\| \leq k + \|\overline{\rho}\|$?

5. DECIDING CURRENCY PRESERVATION

We next study the decision problems in connection with currency-preserving copy functions, namely, CPP($\mathcal{L}_Q$), ECP($\mathcal{L}_Q$) and BCP($\mathcal{L}_Q$) when $\mathcal{L}_Q$ is CQ, UCQ, $\exists$FO$^+$or FO. We provide their combined complexity and data complexity bounds.

Checking currency preservation. We first investigate $\mathsf{CPP}(\mathcal{L}_Q)$, the problem of deciding whether a collection of copy functions in a given specification is currency preserving for a query Q. We show that CPP is nontrivial. Indeed, its combined complexity is already Π_3^p-hard when Q is in CQ, and it is PSPACE-complete when Q is in FO.

One might be tempted to think that fixing denial constraints would make our lives easier. Indeed, in practice denial constraints are often predefined and fixed, and only data, copy functions and query vary. Moreover, as shown in Theorem 3.1 for the consistency problem, fixing denial constraints indeed helps there. However, it does not simplify the analysis of the combined complexity when it comes to CPP. Even when both query and denial constraints are fixed, the problem is Π_2^p-complete (data complexity).

THEOREM 5.1. *For* $\mathsf{CPP}(\mathcal{L}_Q)$*, the combined complexity is*
(*1*) Π_3^p*-complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*, and*
(*2*) *PSPACE-complete when* $\mathcal{L}_Q$ *is FO.*
(*3*) *Its data complexity is* Π_2^p*-complete when* $\mathcal{L}_Q \in \{$*CQ,UCQ,* $\exists$*FO*$^+$*,FO*$\}$.

The combined complexity bounds remain unchanged when denial constraints and copy functions are fixed.

PROOF. We show that $\mathsf{CPP}(\mathcal{L}_Q)$ is (a) Π_3^p-hard when Q is in CQ and in Π_3^p when Q is in $\exists$FO$^+$(combined complexity), (b) PSPACE-complete when Q is in FO (combined complexity); (c) Π_2^p-hard when Q is in CQ and in Π_2^p when Q is in FO (data complexity).

Combined complexity CPP *for CQ:* It suffices to show that $\mathsf{CPP}(\mathrm{CQ})$ is already Π_3^p-hard. We verify this by reduction from the complement of the $\exists^*\forall^*\exists^*$3CNF problem, which is known to be Σ_3^p-complete [Stockmeyer 1976]. The $\exists^*\forall^*\exists^*$3CNF problem is to determine, given a sentence $\varphi = \exists X \forall Y \exists Z\, \psi(X, Y, Z)$, whether φ is true. Here $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_m\}$, $Z = \{z_1, \ldots, z_k\}$, and ψ is an instance $C_1 \wedge \cdots \wedge C_r$ of 3SAT, as described in the proof of Theorem 3.4.

Given an instance $\varphi = \exists X \forall Y \exists Z\, \psi(X, Y, Z)$ of the $\exists^*\forall^*\exists^*$3CNF problem, we define a specification $\mathbf{S}$ (with copy functions $\bar{\rho}$) and a CQ query Q, such that φ is true iff the copy functions $\bar{\rho}$ in $\mathbf{S}$ are *not* currency preserving for Q.

(1) Temporal instances. The specification $\mathbf{S}$ consists of data sources $\mathbf{D}'$ and $\mathbf{D}$, where $\mathbf{D}$ consists of eight relational schemas: $R_{01}(\mathsf{EID}, A)$, $R_X(\mathsf{EID}, X, V)$, $R_Y(\mathsf{EID}, Y, V)$, $R_\vee(\mathsf{EID}, B, A_1, A_2)$, $R_\wedge(\mathsf{EID}, B, A_1, A_2)$, $R_\neg(\mathsf{EID}, A, \bar{A})$, $R_{ac}(\mathsf{EID}, A_1, A_2)$, $R_b(\mathsf{EID}, C)$; and $\mathbf{D}'$ consists of two relations $R'_X(\mathsf{EID}, X, V)$ and $R'_b(\mathsf{EID}, C)$. The corresponding temporal instances I_{01}, $I_\vee$, $I_\wedge$ and $I_\neg$ are shown in Fig. 2. Figure 4 shows the remaining instances. Here I_X (resp. I_Y) is used to represent truth assignments of variables in X (resp. Y), and I_{ac} is an auxiliary instance needed to convert 0 to a constant a distinct from c and d, and constant 1 to c. Furthermore, I'_X also encodes truth values of X, but with initial currency orders defined. We also use I_b and I'_b to control the latest value after data are copied. In all these temporal instances we leave the initial partial currency orders empty, except for I'_X and I'_b, as shown in Fig. 4.

(2) Copy function. We use two functions $\rho_1 : R_X[X, V] \Leftarrow R'_X[X, V]$ and $\rho_2 : R_b[\mathsf{EID}, C] \Leftarrow R'_b[\mathsf{EID}, C]$, initially empty. Thus $\bar{\rho} = \{\rho_1, \rho_2\}$ (no copy functions are defined on R_Y). We constrain the possible extensions of $\bar{\rho}$ by enforcing that each entity in all instances has only two possible tuples. These can be expressed as *fixed* denial constraints.

(3) Query. The CQ query Q is defined as follows (omitting the EID attributes):

$$Q(v) = \exists \vec{x}\ \exists \vec{y}\ \exists \vec{z}\ \big(Q_X(\vec{x}) \wedge Q_Y(\vec{y}) \wedge Q_Z(\vec{z})\ \wedge\ Q_\psi(\vec{x}, \vec{y}, \vec{z}, v) \wedge R_b(v)\big).$$

Here $Q_X(\vec{x})$ (resp. $Q_Y(\vec{y})$) extracts a truth assignment of n variables in X (resp. m variables in Y) by accessing R_X (resp. R_Y), and $Q_Z(\vec{z})$ generates all k binary tuples by means of Cartesian products of R_{01}. Furthermore, $Q_\psi(\vec{x}, \vec{y}, \vec{z}, v)$ is a CQ query that

$$I_X = \begin{array}{|c|c|c|}\hline \mathsf{EID} & X & V \\ \hline 1 & x_1 & 0 \\ 1 & x_1 & 1 \\ \cdots & \cdots & \cdots \\ n & x_n & 0 \\ n & x_n & 1 \\ \hline \end{array} \qquad I_Y = \begin{array}{|c|c|c|}\hline \mathsf{EID} & Y & V \\ \hline 1 & y_1 & 0 \\ 1 & y_1 & 1 \\ \cdots & \cdots & \cdots \\ m & y_m & 0 \\ m & y_m & 1 \\ \hline \end{array}$$

$$I'_X = \begin{array}{cc|c|c|c|l} & & \mathsf{EID} & B & V & \\ \hline s_1 & & 1 & x_1 & 0 & \\ s_2 & & 1 & x_1 & 1 & s_1 \prec_V s_2 \\ s'_1 & & n+1 & x_1 & 0 & \\ s'_2 & & n+1 & x_1 & 1 & s'_2 \prec_V s'_1 \\ & & \cdots & \cdots & \cdots & \\ s_{2n-1} & & n & x_n & 0 & \\ s_{2n} & & n & x_n & 1 & s_{2n-1} \prec_V s_{2n} \\ s'_{2n-1} & & 2n & x_n & 0 & \\ s'_{2n} & & 2n & x_n & 1 & s'_{2n} \prec_V s'_{2n-1} \\ \hline \end{array}$$

$$I_{ac} = \begin{array}{|c|c|c|}\hline \mathsf{EID} & A_1 & A_2 \\ \hline \mathsf{eid}_1 & 0 & a \\ \mathsf{eid}_2 & 1 & c \\ \hline \end{array} \qquad I_b = \begin{array}{|c|c|}\hline \mathsf{EID} & C \\ \hline 1 & c \\ 1 & d \\ \hline \end{array}, \qquad I'_b = \begin{array}{cc|c|c|l} & & \mathsf{EID} & C & \\ \hline u_1 & & 1 & c & \\ u_2 & & 1 & d & \\ & & u_2 \prec_C u_1 & & \end{array}$$

Fig. 4. Temporal instances used in lower bound proof of Theorem 5.1(1).

encodes the truth value of $\psi(X, Y, Z)$ for a given truth assignment μ_X for X, μ_Y for Y and μ_Z for Z, such that $v = c$ if ψ is satisfied by μ_X, μ_Y and μ_Z, and a value distinct a from c and d otherwise. The query Q_ψ can be expressed in CQ in terms of $R_\vee$, $R_\wedge$, $R_\neg$ and R_{ac}, along the same lines as in the proof of Theorem 3.5(1).

We next explain the intuition behind the reduction. Recall that Q is evaluated over the current instance $\mathsf{LST}(D^c)$ of every consistent completion $\mathbf{D}^c$ of $\mathbf{S}$. Here $Q\big(\mathsf{LST}(D^c)\big)$ returns a nonempty set only if for the truth assignments μ_X for X and μ_Y for Y encoded in $\mathsf{LST}(I^c_X)$ and $\mathsf{LST}(I^c_Y)$, respectively, there exists a truth assignment μ_Z for Z such that ψ is satisfied, and *in addition*, $\mathsf{LST}(I^c_b) = \{(1, c)\}$. Since we regard the tuples in all instances (except I_X, I_Y and I_b) as separate entities, we have that $\mathsf{LST}(I^c_{01}) = I_{01}$, $\mathsf{LST}(I^c_\vee) = I_\vee$, $\mathsf{LST}(I^c_\wedge) = I_\wedge$ and $\mathsf{LST}(I^c_\neg) = I_\neg$, for any completion of these instances. By contrast, $\mathsf{LST}(I^c_X)$, $\mathsf{LST}(I^c_Y)$ and $\mathsf{LST}(I^c_b)$ may vary in different completions. It is readily verified that the certain current query answer of Q *w.r.t.* $\mathbf{S}$ is empty. Indeed, $\mathsf{LST}(I^c_b)$ can be either $\{(1, c)\}$ or $\{(1, d)\}$. When $\mathsf{LST}(I^c_b)$ is $\{(1, d)\}$, $Q\big(\mathsf{LST}(D^c)\big)$ is empty, and hence so is the certain current query answer of Q *w.r.t.* $\mathbf{S}$.

Now let us consider the impact of extending copy functions $\bar{\rho} = \{\rho_1, \rho_2\}$. Let $\bar{\rho}^e$ be an extension of $\bar{\rho}$. Denote by $\mathbf{S}^e$ the corresponding specification. Observe that $\bar{\rho}^e$ can extend ρ_2 such that $\mathsf{LST}(I^c_b) = \{(1, c)\}$ given the currency order defined on I'_b. In addition, an extension ρ^e_1 of ρ_1 may limit the set of possible truth assignments for X, which are realized as $\mathsf{LST}(I^c_X)$, for completions of $\mathbf{S}^e$, due to the partial currency order defined in I'_X. For instance, suppose that $\rho^e_1\big((i, x_i, 0)\big) = s_{2i-1}$ and $\rho^e_1\big((i, x_i, 1)\big) = s_{2i}$ then the latest value of x_i is 1 since $s_{2i-1} \prec_V s_{2i}$. By contrast, when $\rho^e_1\big((i, x_i, 0)\big) = s'_{2i-1}$ and $\rho^e_1\big((i, x_i, 1)\big) = s'_{2i}$ then the latest value of x_i is 0 since $s'_{2i} \prec_V s'_{2i-1}$. Hence we can extend the copy function ρ_1 to select a particular truth assignment μ_X for X.

One can readily verify that $\mathbf{S}$ is consistent, *i.e.,* $\mathsf{Mod}(\mathbf{S})$ is nonempty.

We next show that the reduction above is correct, *i.e.,* φ is true iff the copy functions $\bar{\rho}$ in $\mathbf{S}$ are *not* currency preserving for Q.

$\boxed{\Rightarrow}$ Assume that φ is true. Then there exists a truth assignment μ^0_X for X such that for all μ_Y of Y, there exists μ_Z for Z that satisfies ψ together with μ^0_X and μ_Y. Define an extension $\bar{\rho}^e$ of $\bar{\rho}$ such that in its extended specification $\mathbf{S}^e$, for any completion D^c_e of $\mathbf{S}^e$, $\mathsf{LST}(I^c_X)$ represents μ^0_X, and $\mathsf{LST}(I^c_b) = \{(1, c)\}$. As argued above, this is possible. Then $Q\big(\mathsf{LST}(D^c_e)\big)$ is nonempty and in fact, the certain current answer to Q *w.r.t.* $\mathbf{S}^e$ is

nonempty, since no matter how I_Y is completed, the answer to Q is nonempty in this completion. As remarked earlier, the certain current answer to Q *w.r.t.* $\mathbf{S}$ is empty. Hence $\bar{\rho}$ is not currency preserving.

$\boxed{\Leftarrow}$ Conversely, assume that φ is false. Then for all truth assignments μ_X for X, there exists a truth assignment μ_Y for Y, such that for all μ_Z for Z, ψ is not satisfied by μ_X, μ_Y and μ_Z. As a result, no matter how we extend $\bar{\rho}$, the certain current answer to Q *w.r.t.* the extended specification remains empty. Indeed, for any extension of ρ_1 that encodes μ_X, there is a completion of I_Y that encodes a truth assignment μ_Y for Y and makes ψ false no matter what μ_Z for Z is considered. In other words, for such completions D_e^c of the extended specification $\mathbf{S}^e$, $Q\big(\mathsf{LST}(D_e^c)\big)$ is empty and hence, so is the certain current answer to Q *w.r.t.* $\mathbf{S}^e$. Therefore, $\bar{\rho}$ is currency preserving. □

Combined complexity CPP *for FO:* We next show that CPP(FO) is PSPACE-hard. We prove this by reduction from the complement of the Q3SAT problem, which is PSPACE-complete (cf. [Papadimitriou 1994]). We refer to the proof of Theorem 3.5(2) for the statement of the Q3SAT problem. Given a instance $\varphi = P_1X_1 \cdots P_mX_m\psi$ of Q3SAT, we define a specification $\mathbf{S}$ with a collection $\bar{\rho}$ of copy functions and a query Q. We show that φ is true iff $\bar{\rho}$ is *not* currency preserving for Q.

(1) We construct $\mathbf{S}$ as follows. It consists of data sources $\mathbf{D}'$ and $\mathbf{D}$, where $\mathbf{D}'$ has a single relation I_b' and $\mathbf{D}$ consists of two relations I_{01} and I_b. Here the instance I_{01} is the same as in the previous proof, $I_b = \{(\mathsf{eid}, c)\}$ and $I_b' = \{(\mathsf{eid}, c), (\mathsf{eid}, d)\}$. No partial currency orders are present. A single copy function ρ is defined from I_b' to I_b which maps $\rho\big((\mathsf{eid}, c)\big) = (\mathsf{eid}, c)$. No denial constraints are defined in $\mathbf{S}$.

(2) We define the query Q in FO as follows (omitting the EID attributes):

$$Q(v) = P_1\vec{x}_1, \ldots, P_m\vec{x}_m \big(Q_{X_1}(\vec{x}_1) \wedge \cdots \wedge Q_{X_m}(\vec{x}_m) \wedge Q_\psi(\vec{x}_1, \ldots, \vec{x}_m) \wedge R_b(v)\big).$$

Here Q_{X_i} generates all $|\vec{x}_i|$-ary binary tuples by means of Cartesian products of R_{01}, and Q_ψ is the same as ψ.

We show that this coding is indeed a reduction from the complement of Q3SAT to CPP(FO). Observe that $\mathsf{LST}(\mathbf{D}^c) = \mathbf{D}$ for any completion $\mathbf{D}^c$ of $\mathbf{D}$ and hence the certain current answer to Q *w.r.t.* $\mathbf{S}$ coincides with $Q(\mathbf{D})$. Furthermore, there is only one possible extension ρ^e of ρ in which the tuple (eid, d) is copied from I_b' to I_b. Let $\mathbf{D}^e = \big(I_{01}, I_b \cup \{(\mathsf{eid}, d)\}\big)$. It is readily verified that there are two possible completions of $\mathbf{D}^e$ with corresponding current instances $\mathbf{D}_1 = \mathbf{D}$ and $\mathbf{D}_2 = \big(I_{01}, \{(\mathsf{eid}, d)\}\big)$.

Assume first that φ is true. Then the current answer to Q in $\mathbf{D}$ is (c), whereas the current answer to Q in $\mathbf{D}^e$ is $Q(\mathbf{D}_1) \cap Q(\mathbf{D}_2) = (c) \cap (d)$ and thus empty. These tell us that ρ is not currency preserving for Q. Conversely, assume that φ is false. Then Q returns the empty set irrespectively of the current instances of I_b and extensions thereof. In other words, ρ is currency preserving for Q. □

Data complexity CPP *for CQ:* We show that $\mathsf{CPP}(\mathcal{L}_Q)$ is Π_2^p-hard by reduction from the $\forall^*\exists^*$3CNF problem which is known to be Π_2^p-complete [Stockmeyer 1976]. The $\forall^*\exists^*$3CNF problem is to decide, given a sentence $\varphi = \forall X \exists Y\, \psi(X, Y)$, whether φ is true. Here $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_m\}$ and ψ is an instance $C_1 \wedge \cdots \wedge C_r$ of 3SAT over $X \cup Y$. Given an instance $\forall X \exists Y\, \psi(X, Y)$, we define a specification $\mathbf{S}$ and a query Q in CQ, such that φ is true iff the copy functions $\bar{\rho}$ in $\mathbf{S}$ are currency preserving for Q.

(1) Temporal instances. The specification $\mathbf{S}$ consists of data sources $\mathbf{D}$ and $\mathbf{D}'$, where $\mathbf{D}$ consists of three relational schemas $R_{XY}(\mathsf{EID}, X, V)$, $R_C(\mathsf{EID}, , \mathsf{CID}, \mathsf{POS}, P, L, V, C)$ and $R_b(\mathsf{EID}, C)$; and $\mathbf{D}'$ consists of the schemas $R'_X(\mathsf{EID}, X, V)$ and $R'_b(\mathsf{EID}, C)$. The corresponding instances and initial partial currency orders on them are as shown in

I_{XY} =

EID	X	V
1	x_1	0
1	x_1	1
2	x_2	0
2	x_2	1
...	...	...
n	x_n	0
n	x_n	1
$n+1$	y_1	0
$n+1$	y_1	1
$n+2$	y_2	0
$n+2$	y_2	1
...	...	...
$n+m$	y_m	0
$n+m$	y_m	1

I'_X =

	EID	X	V	
s_1	1	x_1	0	$s_1 \prec_V s_2$
s_2	1	x_1	1	
s'_1	$n+1$	x_1	0	$s'_2 \prec_V s'_1$
s'_2	$n+1$	x_1	1	
	...	...	...	
s_{2n-1}	n	x_n	0	$s_{2n-1} \prec_V s_{2n}$
s_{2n}	n	x_n	1	
s'_{2n-1}	$2n+1$	x_n	0	$s'_{2n} \prec_V s'_{2n-1}$
s'_{2n}	$2n+1$	x_n	1	

I_b =

EID	C
1	c
1	d

I'_b =

	EID	C
u_1	1	c
u_2	1	d

$u_2 \prec_C u_1$

Fig. 5. Temporal instances used in lower bound proof of Theorem 5.1(3).

Fig. 5. Intuitively, I_{XY} is used to represent truth assignments for X and Y possibly selected by (non-empty) copy functions of signature $R_{XY}[X, V] \Leftarrow R'_X[X, V]$ that are consistent with the initial partial currency order on I_X, whereas I_b and I'_b are used to control the latest value of the C-attribute by means of the copy function $R_b[\mathsf{EID}, C] \Leftarrow R'_b[\mathsf{EID}, C]$. Furthermore, the instance I_C is used to encode negations of the clauses in ψ. That is, for each $j \in [1, r]$ and clause $C_j = \ell_1^j \vee \ell_2^j \vee \ell_3^j$, consider $\overline{C}_j = \bar{\ell}_1^j \wedge \bar{\ell}_2^j \wedge \bar{\ell}_3^j$ and denote by μ_j the unique truth assignment for variables in C_j that satisfies $\overline{C}_j$. Then, for each $j \in [1, r]$, we add the following three tuples to I_C:

$$(\mathsf{eid}, j, 1, z_1, v_1, c), \quad (\mathsf{eid}, j, 2, z_2, v_2, c), \quad (\mathsf{eid}, j, 3, z_3, v_3, c),$$

where $z_i = \bar{\ell}_i^j$ if $\bar{\ell}_i^j$ is x_k or y_k, and $z_i = \ell_i^j$ otherwise. Furthermore, v_i is $\mu_j(\bar{\ell}_i^j)$ if $\bar{\ell}_i^j$ is x_k or y_k, and it is $\overline{\mu_j(\ell_i^j)}$ otherwise. Here eid is unique for each tuple.

(2) Copy functions and denial constraints. We define empty copy functions $\rho_1 : R_{XY}[X, V] \Leftarrow R'_X[X, V]$ and $\rho_2 : R_b[C] \Leftarrow R'_b[C]$. Let $\bar{\rho} = \{\rho_1, \rho_2\}$. Furthermore, we constrain the possible extensions of $\bar{\rho}$ by enforcing that each entity in all instances has only two possible tuples. These can be expressed as *fixed* denial constraints.

(3) Query. The query CQ Q is defined as follows (omitting the EID attributes):

$$Q = \exists z_1, z_2, z_3, v_1, v_2, v_3, j, w \big(R_{XY}(z_1, v_1) \wedge R_{XY}(z_2, v_2) \wedge R_{XY}(z_3, v_3) \wedge R_C(j, 1, z_1, v_1, w) \wedge R_C(j, 2, z_2, v_2, w) \wedge R_C(j, 3, z_3, v_3, w) \wedge R_b(w)\big).$$

For each $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$, $\mathsf{LST}(I^c_{XY})$ corresponds to truth assignments μ_X and μ_Y for X and Y, respectively. Here I^c_{XY} is the completion of I_{XY} in $\mathbf{D}^c$. Furthermore, $Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ returns a non-empty set iff for μ_X and μ_Y, at least one conjunctive clause $\overline{C}_j$ is satisfied, and *in addition*, $\mathsf{LST}(I^c_b) = \{(1, c)\}$.

Now consider extensions $\bar{\rho}^e$ of $\bar{\rho}$ and denote by $\mathbf{S}^e$ the corresponding specification. Observe that due to the denial constraints, no new tuples can be added to any of the instances. Indeed, any such new tuple would cause $\mathsf{Mod}(\mathbf{S}^e)$ to be empty and such extensions are not taken into account by the definition of currency preservation. Clearly, every extension ρ_1^e of ρ_1 limits the set of possible truth assignments for X that are realized as $\mathsf{LST}(I^c_{XY})$, for completions of $\mathbf{S}^e$. Indeed, copy functions from R'_X to R_{XY} have specific choices for the variables in X, due to the partial orders present in I'_X, as argued in the proof of the combined complexity of CPP (CQ) given above. Furthermore, if $\bar{\rho}^e$ extends ρ_2, then $\mathsf{LST}(I^c_b) = \{(1, c)\}$; otherwise, it can be either

$\{(1, c)\}$ or $\{(1, d)\}$ depending on how it is completed. Observe that $\mathsf{LST}(I_b^c) = \{(1, d)\}$ makes Q empty, irrespective of the other relations. Indeed, this is because none of the tuples in I_C have d as the last attribute value.

Let $T_X(\rho_1^e)$ be the set of truth assignments for X that are witnessed by completions of $\mathbf{S}^e$. Note that $T_X(\rho_1)$ consists of all possible truth assignments since the copy function ρ_1 is empty. Then $\bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S}^e)} Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ is non-empty if for all $\mu_X \in T_X(\rho_1^e)$, all truth assignments μ_Y make at least one conjunctive clause $\overline{C}_j$ true and moreover, ρ_2^e expands ρ_2, *i.e.,* $\mathsf{LST}(I_b^c) = \{(1, c)\}$.

We next show that φ is true iff $\bar{\rho}$ in $\mathbf{S}$ is currency preserving for Q.

$\boxed{\Rightarrow}$ Assume that φ is satisfied. The certain current answers of Q on $\mathbf{S}$ will return empty. This is simply because $\mathsf{LST}(I_b^c) = \{(1, d)\}$ is realized in a completion. We know, however, that for every μ_X, there exists a μ_Y such that ψ evaluates to true. This is in particular true for any subset of truth assignments of $T_X(\rho_1^e)$ of X for extensions $\mathbf{S}^e$ of $\mathbf{S}$. As a consequence, the certain current answers of Q on $\mathbf{S}^e$ will be empty, even when $\mathsf{LST}(I_b^c) = \{(1, c)\}$. In other words, $\bar{\rho}$ is currency preserving.

$\boxed{\Leftarrow}$ Assume that φ is not satisfied. Observe that again, the current answers of Q on $\mathbf{S}$ will return empty. This is because $\mathsf{LST}(I_b^c) = \{(1, d)\}$ is realized in a completion. It can be easily verified that $\bar{\rho}$ is not currency preserving. Indeed, since φ is not satisfied there must exist a μ_X such that $\exists Y\, \psi(\mu_X, Y)$ is false. By extending $\bar{\rho}$ to $\bar{\rho}^e = \{\rho_1^e, \rho_2^e\}$ such that $T_X(\rho_1^e) = \{\mu_X\}$ and by extending ρ_2 such that the only current instance of I_b is $\mathsf{LST}(I_b^c) = \{(1, c)\}$, we obtain that the certain current answers for $\mathbf{S}^e$ is non-empty. Hence, $\bar{\rho}$ is not currency preserving. □

Upper bounds CPP*:* We next provide upper bounds for $\mathsf{CPP}(\mathcal{L}_Q)$. We develop a decision algorithm that takes a specification $\mathbf{S}$ and query $Q \in \mathcal{L}_Q$ as input, and returns "yes" if the copy functions $\bar{\rho}$ in $\mathbf{S}$ are not currency preserving for Q.

The algorithm is as follows. As an initial step, it checks whether $\mathbf{S}$ is inconsistent. If so, then $\bar{\rho}$ is not currency preserving by the definition of currency preservation, and hence the algorithm returns "yes". Otherwise, if $\mathbf{S}$ is consistent, the algorithm proceeds as follows. We denote by $adom(\mathbf{S}, Q)$ all constants appearing in any of the tuples in the temporal instances and denial constraints in $\mathbf{S}$ together with all occurring constants in Q.

(1) Guess a pair $(\bar{t}, \bar{\rho}^e)$, where $\bar{t}$ is a tuple of the result schema of Q and with values taken from $adom(\mathbf{S}, Q)$, and $\bar{\rho}^e$ is a candidate extension of $\bar{\rho}$.
(2) Verify whether $\bar{\rho}^e$ is an element of $\mathsf{Ext}(\bar{\rho})$. If so, let $\mathbf{S}^e$ be the corresponding extended specification of $\mathbf{S}$ by $\bar{\rho}^e$. Verify whether $\mathsf{Mod}(\mathbf{S}^e) \neq \emptyset$. If not, reject the current guess.
(3) Verify whether $\bar{t} \in Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ for every $\mathbf{D}^c \in \mathsf{Mod}(S)$.
   (a) If so, check whether there exists a $\mathbf{D}^e \in \mathsf{Mod}(\mathbf{S}^e)$ such that $\bar{t} \notin Q\big(\mathsf{LST}(\mathbf{D}^e)\big)$. If not, reject the current guess, otherwise return "yes".
   (b) If not, check whether for every $\mathbf{D}^e \in \mathsf{Mod}(\mathbf{S}^e)$, $\bar{t} \in Q\big(\mathsf{LST}(\mathbf{D}^e)\big)$. If not, reject the current guess, otherwise return "yes".

Based on the algorithm, we present an analysis of the complexity of $\mathsf{CPP}(\mathcal{L}_Q)$ as follows. We start with the combined complexity. Observe that initial step and step 2 can be done in Π_2^p by Theorem 3.1. When $\mathcal{L}_Q$ is $\exists$FO$^+$, we know from Theorem 3.5 that steps 3(a) and 3(b) can be done in Π_2^p and Σ_2^p, respectively. Hence, the overall complexity of the algorithm is $\Sigma_3^p = \mathrm{NP}^{\Pi_2^p}$. Consequently, $\mathsf{CPP}(\mathcal{L}_Q)$ is in Π_3^p for $\exists$FO$^+$. Similarly, when $\mathcal{L}_Q$ is FO, Theorem 3.5 tells us that steps 3(a) and 3(b) can be done in PSPACE. As a result, $\mathsf{CPP}(\mathcal{L}_Q)$ is in PSPACE for FO.

When data complexity is concerned, again from Theorems 3.1 and 3.5 it follows that

the initial step can be done in coNP, steps 2, 3(a) and 3(b) can be done in NP, coNP and NP, respectively, even when $\mathcal{L}_Q$ is FO. Therefore, the data complexity of the algorithm is $\Sigma_2^p = \text{NP}^{\text{NP}}$. Hence $\mathsf{CPP}(\mathcal{L}_Q)$ is in Π_2^p. □

The feasibility of currency preservation. We next consider $\mathsf{ECP}(\mathcal{L}_Q)$ to decide, given a query Q and a consistent specification $\mathbf{S}$ in which copy functions $\bar{\rho}$ are not currency preserving for Q, whether we can extend $\bar{\rho}$ to preserve currency. The good news is that the answer to this question is affirmative: we can always extend $\bar{\rho}$ and make them currency preserving for Q. Hence the decision problem ECP is in $O(1)$ time, although it may take much longer to explicitly construct a currency preserving extension of $\bar{\rho}$.

When $\mathbf{S}$ is not necessarily consistent, it is easy to verify that it is Σ_2^p-complete to decide whether $\bar{\rho}$ can be made currency preserving for Q. Indeed, this problem is equivalent to CPS, since $\bar{\rho}$ can be made currency preserving for Q iff $\mathbf{S}$ is consistent. From Theorem 3.1 it follows that the combined complexity of this problem is Σ_2^p-complete, and its data complexity is NP-complete.

PROPOSITION 5.2. $\mathsf{ECP}(\mathcal{L}_Q)$ *is decidable in* $O(1)$ *time for both the combined complexity and data complexity, when* $\mathcal{L}_Q$ *is CQ, UCQ,* $\exists$*FO*$^+$ *or FO.*

PROOF. Consider data sources $\mathbf{D} = (D_1, \ldots, D_p)$ and $\mathbf{D}' = (D'_1, \ldots, D'_q)$, with (1) a set Σ_i (resp. Σ'_j) of denial constraints on D_i for each $i \in [1, p]$ (resp. D'_j for $j \in [1, q]$), and (2) a collection $\overline{\rho}$ of copy functions $\rho_{(j,i)}$ that import tuples from D'_j to D_i, for $i \in [1, p]$ and $j \in [1, q]$. We say that an extension $\rho^e_{(j,i)}$ of $\rho_{(j,i)}$ is *maximum* if either (a) no more tuples from D'_j can be copied to D_i, or (b) adding any new tuple from D'_j to D_i makes the modification $\mathbf{S}^e$ of $\mathbf{S}$ inconsistent, *i.e.,* $\mathsf{Mod}(\mathbf{S}^e) = \emptyset$. In other words, there exists no extension $\rho'_{(j,i)}$ of $\rho^e_{(j,i)}$ such that $\rho'_{(j,i)} \neq \rho^e_{(j,i)}$ and it makes a consistent specification.

We show that for each $\rho_{(j,i)}$, we can find a maximum extension $\rho^e_{(j,i)}$ of $\rho_{(j,i)}$. Indeed, we simply extend $\rho_{(j,i)}$ by considering tuples t in D'_j one by one. If the extension of $\rho_{(j,i)}$ with tuple t makes the modified specification inconsistent, we do not copy t and consider the next tuple (according to some arbitrary order) in D'_j. We repeat the process until all tuples in D'_j are checked. This yields an extension $\rho^e_{(j,i)}$. Obviously $\rho^e_{(j,i)}$ is maximum. We extend $\rho_{(j,i)}$ in this way for all $i \in [1, p]$ and $j \in [1, q]$. Putting these $\rho^e_{(j,i)}$'s together, we get an extension $\overline{\rho}^e$ of $\overline{\rho}$.

We show that $\overline{\rho}^e$ is currency preserving. Indeed, by the construction of $\overline{\rho}^e$, we have that the specification $\mathbf{S}^e$ derived from $\overline{\rho}^e$ and $\mathbf{S}$ is consistent, *i.e.,* $\mathsf{Mod}(\mathbf{S}^e) \neq \emptyset$. Furthermore, $\mathsf{Ext}(\overline{\rho}^e) = \emptyset$, *i.e.,* $\overline{\rho}^e$ cannot possibly be further extended. From the definition of currency preservation it follows that $\overline{\rho}^e$ is currency preserving for Q, no matter whether Q is in CQ, UCQ, $\exists$FO$^+$ or FO. □

Bounded extensions. In contrast to ECP, when it comes to deciding whether $\bar{\rho}$ can be made currency-preserving by copying data within a bounded size, the analysis becomes far more intricate. Indeed, the result below tells us that even for CQ, BCP is Σ_4^p-hard, and fixing denial constraints and copy functions does not help. When both queries and denial constraints are fixed, BCP is Σ_3^p-complete.

THEOREM 5.3. *For* $\mathsf{BCP}(\mathcal{L}_Q)$*, the combined complexity is*

(*1*) Σ_4^p*-complete when* $\mathcal{L}_Q$ *is CQ, UCQ or* $\exists$*FO*$^+$*, and*
(*2*) *PSPACE-complete when* $\mathcal{L}_Q$ *is FO.*
(*3*) *Its data complexity is* Σ_3^p*-complete when* $\mathcal{L}_Q \in \{$*CQ,UCQ,* $\exists$*FO*$^+$*,FO*$\}$.

$$I_W = \begin{array}{|cc|}\hline \mathsf{EID} & W \\ \hline 1 & \bot \\ \dots & \dots \\ p & \bot \\ \hline \end{array}, \quad I'_W = \begin{array}{|cc|}\hline \mathsf{EID} & W \\ \hline 1 & 1 \\ 1 & 0 \\ \dots & \dots \\ p & 1 \\ p & 0 \\ \hline \end{array}$$

Fig. 6. Temporal instances used in lower bound proof of Theorem 5.3(1)

The combined complexity bounds remain unchanged when denial constraints and copy functions are fixed.

PROOF. We show that $\mathsf{BCP}(\mathcal{L}_Q)$ is (a) Σ_4^p-hard (combined complexity) and Σ_3^p-hard (data complexity) when Q is in CQ, (b) PSPACE-hard (combined complexity) when Q is in FO, (c) it is in Σ_4^p (combined complexity) when Q is in $\exists$FO$^+$, and (d) in PSPACE (combined complexity) and in Σ_3^p (data complexity) when Q is in FO.

Combined complexity BCP *for CQ:* We show that BCP(CQ) is Σ_4^p-hard by reduction from the $\exists^*\forall^*\exists^*\forall^*$3DNF problem, which is known to be Σ_4^p-complete [Stockmeyer 1976]. An instance of the $\exists^*\forall^*\exists^*\forall^*$3DNF problem is a sentence $\varphi = \exists W \forall X \exists Y \forall Z \psi(W, X, Y, Z)$, where $W = \{w_l \mid l \in [1, p]\}$, $X = \{x_i \mid i \in [1, n]\}$, $Y = \{y_j \mid j \in [1, m]\}$, $Y = \{z_s \mid s \in [1, q]\}$, and ψ is of the form $C_1 \vee \cdots \vee C_r$. Furthermore, for each $i \in [1, r]$, C_i is a conjunction of three literals (variables or negated variables) taken from $W \cup X \cup Y \cup Z$. Given φ, the $\exists^*\forall^*\exists^*\forall^*$3DNF problem is to determine whether φ is true.

Given φ, we define a consistent specification $\mathbf{S}$ with a collection $\overline{\rho}$ of copy functions, a query Q and a positive number k. We show that φ is true iff there exists an extension $\overline{\rho}^e$ of ρ such that $\overline{\rho}^e$ is currency preserving for Q and $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. We define k to be $p(\log(p)+1)$ bits, where $|p|$ is the number of variables in W. As will be seen shortly, our copy functions import truth assignments for variables in W, and it takes $p(\log(p) + 1)$ bits to code such an assignment. Below we give $\mathbf{S}$ and Q, with fixed denial constraints.

(1) Temporal instances. The specification $\mathbf{S}$ includes data sources $\mathbf{D}$ and $\mathbf{D}'$, where $\mathbf{D}$ consists of nine relations: $R_{01}(\mathsf{EID}, A)$, $R_W(\mathsf{EID}, W)$, $R_X(\mathsf{EID}, X, V, K)$, $R_Y(\mathsf{EID}, Y, V)$, $R_\vee(\mathsf{EID}, B, A_1, A_2)$, $R_\wedge(\mathsf{EID}, B, A_1, A_2)$, $R_\neg(\mathsf{EID}, A, \bar{A})$, $R_{ca}(\mathsf{EID}, A_1, A_2)$ and $R_b(\mathsf{EID}, C)$, and $\mathbf{D}'$ consists of three schemas: $R'_W(\mathsf{EID}, W)$, $R'_X(\mathsf{EID}, X, V, K)$ and $R'_b(\mathsf{EID}, C)$. The instances I_{01}, $I_\wedge, I_\vee$ and $I_\neg$ are the same as their counterparts given in the proof of Theorem 3.5(1), to encode Boolean domain, conjunction, disjunction and negation, respectively (see Fig. 2). The instance I_{ca} consists of two tuples $(\mathsf{eid}, 0, c)$ and $(\mathsf{eid}', 1, a)$ and is used to convert 0 to c and 1 to a constant a distinct from c and d. Here eid and eid' denote two new distinct identifiers. The instances I_X, I'_X and I_Y represent truth assignments for variables in X and Y, which are the same as their counterparts shown in Fig. 4, except that tuples in I_X and I'_X carry an extra attribute K with a constant value of $k+1$ bits. The instances I_W and I'_W are given in Fig. 6. Intuitively, I'_W consists of $2p$ tuples and is used to encode truth assignments for variables in W. The instance I_W has p tuples $(j, \bot)$, indicating p entities for which values will be copied from I'_W, where $\bot$ denotes a value different from 0 and 1. In addition, we use the same I_b and I'_b as shown in Fig. 4, to check whether an extension $\overline{\rho}^e$ is currency preserving for query Q (given below). We assume that each of c and d in I_b and I'_b is a constant of $k+1$ bits. In none of these instances, except I'_X and I'_b, is an initial partial currency order defined.

(2) Denial constraints. For I_W, we define the following constraints: (a) φ_1, asserting that for each $i \in [1, p]$, there exist at most two tuples with the same EID attribute; and (b) φ_2, assuring that for each $i \in [1, p]$, if there exist $t_1 = (i, \bot)$ and $t_2 = (i, x)$ with $x = 0$ or $x = 1$, then $t_1 \prec_W t_2$, *i.e.,* $0/1$-values (copied from I'_W) are more current that the $\bot$-value, and hence will be chosen as the truth value of w_i in $\mathsf{LST}(I^e_W)$, where

I^e_W denotes the extension of I_W by copying new values from I'_W. Obviously these can be expressed as denial constraints. In addition, we use the same set of (fixed) denial constraints described in the proof of Theorem 5.1(1), to ensure that truth assignments in I_X selected after copying from I'_X are valid.

(3) Copy functions. Three fixed copy functions are defined in $\overline{\rho}$: (a) $\rho_W : R_W[\mathsf{EID}, W] \Leftarrow R'_W[\mathsf{EID}, W]$ imports values from I'_W to I_W, (b) $\rho_X : R_X[X, V, K] \Leftarrow R'_X[X, V, K]$ copies values from I'_X to I_X, and (c) $\rho_b : R_b[\mathsf{EID}, C] \Leftarrow R'_b[\mathsf{EID}, C]$ imports values from I'_b to I_b. All these copy functions are initially empty. Let $\rho = \{\rho_W, \rho_X, \rho_b\}$.

(4) Query. We define Q in CQ as follows (omitting the EID attributes):

$$Q(v) = \exists \vec{w}\, \exists \vec{x}\, \exists \vec{y}\, \exists \vec{z}\, \big(Q_W(\vec{w}) \wedge Q_X(\vec{x}) \wedge Q_Y(\vec{y}) \wedge Q_Z(\vec{z}) \ \wedge \ Q_\psi(\vec{w}, \vec{x}, \vec{y}, \vec{z}, v) \wedge R_b(v)\big).$$

Here $Q_W(\vec{w})$ is $\bigwedge_{i\in[1,p]} R_W(i, w_i)$, where $w_i \in W$. It extracts from R_W a truth assignment μ_W for variables in W. Similarly, $Q_X(\vec{x})$ and $Q_Y(\vec{y})$ extract a truth assignment for variables of X and Y from R_X and R_Y, respectively; and $Q_Z(\vec{z})$ generates all truth assignments for variables of Z using R_{01}. Along the same lines as the proof of Theorem 5.1(1), $Q_\psi(\vec{w}, \vec{x}, \vec{y}, \vec{z}, v)$ is a CQ query that encodes the truth value of $\neg\psi(W, X, Y, Z)$ for a given truth assignment μ_W for W, μ_X for X, μ_Y for Y and μ_Z for Z, such that $v = c$ if ψ is *not* satisfied by μ_W, μ_X, μ_Y and μ_Z, and a distinct value a from c and d otherwise. The query Q_ψ can be expressed in CQ in terms of $R_\vee$, $R_\wedge$, $R_\neg$ and R_{ca}.

One can readily verify that $\mathbf{S}$ is consistent, *i.e.,* $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$. In addition, the schemas and denial constraints are fixed, *i.e.,* they are independent of φ.

We next show that φ is true iff there exists an extension $\overline{\rho}^e$ of ρ such that $\overline{\rho}^e$ is currency preserving for Q and $|\overline{\rho}^e| \leq |\overline{\rho}| + k$.

$\boxed{\Rightarrow}$ Suppose that φ is true. Then there exists a truth assignment μ_W for variables in W such that $\forall X \exists Y \forall Z \psi(\mu_W, X, Y, Z)$ is true. We define an extension $\overline{\rho}^e$ of $\overline{\rho}$, where ρ^e_W in $\overline{\rho}^e$ extends ρ_W by copying $(i, 1)$ from I'_W to I_W if $\mu_W(w_i) = 1$, and copying $(i, 0)$ if $\mu_W(w_i) = 0$, for each $i \in [1, p]$. The other copy functions of $\overline{\rho}$ remain unchanged in $\overline{\rho}^e$. Obviously $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. We show that $\overline{\rho}^e$ is currency preserving for Q. Let I^e_W denote the extension of I_W by ρ^e_W, and $\mathbf{S}^e$ denote the extension of $\mathbf{S}$ with ρ^e_W. Observe the following. (a) By the denial constraint φ_2 defined on R_W, this truth assignment is selected in $\mathsf{LST}(I^e_W)$, no matter what consistent completion of I^e_W is considered. In other words, the truth assignment for W remains fixed in all consistent completions of I^e_W. (b) Along the same lines as the proof of Theorem 5.1(1), one can verify that the certain current answer to Q *w.r.t.* $\mathbf{S}^e$ is empty. Indeed, when $\mathsf{LST}(I^c_b)$ is $\{(1, d)\}$, the answer to Q in such a completion of $\mathbf{S}^e$ is empty, and hence so is the certain current query answer of Q *w.r.t.* $\mathbf{S}^e$. (c) No matter how $\overline{\rho}^e$ is further extended, the certain current answer to Q remains empty. Indeed, observe that in any completion of an extension of $\mathbf{S}^e$, the answer to Q is nonempty only if for the truth assignments μ_W for W, μ_X for X and μ_Y for Y encoded in $\mathsf{LST}(I^c_W)$, $\mathsf{LST}(I^c_X)$ and $\mathsf{LST}(I^c_Y)$, respectively, there exists a truth assignment μ_Z for Z such that ψ is *not* satisfied. Since φ is true, given the truth assignment μ_W encoded by $\mathsf{LST}(I^c_W)$ in $\mathbf{S}^e$, for any truth assignment μ_X for X that may be copied from I'_X by extending ρ_1, there exists a completion of I_Y that encodes a truth assignment μ_Y for Y, such that for all truth assignment μ_Z for Z, ψ is satisfied; hence, the answer to Q is empty in this completion. In other words, the certain current answer to Q is empty *w.r.t.* any extension of S^e that is obtained by extending $\overline{\rho}^e$. Therefore, we can conclude that $\overline{\rho}^e$ is currency preserving for Q.

$\boxed{\Leftarrow}$ Conversely, suppose that φ is false. Assume for a contradiction that there exists an extension $\overline{\rho}^e$ of ρ such that $\overline{\rho}^e$ is currency preserving for Q and moreover, $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. Then by the choice of the c, d values in I'_b and the K value in I'_X, $\overline{\rho}^e$ can extend neither ρ_X nor ρ_b by copying data from I'_X or I'_b. As a result, the certain current answer to

Q *w.r.t.* $\mathbf{S}^e$ is empty, where $\mathbf{S}^e$ denotes the extension of $\mathbf{S}$ with $\overline{\rho}^e$. This is because when $\mathsf{LST}(I_b^c)$ is $\{(1, d)\}$, the answer to Q in such a completion of $\mathbf{S}^e$ is empty. Now consider an extension $\overline{\rho}^+$ of $\overline{\rho}^e$, which extends (a) ρ_W such that $\mathsf{LST}(I_W^c)$ encodes a truth assignment μ_W for variables in W; (b) ρ_b such that $\mathsf{LST}(I_b^c)$ is $\{(1, c)\}$, and (c) it extends ρ_X to produce $\mathsf{LST}(I_X^c)$ that encodes a truth assignment μ_X for variables in X, such that for any truth assignment μ_Y for Y, there exists a truth assignment μ_Z for Z that satisfies $\neg\psi$ together with μ_W, μ_X and μ_Y. This is possible because φ is false. Let S^+ be the extension of $\mathbf{S}^e$ with $\overline{\rho}^+$. Then the certain current answers to Q *w.r.t.* S^+ is nonempty. Thus $\overline{\rho}^e$ is not currency preserving, contradicting the assumption above. □

Combined complexity BCP *for FO:* We next show that BCP(FO) is PSPACE-hard even when k is *fixed*. We prove this by reduction from the complement of Q3SAT. We refer to the proof of Theorem 3.5(2) for the statement of the Q3SAT problem.

Given an instance φ of Q3SAT, we construct a specification $\mathbf{S}$ with copy functions $\bar{\rho}$ and a query Q. We show that φ is false iff there exists a currency preserving extension $\overline{\rho}^e$ of $\bar{\rho}$ such that $|\overline{\rho}^e| \leq |\bar{\rho}| + k$.

(1) Temporal instances. The specification $\mathbf{S}$ consists of two data sources $\mathbf{D}$ and $\mathbf{D}'$, where $\mathbf{D}$ consists of two relation schemas $R_{01}(\mathsf{EID}, A)$ and $R_b(\mathsf{EID}, B)$. The corresponding instances are $I_{01} = \{(1, 0), (2, \bot)\}$ and $I_b = \{(1, v_a)\}$, where $\bot$ is a constant different from 1, and v_a is an arbitrary constant. Furthermore, $\mathbf{D}'$ consists of the same relations $R'_{01}(\mathsf{EID}, A)$ and $R'_b(\mathsf{EID}, B)$ and the corresponding instances are $I'_{01} = \{(1, 0), (2, 1)\}$ and $I'_b = \{(1, v_a), (1, v_b)\}$. An initial partial currency order $(1, v_a) \prec_B (1, v_b)$ is specified on I'_b. We assume that $2 \leq k$ and that v_b has $k + 1$ bits.

(2) Copy functions. The specification $\mathbf{S}$ contains two copy functions $\bar{\rho} = \{\rho_1, \rho_b\}$ of signature $R_{01}[\mathsf{EID}, A] \Leftarrow R'_{01}[\mathsf{EID}, A]$ and $R_b[\mathsf{EID}, B] \Leftarrow R'_b[\mathsf{EID}, B]$, respectively. The copy functions are defined as $\rho_1\big((1, 0)\big) = (1, 0)$ and $\rho_b\big((1, v_a)\big) = (1, v_a)$.

(3) Denial constraints. We use a denial constraint on R_{01} to assure that for any tuples $t_1 = (\mathsf{eid}, \bot)$ and $t_2 = (\mathsf{eid}, 1)$ in I_{01}, $t_1 \prec_A t_2$.

(4) Query. We use an FO query Q, similar to the one given in the proof of Theorem 5.1(2). That is, for $\varphi = P_1 X_1 \cdots P_m X_m \psi$ we define (omitting EID attributes):

$$Q(c) = P_1\vec{x}_1, \ldots, P_m\vec{x}_m \ \big(\big(Q_{01} \wedge Q_{X_1}(\vec{x}_1) \wedge \cdots \wedge Q_{X_m}(\vec{x}_m) \wedge Q_\psi(\vec{x}_1, \ldots, \vec{x}_m) \wedge R_b(c)\big) \vee (R_{01}(2, c) \wedge c = \bot)\big),$$

where $Q_{01} = R_{01}(1, 0) \wedge R_{01}(2, 1)$ and Q_{X_i} leverages R_{01} to generate all truth assignments for X_i. The query Q_ψ is the same as ψ. Note that k is fixed, *i.e.,* it is independent of φ. Observe that $\bar{\rho}$ is not currency preserving since the certain answer of Q *w.r.t.* $\mathbf{S}$ is $\{(\bot)\}$ but this tuple is removed from the certain answer when ρ_1 is extended to ρ'_1 which copies $(2, 1)$ from I'_{01} into I_{01}.

We show that the coding is a reduction from the complement of Q3SAT to BCP(FO).

$\boxed{\Leftarrow}$ First assume that φ is false. Then ρ can be extended into a currency preserving copy function by letting $\bar{\rho}^e = \{\rho'_1, \rho_b\}$, where ρ'_1 is as previously defined. It is readily verified that $\bar{\rho}^e$ is a currency preserving copy function, since the answer to Q is empty, no matter how the copy function is extended.

$\boxed{\Rightarrow}$ Conversely, assume that φ is true. By the choice of v_b, the only possible extensions $\overline{\rho}^e$ of ρ such that $|\overline{\rho}^e| \leq |\overline{\rho}| + k$ involve copying from I'_{01} to I_{01}. However, such extensions are not currency preserving when φ is true. Indeed, there is an extension ρ' of ρ that also imports $(1, v_b)$ from I'_b to I_b, such that the certain answer to Q *w.r.t.* $\mathbf{S}$ is $\{(v_a)\}$, while the certain answer to Q *w.r.t.* $\mathbf{S}'$ is $\{(v_b)\}$. Here $\mathbf{S}'$ is the extension of $\mathbf{S}$ by ρ'. Thus there does not exist a currency preserving extension $\overline{\rho}^e$ of ρ such that $|\overline{\rho}^e| \leq |\rho| + k$. □

Data complexity BCP *for CQ:* We next show that BCP(CQ) is Σ_3^p-hard when query Q and denial constraints are fixed (for data complexity). We prove the lower bound by reduction from the $\exists^*\forall^*\exists^*$3CNF problem (see the proof of Theorem 5.1 for the statement of the $\exists^*\forall^*\exists^*$3CNF problem).

Given an instance $\varphi = \exists X \forall Y \exists Z \psi(X, Y, Z)$ of the $\exists^*\forall^*\exists^*$3CNF problem, we define a specification $\mathbf{S}$ that includes a collection $\bar{\rho}$ of copy functions, a positive number k and a fixed query Q, *i.e.,* Q does not depend on φ. We show that φ is true iff there exists an extension $\bar{\rho}^e$ of ρ such that $\bar{\rho}^e$ is currency preserving for Q and $|\bar{\rho}^e| \leq |\bar{\rho}| + k$. We define k to be $2n(2\log(n) + 2)$ bits, where n is the number of variables in X, and construct $\mathbf{S}$ and Q as follows.

(1) Temporal instances. The specification $\mathbf{S}$ consists of two data sources $\mathbf{D}$ and $\mathbf{D}'$, where $\mathbf{D}$ consists of three relations specified by schemas $R_{XYZ}(\mathsf{EID}, X, V, B)$, $R_C(\mathsf{EID}, \mathsf{CID}, \mathsf{POS}, L, V, C)$ and $R_b(\mathsf{EID}, C)$, and $\mathbf{D}'$ consists of three relations specified by $R'_X(\mathsf{EID}, X, V, B)$, $R'_Y(\mathsf{EID}, Y, V, B)$ and $R'_b(\mathsf{EID}, C)$. Here the instance I_{XYZ} of R_{XYZ} is used to represent truth assignments for X, Y and Z, which, as will be seen shortly, are constrained by copy functions from instances I'_X and I'_Y of R'_X and R'_Y, respectively. The instance I_{XYZ} is similar to I_{XY} shown in Fig. 5, except the following: (a) for each variable in $X \cup Y \cup Z$, it contains two tuples with the same EID but different V values 0 and 1; and (b) each tuple t carries an extra attribute B such that $t[B]$ is a constant 0 if $t[X]$ is a variable in X, and $t[B]$ is a constant K of $k+1$ bits if $t[X]$ is a variable in $Y \cup Z$. The presence of these tuples constrain the impact of extending copy functions in that they do not add new tuples but instead only copy available currency information. Instances I'_X and I'_Y are used to represent truth assignments for X and Y, respectively. They are similar to I'_X shown in Fig. 5, with partial currency orders defined on them, except that here each tuple s carries an extra attribute B with value 0 if s is in I'_X, and value K if s is in I'_Y. Since K has $k+1$ bits, this prevents truth assignments for Y to be copied when bounded copy functions are extended. Instances I_b and I'_b are precisely the same as their counterparts shown in Fig. 5, but we assume here that c and d are constants of at least $k+1$ bits. We shall use I_b and I'_b to check whether an extension of copy functions is currency preserving. As before, since c and d have $k+1$ bits, extensions of bounded copy functions cannot copy these constants.

Finally, the instance I_C of R_C is used to encode the negations of the clauses in ψ along the same lines as the proof of Theorem 5.1(3). More specifically, for each $j \in [1, r]$, consider the negation $\overline{C}_j = \bar{\ell}_1^j \wedge \bar{\ell}_2^j \wedge \bar{\ell}_3^j$ of clause C_j and denote by μ_j the unique truth assignment for variables in C_j that satisfies $\overline{C}_j$. Then, for each $j \in [1, r]$, we add the following three tuples to I_C:

$$(\mathsf{eid}, j, 1, z_1, v_1, c), \quad (\mathsf{eid}, j, 2, z_2, v_2, c), \quad (\mathsf{eid}, j, 3, z_3, v_3, c),$$

where $z_i = \bar{\ell}_i^j$ if $\bar{\ell}_i^j$ is a variable, and $z_i = \ell_i^j$ otherwise. Moreover, v_i is $\mu_j(\bar{\ell}_i^j)$ if $\bar{\ell}_i^j$ is a variable, and it is $\overline{\mu_j(\ell_i^j)}$ otherwise. Here eid is unique for each tuple. Except in I'_X, I'_Y and I'_b, partial currency orders are empty on these relations.

(2) Copy functions and denial constraints. We define three copy functions $\rho_X : R_{XYZ}[X, V, B] \Leftarrow R'_X[X, V, B]$, $\rho_Y : R_{XYZ}[X, V, B] \Leftarrow R'_Y[Y, V, B]$ and $\rho_b : R_b[\mathsf{EID}, C] \Leftarrow R'_b[\mathsf{EID}, C]$, each of which is initially empty. Let $\bar{\rho} = \{\rho_X, \rho_Y, \rho_b\}$. Furthermore, we constrain the possible extensions of $\bar{\rho}$ by enforcing that each entity in all instances has only two possible tuples, which can be expressed as *fixed* denial constraints.

(3) Query. We define the query Q in CQ as follows (omitting the EID attributes):

$$Q = \exists z_1, z_2, z_3, v_1, v_2, v_3, b_1, b_2, b_3, j, w \big(R_{XYZ}(z_1, v_1, b_1) \wedge R_{XYZ}(z_2, v_2, b_2) \wedge$$
$$R_{XYZ}(z_3, v_3, b_3) \wedge R_C(j, 1, z_1, v_1, w) \wedge R_C(j, 2, z_2, v_2, w) \wedge R_C(j, 3, z_3, v_3, w) \wedge R_b(w)\big).$$

This query is the same as its counterpart given in the proof of Theorem 5.1(3), except that we use R_{XYZ} here instead of R_{XY}. As remarked there, for each $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$, $\mathsf{LST}(I^c_{XYZ})$ encodes truth assignments μ_X, μ_Y and μ_Z for X, Y and Z, respectively, where I^c_{XYZ} is the completion of I_{XYZ} in $\mathbf{D}^c$. The query $Q\big(\mathsf{LST}(\mathbf{D}^c)\big)$ returns a non-empty set iff for μ_X, μ_Y and μ_Z, at least one conjunctive clause $\overline{C}_j$ is satisfied.

It is easy to verify that $\mathbf{S}$ is consistent, *i.e.,* $\mathsf{Mod}(\mathbf{S}) \neq \emptyset$. In addition, the query, schemas and denial constraints are all *fixed*, *i.e.,* they are independent of φ.

We next show that φ is true iff there exists an extension $\overline{\rho}^e$ of ρ such that $\overline{\rho}^e$ is currency preserving for Q and $|\overline{\rho}^e| \leq |\overline{\rho}| + k$.

$\boxed{\Rightarrow}$ Assume that φ is true. Then there exists a truth assignment μ_X for variables in X such that $\forall Y \exists Z \psi(\mu_X, Y, Z)$ is true. We define an extension $\overline{\rho}^e = \{\rho^e_X, \rho^e_Y, \rho^e_b\}$ of $\overline{\rho}$, where ρ^e_X extends ρ_X by copying $(i, x_i, 0, 0)$ and $(i, x_i, 1, 0)$ from I'_X to I_{XYZ} if $\mu_X(x_i) = 1$, and copying $(n+i, x_i, 0, 0)$ and $(n+i, x_i, 1, 0)$ if $\mu_X(x_i) = 0$, for each $i \in [1, n]$. The copy functions ρ^e_Y and ρ^e_b remain the same as ρ_Y and ρ_b, respectively. It is easy to see that $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. We next show that $\overline{\rho}^e$ is currency preserving for Q. Let I^e_{XYZ} denote the extension of I_{XYZ} by ρ^e_X, and $\mathbf{S}^e$ denote the extension of $\mathbf{S}$ with $\overline{\rho}^e$. Then as argued in the proof of Theorem 5.1(3), the certain current answer to Q is empty *w.r.t.* $\mathbf{S}^e$. Observe that for each consistent completion $(I^e_{XYZ})^c$ of I^e_{XYZ}, $\mathsf{LST}((I^e_{XYZ})^c)$ encodes (a) a fixed truth assignment μ_X for variables in X when $(I^e_{XYZ})^c$ is restricted to X variables, which remains unchanged in all consistent completions of I^e_{XYZ} because of the denial constraints defined on I_{XYZ}, and (b) a truth assignment μ_Y for Y and a truth assignment μ_Z for Z, which vary in different completions of I^e_{XYZ}. Since $\exists Z \psi(\mu_X, \mu_Y, Z)$ is true for all μ_Y of Y, no matter how ρ^e_Y and ρ^e_b are extended by copying more data values, there exists a completion of I_{XYZ} such that it encodes a truth assignment for Y and a truth assignment for Z, and moreover, $\neg\psi$ is not satisfied, *i.e.,* Q on the current instance of this completion is empty. That is, the certain current answer to Q is empty *w.r.t.* any extension of $\mathbf{S}^e$ that is obtained by extending $\overline{\rho}^e$. Hence $\overline{\rho}^e$ is currency preserving for Q.

$\boxed{\Leftarrow}$ Conversely, suppose that φ is false. Assume by contradiction that there exists an extension $\overline{\rho}^e$ of ρ such that $\overline{\rho}^e$ is currency preserving for Q and $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. Then by the choice of values c, d and K, $\overline{\rho}^e$ can neither copy truth assignments for Y from I'_Y nor extend I_b by copying from I'_b. In other words, $\overline{\rho}^e$ can only extend ρ_X. Let $\mathbf{S}^e$ denote the extension of $\mathbf{S}$ with $\overline{\rho}^e$. Then again as argued in the proof of Theorem 5.1(3), the certain current answer to Q is empty *w.r.t.* $\mathbf{S}^e$. We show that, however, there exists an extension of $\overline{\rho}^e$ such that the certain current answer to Q with that extension is nonempty. Indeed, since φ is false, no matter how $\overline{\rho}^e$ extends ρ_X, there exists a truth assignment μ_Y for Y such that for all truth assignments μ_Z for Z, $\neg\psi(\mu_X, \mu_Y, \mu_Z)$ is true. We define an extension $\overline{\rho}^+$ of $\overline{\rho}^e$ in which the extension of ρ_X encodes a truth assignment μ_X for X, the extension of ρ_Y encodes a truth assignment μ_Y for Y, such that for all completions of currency orders on Z variables (*i.e.,* μ_Z), $\neg\psi(\mu_X, \mu_Y, \mu_Z)$ is true. This is possible since φ is false. In addition, $\overline{\rho}^+$ extends ρ_b such that $\mathsf{LST}(I^c_b)$ is $\{(1, c)\}$. Let $\mathbf{S}^+$ denote the extension of $\mathbf{S}^e$ with $\overline{\rho}^+$. Then as argued in the proof of Theorem 5.1(3), the certain current answers to Q *w.r.t.* $\mathbf{S}^+$ is nonempty. Therefore, $\overline{\rho}^e$ is not currency preserving for Q. This contradicts the assumption above. □

Upper bounds BCP*:* We next verify the upper bounds by providing a non-deterministic algorithm that, given a consistent specification $\mathbf{S}$ with a collection $\overline{\rho}$ of copy functions, a query Q and a positive number k as input, returns "yes" if there exists a currency preserving extension $\overline{\rho}^e$ of $\overline{\rho}$ such that $|\overline{\rho}^e| \leq |\overline{\rho}| + k$. Let $\mathbf{D}$ and $\mathbf{D}'$ be the data sources in $\mathbf{S}$. The algorithm works as follows. The algorithm first checks whether $\mathbf{S}$ is consistent. If not, it returns "no". If $\mathbf{S}$ is consistent, it executes the following steps:

(1) Guess an extension $\overline{\rho}^e$ of $\overline{\rho}$ by copying additional data values of at most size k from $\mathbf{D}'$ to $\mathbf{D}$.
(2) Construct the extension $\mathbf{S}^e$ of $\mathbf{S}$ by $\overline{\rho}^e$.
(3) Check whether $\overline{\rho}^e$ is currency preserving for Q, by invoking the algorithm for CPP given in the proof of Theorem 5.1, with parameters $\mathbf{S}^e$ and Q. Return "yes" if that algorithm returns an affirmative answer. If not, reject the current guess and repeat the process.

The algorithm shows that the combined complexity of BCP($\exists$FO$^+$) is in Σ_4^p. Indeed, the initial step can be done in Σ_2^p by Theorem 3.1. Furthermore, step (2) is in PTIME and as shown in the proof of Theorem 5.1, the oracle for checking CPP($\exists$FO$^+$) is in Π_3^p for combined complexity. Thus the algorithm is in $\text{NP}^{\Pi_3^p} = \Sigma_4^p$ for the combined complexity when $\mathcal{L}_Q$ is $\exists$FO$^+$. When $\mathcal{L}_Q$ is FO, the oracle for checking CPP(FO) is in PSPACE for combined complexity, as verified in the proof of Theorem 5.1. Hence BCP(FO) is in PSPACE for combined complexity. Finally, for the data complexity, by Theorem 3.1 the initial step can be done in NP and by Theorem 5.1, the oracle for checking CPP is in Π_2^p. Therefore, the algorithm is in $\text{NP}^{\Pi_2^p} = \Sigma_3^p$ for the data complexity. □

6. TRACTABLE CASES

We next identify tractable cases for problems associated with reasoning about currency constraints (Section 3) and for problems related to copying (Section 4). More specifically, we show that all problems become tractable in the absence of denial constraints and, where appropriate, when the query language is restricted to SP queries as defined in Section 4. Observe that when no denial constraints are present, these tractable cases cover practical scenarios in which reliable timestamps are provided for part (or all) of the data. Indeed, such scenarios can be modeled without denial constraints but with initial currency orders in the temporal database instances.

As shown by Theorems 3.1 and 3.4, denial constraints make the analyses of CPS, COP and DCIP intricate. Below we consider specifications with no denial constraints, but containing partial currency orders and copy functions. The result below shows that the absence of denial constraints indeed simplifies the analyses.

THEOREM 6.1. *In the absence of denial constraints,* CPS*,* COP *and* DCIP *are in PTIME.* □

PROOF. We provide PTIME algorithms for each of these problems.

CPS*:* Let $\mathbf{S}$ be a specification consisting of (1) a collection of temporal instances $D_{(t,i)}$ of schema R_i, for $i \in [1, s]$; and (2) (possibly empty) copy functions $\rho_{(j,i)}$ from $D_{(t,i)}$ to $D_{(t,j)}$ of a certain signature $\sigma(\rho_{(j,i)}) : R_i[\vec{A}] \Leftarrow R_j[\vec{B}]$, where $\vec{A}$ consists of all attributes of R_i (except possibly the EID-attribute) and $\vec{B}$ consists of the same number of corresponding attributes in R_j. We provide a PTIME algorithm that decides whether $\mathbf{S}$ is consistent.

We use the following notations: For each $i \in [1, s]$ and $p \in [1, |R_i|]$, we denote by $\mathsf{PO}_{i,p}$ a binary relation on tuples in $D_{(t,i)}$ that is used to encode a partial order on them. Similarly, $\mathsf{PO}_{i,p}^{\mathsf{upd}}$ denotes an updated version of $\mathsf{PO}_{i,p}$. We always assume that these binary relations are transitively closed (possibly at a cost of a quadratic time computation). The algorithm performs the following steps:

(1) For $i \in [1, s]$, $p \in [1, |R_i|]$
$\quad\quad \mathsf{PO}_{i,p} := \{ (u, v) \mid D_{(t,i)}(u) \wedge D_{(t,i)}(v) \wedge u \prec_{(i,p)} v \}$;
(2) Set cycle := false and change := true;
(3) While (change = true) and (cycle = false) do

(a) For each $i \in [1, s]$, $j \in [1, s]$, $i \neq j$, $p \in [1, |R_i|]$, $q \in [1, |R_j|]$, and
for each $\rho_{(j,i)} : R_i[\vec{A}] \Leftarrow R_j[\vec{B}]$ such that A_p is copied from B_q, do
i. $\mathsf{PO}^{\mathsf{upd}}_{i,p} := \mathsf{PO}_{i,p} \cup \{\, (u, v) \notin \mathsf{PO}_{i,p} \mid \mathsf{PO}_{j,q}(\rho_{(j,i)}(u), \rho_{(j,i)}(v)) \,\}$;
ii. $\mathsf{PO}^{\mathsf{upd}}_{j,q} := \mathsf{PO}_{j,q} \cup \{\, (\rho_{(j,i)}(u), \rho_{(j,i)}(v)) \mid \mathsf{PO}_{i,p}(u, v) \,\}$;
(b) If there exist $i \in [1, s]$ and $p \in [1, |R_i|]$ such that $\mathsf{PO}^{\mathsf{upd}}_{i,p}$ contains a cycle then
i. Set $\mathsf{cycle} = \mathsf{true}$;
(c) Else if exist $i \in [1, s]$ and $p \in [1, |R_i|]$ such that $\mathsf{PO}_{i,p} \neq \mathsf{PO}^{\mathsf{upd}}_{i,p}$ then
i. Set $\mathsf{change} = \mathsf{true}$;
ii. Let $\mathsf{PO}_{i,p} = \mathsf{PO}^{\mathsf{upd}}_{i,p}$, for each $i \in [1, s]$, $p \in [1, |R_i|]$;
(d) Else set $\mathsf{change} = \mathsf{false}$;
(4) If $\mathsf{cycle} = \mathsf{false}$ then return "yes" otherwise return "no".

The algorithm starts by initializing $\mathsf{PO}_{i,p}$ with the initial partial currency order $\prec_{(i,p)}$ (Step 1). Then, as long as no cycles are detected in $\mathsf{PO}_{i,p}$ and $\mathsf{PO}^{\mathsf{upd}}_{i,p}$ differs from $\mathsf{PO}_{i,p}$, the algorithm updates $\mathsf{PO}_{i,p}$ with additional order information in Step 3. Here a cycle means that both (u, v) and (v, u) belong to $\mathsf{PO}_{i,p}$, for tuples u and v in $D_{(t,i)}$. There are two ways in which the algorithm adds order information using a copy function $\rho_{(j,i)}$: in Step 3(a)i, order information is transferred from $\mathsf{PO}_{j,q}$ to $\mathsf{PO}_{i,p}$; and in Step 3(a)ii, order information is transferred from $\mathsf{PO}_{i,p}$ to $\mathsf{PO}_{j,q}$.

We next show the correctness of the algorithm, *i.e.,* it returns "yes" iff $\mathbf{S}$ is consistent. We denote by $\mathsf{PO}^{\ell}_{i,p}$ the relation computed after ℓ iterations. Furthermore, if no cycles are detected and a fixed point has been reached (in Step 4), we denote by $\mathsf{PO}^{\infty}_{i,p}$ the final relation computed by the algorithm. If cycles were present, we let $\mathsf{PO}^{\infty}_{i,p} = \emptyset$.

$\boxed{\Rightarrow}$ Suppose that the algorithm returns "yes". Then for all $i \in [1, s]$ and $p \in [1, |R_i|]$ the relation $\mathsf{PO}^{\infty}_{i,p}$ corresponds to a partial order on tuples in $D_{(t,i)}$, which is compatible with the given partial order $\prec_{(i,p)}$. Indeed, by Step 1, $\prec_{(i,p)} \subseteq \mathsf{PO}^{\infty}_{i,p}$. Furthermore, Step 3 only adds information to $\mathsf{PO}_{i,p}$ and since the final relation $\mathsf{PO}^{\infty}_{i,p}$ is acyclic, no contradicting order information has been added. Hence, $\mathsf{PO}^{\infty}_{i,p}$ corresponds to a partial order (recall that we assume the relations are transitively closed) and contains $\prec_{(i,p)}$.

We next show how a consistent completion $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$ can be constructed from the partial orders $\mathsf{PO}^{\infty}_{i,p}$. Let $i \in [1, s]$, $p \in [1, |R_i|]$. Initially, we set $\prec^c_{(i,p)} = \mathsf{PO}^{\infty}_{i,p}$. Then, for any two tuples u_1, u_2 in $D_{(t,i)}$ that represent the same entity, we consider the following cases: (a) $\mathsf{PO}^{\infty}_{i,p}(u_1, u_2)$; (a') $\mathsf{PO}^{\infty}_{i,p}(u_2, u_1)$; and (b) u_1 and u_2 are incomparable under $\mathsf{PO}^{\infty}_{i,p}$. It suffices to treat only one of (a) or (a'), since those cases are symmetrical.

For case (a), we have already that $u_1 \prec^c_{(i,p)} u_2$. We show that this choice is not in conflict with the copy functions. Suppose that the A_p-attribute values in tuples u_1 and u_2 are respectively copied from the B_q-attribute values in v_1 and v_2 in an instance $D_{(t,j)}$. From $\mathsf{PO}^{\infty}_{(i,p)}(u_1, u_2)$ and Step 3(a)ii, it follows that $\mathsf{PO}^{\infty}_{j,q}(v_1, v_2)$. Since $\mathsf{PO}^{\infty}_{j,q}$ is acyclic, it is not the case that $\mathsf{PO}^{\infty}_{j,q}(v_2, v_1)$ is also reached.

For case (b), there exists a consistent completion in which $u_1 \prec^c_{(i,p)} u_2$, and another consistent completion in which $u_2 \prec^c_{(i,p)} u_1$. Indeed, we can choose either $u_1 \prec^c_{(i,p)} u_2$ or $u_2 \prec^c_{(i,p)} u_1$, and then propagate this choice to tuples that are copied to or from u_1 and u_2. Since Step 3(a) in the algorithm propagates currency orders, tuples that are copied to or from u_1 and u_2 cannot already be comparable themselves under some $\mathsf{PO}^{\infty}_{j,q}$ For instance, let ρ be such that $\rho(u_1) = v_1$ and $\rho(u_2) = v_2$, where ρ is a copy function

from R_j to R_i that copies B_q to A_p. Since u_1 and u_2 are incomparable under $\mathsf{PO}^{\infty}_{i,p}$, it follows from Step 3(a)i that v_1 and v_2 are incomparable under $\mathsf{PO}^{\infty}_{j,q}$. In other words, incomparable tuples can only be copied from incomparable tuples. We can therefore gather all pairs of tuples (like v_1 and v_2) in different instances (depending on the copy function) and select a consistent currency order for both u_1, u_2 and the collected pairs. For instance, we can choose $u_1 \prec_{(i,p)} u_2$ and $v_1 \prec_{(j,q)} v_2$, and transitively close this partial completion. We repeat this for any two such tuples u_1 and u_2 for which $\prec^c_{(i,p)}$ is still undefined. In this way, we obtain a consistent completion for $D_{(i,p)}$. We proceed in a similar way for all $i \in [1, s]$ and $p \in [1, |R_i|]$ and obtain a completion $\mathbf{D}^c \in \mathsf{Mod}(S)$.

$\boxed{\Leftarrow}$ Suppose that we have a consistent completion $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$. Denote by $\prec^c_{(i,p)}$ the completed currency orders in $\mathbf{D}^c$ for $D_{(t,i)}$ and attribute A_p. We prove by induction on increasing ℓ that for each ℓ, it is the case that $\mathsf{PO}^{\ell}_{i,p} \subseteq \{(u, v) \mid u, v \in D_{(t,i)}, u \prec^c_{(i,p)} v\}$. Clearly, after Step 1, we have $\mathsf{PO}^{0}_{i,p} = \prec_{(i,p)} \subseteq \prec^c_{(i,p)}$. For the induction step, assume that $\mathsf{PO}^{(\ell-1)}_{i,p} \subseteq \prec^c_{(i,p)}$ with $\ell \geq 1$. Assume by contradiction that $\mathsf{PO}^{\ell}_{i,p} \not\subseteq \prec^c_{(i,p)}$. This implies that either (a) a tuple (u, v) is added to $\mathsf{PO}^{\ell}_{i,p}$ in Step 3(a)i such that $v \prec^c_{(i,p)} u$; or (b) a tuple (u', v') is added to $\mathsf{PO}^{\ell}_{j,q}$ in Step 3(a)ii such that $v' \prec^c_{(j,q)} u'$. In case (a), we have $\mathsf{PO}^{(\ell-1)}_{j,q}(\rho(u), \rho(v))$ and hence, by the induction hypothesis, $\rho(u) \prec^c_{(j,q)} \rho(v)$. Since the completion $\mathbf{D}^c$ satisfies the constraints imposed by the copy function, it follows $u \prec^c_{(i,p)} v$, a contradiction. Case (b) leads in the same way to a contradiction. We conclude by contradiction that $\mathsf{PO}^{\ell}_{i,p} \subseteq \prec^c_{(i,p)}$. The algorithm always terminates due to the fact that only order information is added and there is trivial upper bound for each partial order. Hence $\mathsf{PO}^{\infty}_{i,p} \subseteq \prec^c_{(i,p)}$. Since $\prec^c_{(i,p)}$ is acyclic, the algorithm returns "yes".

The algorithm is clearly in PTIME. Indeed, suppose that in each iteration of Step 3, a single tuple is added, then one needs at most $O(|\mathbf{S}|^2)$ time to add all possible tuples. □

We next show the tractability of COP and DCIP. The PTIME algorithms for these problems rely on the following property of the partial orders $\mathsf{PO}^{\infty}_{i,p}$, computed by the previous algorithm for CPS. Let $\mathbf{S}$ be a specification as given in the proof for CPS above.

LEMMA 6.2. *For each $i \in [1, s]$, $p \in [1, |R_i|]$, we have that $\mathsf{PO}^{\infty}_{i,p} = \bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} \prec^c_{(i,p)}$. That is, the partial orders $\mathsf{PO}^{\infty}_{i,p}$ are certain in every completion and are maximal,* i.e., *no order information can be added without eliminating certain consistent completions.*

PROOF. It is readily verified that the $\boxed{\Rightarrow}$ direction in the previous proof implies that $\bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} \prec^c_{(i,p)} \subseteq \mathsf{PO}^{\infty}_{i,p}$, while the $\boxed{\Leftarrow}$ direction implies that $\mathsf{PO}^{\infty}_{i,p} \subseteq \bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} \prec^c_{(i,p)}$. From these, the desired equality follows. □

COP: Let $\mathbf{S}$ be a specification as described in the proof for CPS above, and O_t be a given currency order. We provide a PTIME algorithm that decides whether O_t is certain for $\mathbf{S}$. We know from Lemma 6.2 that for each $i \in [1, s]$, and $p \in [1, |R_i|]$, $\mathsf{PO}^{\infty}_{i,p} = \bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} \prec^c_{(i,p)}$. Given this we can conclude that O_t is certain iff $\prec_{(i,p)} \subseteq \prec^o_{(i,p)} \subseteq \mathsf{PO}^{\infty}_{i,p}$, where $\prec^o_{(i,p)}$ represents the order for attribute A_p in R_i as specified by O_t. This can be verified in PTIME, using the PTIME algorithm for computing the $\mathsf{PO}^{\infty}_{i,p}$ relations. □

DCIP: Let $\mathbf{S}$ be a specification as described in the proof for CPS above. We provide a PTIME algorithm that decides whether $\mathbf{S}$ is deterministic for current instances.

Let $\mathsf{PO}^{\infty}_{i,p}$ be the partial orders returned by the PTIME algorithm for CPS. From Lemma 6.2 one can easily verify that $\mathbf{S}$ is deterministic iff the following holds: for each $i \in [1, s]$, each $p \in [1, |R_i|]$, and each entity identity eid occurring in $D_{(t,i)}$, the restriction of the partial order $\mathsf{PO}^{\infty}_{i,p}$ to tuples corresponding to eid has only sinks that agree on the A_p attributes. Here a sink means a tuple that has no successors in the partial order. The absence of successors allows us to find completions of the order in which these tuples are put last. Hence, the determinacy condition simply asks for any possible latest tuple to agree on all certain attributes. As a result, the PTIME algorithm (1) computes for $i \in [1, s]$ and $p \in [1, |R_i|]$ the relations $\mathsf{PO}^{\infty}_{(i,p)}$; and (2) verifies for each $\mathsf{eid} \in D_{(t,i)}$ whether $\mathsf{PO}^{\infty}_{i,p}|_{\mathsf{EID}=\mathsf{eid}}$ has only sinks that agree on the A_p attribute. If so, then the algorithm returns "yes", otherwise "no". □

Corollary 3.7 tells us that in the presence of denial constraints, CCQA does not become easier for SP queries than for $\exists\mathrm{FO}^+$. We next show that for SP queries, the absence of denial constraints simplifies the analysis. Indeed, for SP queries and without denial constraints, CCQA is in PTIME.

PROPOSITION 6.3. *For SP queries,* CCQA*(SP) is in PTIME in the absence of denial constraints.*

PROOF. Consider a specification $\mathbf{S}$ in which no denial constraints are defined, while copy functions may be present. Let Q be an SP query. Recall from Section 3 that an SP query is of the form $Q(\vec{x}) = \exists e\, \vec{y}\,\big(R(e, \vec{x}, \vec{y}) \wedge \psi\big)$, where ψ is a conjunction of equality atoms. Here $R(\mathsf{EID}, A_1, \ldots, A_n)$ is a relation schema in $\mathbf{S}$. Let $D_t = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ denote the corresponding temporal instance of R in $\mathbf{S}$. Given $\mathbf{S}$, Q and a tuple t, CCQA is to determine whether t is a certain current answer to Q *w.r.t.* $\mathbf{S}$, *i.e.,* whether $t \in \bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} Q\big(\mathsf{LST}(D^c_t)\big)$, where D^c_t denotes a completion of D_t in $\mathbf{D}^c$.

We develop a PTIME algorithm for CCQA in this context as follows. The algorithm uses the partial order relations defined in the proof of Theorem 6.1. In the following, E denotes the set of all entities occurring in D_t.

(1) Compute the partial order relations $\mathsf{PO}^{\infty}_{\ell}$ for each attribute A_ℓ in R.
(2) For each $e \in E$ and each attribute A_ℓ of R, let $S(e, A_\ell) = \{s[A_\ell] \mid s \text{ is a sink in } \mathsf{PO}^{\infty}_{\ell}\}$. That is, $S(e, A_\ell)$ is the set of all possible most current values of the A_ℓ-attribute that the entity e can have. We define

$$\mathsf{poss}(e, A_\ell) = \begin{cases} s[A_\ell] & \text{if } S(e, A_\ell) = \{s[A_\ell]\} \\ c_{e,\ell} & \text{if } |S(e, A_\ell)| > 1, \end{cases}$$

where $c_{e,\ell}$ is a new constant. In other words, $\mathsf{poss}(e, A_\ell)$ is either the unique most current value of attribute A_ℓ of entity e, or a new constant $c_{e,\ell}$ indicating that multiple distinct current values for A_ℓ and e exist. Let $\mathsf{poss}(e, \mathbf{S})$ be the tuple $\big(e, \mathsf{poss}(e, A_1), \ldots, \mathsf{poss}(e, A_n)\big)$.
(3) Let $\mathsf{poss}(\mathbf{S}) = \bigcup_{e \in E} \mathsf{poss}(e, \mathbf{S})$ and evaluate $Q(\mathsf{poss}(\mathbf{S}))$.
(4) Remove all tuples from $Q\big(\mathsf{poss}(\mathbf{S})\big)$ that contain a new constant (*i.e.,* that contain a constant $c_{e,\ell}$). Denote the resulting set by $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$.
(5) Check whether $t \in \hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$. Return "yes" if so, and return "no" otherwise.

The algorithm is in PTIME. Indeed, Step 1 is in PTIME by Theorem 6.1. Step 3 is in PTIME since Q is an SP query. All the other steps are trivially in PTIME.

We next show the correctness of the algorithm. Let $\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})$ and D_t^c be the completion of D_t in $\mathbf{D}^c$. Recall from the definition of current instances that $\mathsf{LST}(e, D_t^c)$ is a tuple of the form $(e, a_1, \ldots, a_n)$, where for each $\ell \in [1, n]$, a_ℓ is the most current value of the entity e for the attribute A_ℓ relative to D_t^c. As shown in the proof of Theorem 6.1(3), these current values are always witnessed by values that appear in sinks of $\mathsf{PO}_\ell^\infty|_{\mathsf{EID}=e}$. Hence, $\mathsf{poss}(e, \mathbf{S})$, as defined in Step 2, indicates whether there exists a unique most current tuple for e (in case that $\mathsf{poss}(e, \mathbf{S})$ does not contain any new constants) or not (in case that a new constant appears in $\mathsf{poss}(e, \mathbf{S})$). Indeed, in the absence of denial constraints the currency orders among different attributes are independent and every new constant contributes to a different current tuple. Clearly, when new constants in $\mathsf{poss}(e, \mathbf{S})$ interact with the selection conditions in Q, $Q(\mathsf{poss}(e, \mathbf{S}) = \emptyset$, and thus $\mathsf{poss}(e, \mathbf{S})$ does not contribute to the certain current answers to Q *w.r.t.* $\mathbf{S}$. Let $\mathsf{poss}(\mathbf{S}) = \bigcup_{e \in E} \mathsf{poss}(e, \mathbf{S})$ and consider $Q\big(\mathsf{poss}(\mathbf{S})\big)$ (Step 3). Here for every $e \in E$, $Q(\mathsf{poss}(e, \mathbf{S}))$ is a tuple that satisfies the selection conditions. However, apart from normal constants this tuple may contain new constants. This implies that $Q\big(\mathsf{poss}(e, \mathbf{S})\big)$ may still represent distinct possibilities, each of which realized by the query answer in some completion of $\mathbf{S}$. In order to compute the certain answers, one thus needs to eliminate the entities that contain a new constant. In other words, we have to consider $\hat{Q}\big(\mathsf{poss}(e, \mathbf{S})\big)$ (Step 4). Finally, we need to verify whether $t \in \hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$ as is done in Step 5. □

We have seen in Theorems 5.1 and 5.3 that fixing denial constraints does not make our lives easier when it comes to CPP or BCP. However, when denial constraints are absent, these problems become tractable for SP queries.

THEOREM 6.4. *When denial constraints are absent, for SP queries both the combined complexity and the data complexity are in PTIME for* CPP *and* BCP *(when the bound k on the size of additional data copied is fixed).*

PROOF. We first develop a PTIME algorithm for CPP(SP), which will then be used to show that BCP(SP) is also in PTIME, all in the absence of denial constraints.

CPP *for SP:* Consider a specification $\mathbf{S}$ and an SP query $Q(\vec{x}) = \exists e\, \vec{y}(R(\vec{x}, \vec{y}) \wedge \psi)$ for some relation R in $\mathbf{S}$, a subset of attributes $\vec{x}$ of R and a selection condition ψ. Recall that for SP queries, $\bigcap_{\mathbf{D}^c \in \mathsf{Mod}(\mathbf{S})} Q\big(\mathsf{LST}(\mathbf{D}^c)\big) = \hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$, where $\mathsf{poss}(\mathbf{S})$ encodes whether or not a unique current tuple exists in all completions of the temporal instance $D_t = (D, \prec_{A_1}, \ldots, \prec_{A_n})$ of schema R in $\mathbf{S}$. We refer to the proof of Proposition 6.3 for the definition of $\mathsf{poss}(\mathbf{S})$, $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$ and its relation to certain current answers.

We provide a PTIME algorithm that checks whether there exists an extension $\bar{\rho}^e$ of copy functions $\bar{\rho}$ in $\mathbf{S}$, such that $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big) \neq \hat{Q}\big(\mathsf{poss}(\mathbf{S}^e)\big)$, where $\mathbf{S}^e$ is the extension of $\mathbf{S}$ by $\bar{\rho}^e$. If such an extension exists, then the algorithm returns "no"; otherwise it returns "yes". More precisely, we check whether *none* of the following conditions is satisfied:

(C1) There exists a tuple $r_1 \in \hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$ for which there exists an extension $\bar{\rho}^e$ of $\bar{\rho}$ such that $r_1 \notin \hat{Q}\big(\mathsf{poss}(\mathbf{S}^e)\big)$. In other words, $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big) \not\subseteq \hat{Q}\big(\mathsf{poss}(\mathbf{S}^e)\big)$.

(C2) There exists an entity eid in D_t for which there exists an extension $\bar{\rho}^e$ of $\bar{\rho}$ such that the tuple $r_2 = \hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big)$ does not belong to $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$. In other words, $\hat{Q}\big(\mathsf{poss}(\mathbf{S}^e)\big) \not\subseteq \hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$.

Clearly, $\bar{\rho}$ is currency preserving for Q if and only if neither (C1) nor (C2) holds. We next provide PTIME procedures to check these conditions, from which the PTIME complexity of CPP follows.

For each tuple r_1 in $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$, we first identify entities eid in D_t for which $r_1 = \hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S})\big)$. We collect these entities in a set $E(r_1)$. Observe that for condition (C1), the only way that r_1 can be removed from the query result for some extension $\mathbf{S}^e$ of $\mathbf{S}$ is when for each $\mathsf{eid} \in E(r_1)$, the current tuple $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big)$ is either empty or is a tuple different from r_1. By contrast, for condition (C2) it suffices to find *one* $\mathsf{eid} \in E(r_1)$ that gives rise to a distinct new tuple, not appearing anywhere else in the certain current answers of Q *w.r.t.* $\mathbf{S}$.

We use the following notation. Let E denote the set of distinct entity identifiers in D_t. For each $\mathsf{eid} \in E$ and each attribute A_ℓ in R, we denote by $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ the set of tuples in D_t that contribute to the current tuple $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S})\big)$. More specifically, $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ consists of tuples in D_t that (i) are most current in some completion D_t^c of D_t *w.r.t.* A_ℓ (and hence may contribute to the current tuple); (ii) satisfy the selection condition ψ in Q (and thus contribute to the query result); and (iii) share the same eid and A_ℓ-value in case A_ℓ belongs to the projected attributes and attributes involved in the selection condition (and hence relate to same entity and have the same A_ℓ-value as the current tuple). Observe that we can compute $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ in PTIME by leveraging the algorithm given in the proof of Theorem 6.1(1).

For each tuple r_1 in the query result $\hat{Q}\big(\mathsf{poss}(\mathbf{S})\big)$ and for each $\mathsf{eid} \in E(r_1)$ we perform a number of tests as follows.

(1) If we can extend $\bar{\rho}$ to $\bar{\rho}^e$ such that the tuple $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ contains an attribute A_ℓ which is projected on or involved in the selection condition of Q and such that its A_ℓ-value is a new constant, then $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big) = \emptyset$, and we are done for the entity eid under consideration. The existence of such a copy function can be easily checked. We distinguish between the following cases. Let A_ℓ be an attribute that is either projected on or involved in the selection condition of Q.

— If there exist a tuple $t_1 \in \mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ and copy function ρ from D'_t to D_t such that $\rho(t_1) = s_1$, then we check whether D'_t contains a tuple s_2 that satisfies (i) $s_1[B_\ell] \neq s_2[B_\ell]$, *i.e.,* its value in the B_ℓ-attribute in R' (corresponding to A_ℓ in R) is different from the current value; and (ii) s_2 is incomparable with s_1, *i.e.,* neither $s_1 \prec_{B_\ell} s_2$ nor $s_2 \prec_{B_\ell} s_1$ is certain in D'_t. If such s_2 exists, then $t_2 \in D_t^e$ with $\rho^e(t_2) = s_2$ will replace t_1 in a completion of D_t^e. At the same time, t_1 will still be current in another completion (because they are incomparable). Hence, $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ has a new constant in its A_ℓ-attribute and hence $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big) = \emptyset$. We call such s_2 a *spoiler*. These can be detected by calling the PTIME algorithm for COP (Theorem 6.1(2)) for each tuple in D'_t. If this test is successful we flag eid with (C1) and consider the next element in $E(r_1)$. Otherwise we continue.

— If there exists a tuple $t_1 \in \mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ for which a copy function ρ from D'_t to D_t is specified but $\rho(t_1)$ is undefined, we check whether D'_t contains a tuple s_2 that satisfies (i) $s_1[B_\ell] \neq s_2[B_\ell]$; and (ii) $s' \prec_{B_\ell} s_2$ is certain in D'_t for all tuples s' that *are* copied to D_t. We can then import s_2 to a new tuple $t_2 \in D_t^e$ with $\rho^e(t_2) = s_2$. For the same reasons as above, the A_ℓ-attribute of $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ will contain a new constant and hence $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big) = \emptyset$. If this test is successful we flag eid with (C1) and consider the next element in $E(r_1)$. Otherwise we continue.

— If only empty copy functions are specified from D'_t to D_t, we simply check whether D'_t contains a tuple s_1 such that it has a different value in B_ℓ from the current tuple. Then we can import it to a new tuple $t_1 \in D_t^e$ with $\rho^e(t_1) = s_1$. Indeed, t_1 will be incomparable with any of the tuples in $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ and therefore, it makes $\hat{Q}\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big) = \emptyset$. This can be checked again in PTIME.

If this test is successful we flag eid with (C1) and consider the next element in $E(r_1)$. Otherwise we continue.

If none of the above tests is successful, it implies the following. Consider all tuples s_2 in D'_t for which $s_2[B_\ell]$ has a value that differs from the A_ℓ-attribute value of the current tuple. Then, for each such s_2 we can construct a tuple t_2 (with $\rho^e(t_2) = s_2$) via copying, such that tuple t_2 is either more current or more stale (certain order) than all $t_1 \in \mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$. Indeed, otherwise this would lead to the existence of a tuple in $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$ for which newly copied tuples can be both more current and stale, *i.e.,* a tuple as required by one of the above cases. As a consequence, if none of the tests so far is successful, all extensions $\bar{\rho}^e$ are such that $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e))$ contains normal constants in attributes projected out or involved in selections.

(2) We next look for extensions such that $Q\big(\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)\big)$ is empty, or equivalently, such that $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ does not satisfy the selection condition in Q. This happens when a selection condition in Q of the form (i) $\sigma_{A=a}$ or (ii) $\sigma_{A=A'}$ is violated. We next show that both kinds of violations can be detected in PTIME.

— For (i) we simply need to find an extension such that $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ differs from a in the A attribute. We do this as follows: we check whether for each tuple $t_1 \in \mathsf{LWit}(\mathsf{eid}, D_t, A)$, there exists a copy function ρ from D'_t to D_t and tuples s_1, s_2 in D'_t such that $\rho^e(t_1) = s_1$ and $\rho^e(t_2) = s_2$, and moreover, $s_2[B]$ has a value different from a and $s_1 \prec_B s_2$ is certain in D'_t.

— For (ii) we need to find an extension such that $\mathsf{poss}(\mathsf{eid}, \mathbf{S}^e)$ has different A and A' attributes. Along the same lines as in the previous case, we find tuples s_1, s_2 and s_3, such that $s_2[B] \neq s_3[B']$ and, as before, when s_2 and s_3 are copied to tuples in D^e_t, they produce tuples t_2 and t_3 in D^e_t that are more current (certain) than all the tuples in $\mathsf{LWit}(\mathsf{eid}, D_t, A)$ and $\mathsf{LWit}(\mathsf{eid}, D_t, A')$, respectively.

We flag eid with (C1) if it passes one of the above tests and continue with the next entity in $E(r_1)$. Otherwise we continue.

(3) It remains to check whether any of the extensions gives rise to either a tuple different from r_1 (in which case r_1 is eliminated for the current eid) or a tuple different from any of other tuples in the query result (we know at this stage that all extensions provide a tuple in the query result). For this, we only need to consider attributes in the projection of Q. Indeed, any change in the other attributes does not affect the query result. That is, we test whether there exists an attribute A_ℓ, such that there exists a tuple s_1 (as previously described) for which t_1, the tuple to which s_1 is copied to, is more current than all tuples in $\mathsf{LWit}(\mathsf{eid}, D_t, A_\ell)$. Furthermore, either its A_ℓ attribute is distinct from $r_1[A_\ell]$ but may still appear in some other result tuple, or its A_ℓ-attribute is different from any other value in the query result. In the first case, we flag eid with (C1) and move to the next entity in $E(r_1)$. In the second case we flag it with (C2) and conclude that $\bar{\rho}$ is not currency preserving since a new tuple is generated.

If all eid's in $E(r_1)$ are flagged with (C1), then again $\bar{\rho}$ is not currency preserving. If not, we repeat the process for the next tuple in the query result. If at the end of this process it has not been concluded that $\bar{\rho}$ is not currency preserving, then this implies that the query result is unchanged for any extension of the copy function. In other words, $\bar{\rho}$ is currency preserving. □

BCP *for SP:* We show that BCP(SP) is in PTIME in the absence of denial constraints and for fixed k. Consider a specification $\mathbf{S}$ and an SP query Q. The following PTIME algorithm tests whether there exists an extension $\bar{\rho}^e$ of $\bar{\rho}$ with $|\bar{\rho}^e| \leqslant |\bar{\rho}| + k$ such

Table II. Complexity of problems for reasoning about data currency (CPS, COP, DCIP)

	CPS	COP	DCIP
Data complexity	NP-complete (Th 3.1)	coNP-complete (Th 3.4)	coNP-complete (Th 3.4)
Combined complexity	Σ_2^p-complete (Th 3.1)	Π_2^p-complete (Th 3.4)	Π_2^p-complete (Th 3.4)
Special case	In the absence of denial constraints		
Combined and data complexity	PTIME (Th 6.1)	PTIME (Th 6.1)	PTIME (Th 6.1)

Table III. Complexity of problems for query answering and for determining currency preservation

Complexity	CCQA($\mathcal{L}_Q$)	CPP($\mathcal{L}_Q$)	ECP($\mathcal{L}_Q$)	BCP($\mathcal{L}_Q$)
Data	coNP-complete (Th 3.5)	Π_2^p-complete (Th 5.1)	$O(1)$ (Prop 5.2)	Σ_3^p-complete (Th 5.3)
Combined ($\mathcal{L}_Q$)				
CQ, UCQ, $\exists$FO$^+$	Π_2^p-complete (Th 3.5)	Π_3^p-complete (Th 5.1)	$O(1)$ (Prop 5.2)	Σ_4^p-complete (Th 5.3)
FO	PSPACE-complete (Th 3.5)	PSPACE-complete (Th 5.1)	$O(1)$ (Prop 5.2)	PSPACE-complete (Th 5.3)
Special case	SP queries in the absence of denial constraints			
Combined & data	PTIME (Prop. 6.3)	PTIME (Th 6.4)	$O(1)$ (Prop 5.2)	PTIME (Th 6.4)

that $\bar{\rho}^e$ is currency preserving for Q. Since k is fixed, there are only polynomial many extensions $\bar{\rho}^e$ of $\bar{\rho}$. For each of those, we check whether $\bar{\rho}^e$ is currency preserving for Q. Hence, we need to call the above PTIME algorithm for CPP polynomially many times. Therefore, BCP(SP) is in PTIME in this setting. □

7. CONCLUSIONS

We have proposed a model to specify the currency of data in the absence of reliable timestamps but in the presence of copy relationships. We have also introduced a notion of currency preservation to assess copy functions for query answering. We have identified seven fundamental problems associated with data currency and currency preservation (CPS, COP, DCIP, CCQA($\mathcal{L}_Q$), CPP($\mathcal{L}_Q$), ECP($\mathcal{L}_Q$) and BCP($\mathcal{L}_Q$)). We have provided an almost complete picture of the lower and upper bounds of these problems, all matching, for their data complexity as well as combined complexity when $\mathcal{L}_Q$ ranges over a variety of query languages. These results are not only of theoretical interest in their own right, but may also help practitioners distinguish current values from stale data, answer queries with current data, and design proper copy functions to import data from external sources.

The main complexity results are summarized in Tables II and III, annotated with their corresponding theorems. One case we did not study is when queries are in SP, in the *presence* of denial constraints. The results of Tables II and III do not carry over to that setting, since the lower bound proofs for CCQA($\mathcal{L}_Q$), CPP($\mathcal{L}_Q$), ECP($\mathcal{L}_Q$) and BCP($\mathcal{L}_Q$) use queries that involve joins, notably for data complexity when $\mathcal{L}_Q$ is CQ.

The study of data currency is still preliminary. An open issue concerns generalizations of copy functions. To simplify the presentation we assume a single copy function from one relation to another. Nonetheless we believe that all the results remain intact when multiple such functions coexist. For currency-preserving copy functions, we assume that the signatures "cover" all attributes (except EID) of the importing relation. It is nontrivial to relax this requirement, however, since otherwise unknown values need to be introduced for attributes whose value is not provided by the extended copy functions. To this end it is helpful to identify syntactic characterizations of generic currency-preserving copy functions. Another generalization of our model is to extend

current preservation for answering a class of queries rather than a single query. Indeed, in practice currency preservation is often needed for multiple queries.

A second issue is about practical use of the study. As shown in Tables II and III, most of the problems are intractable. To cope with the high complexity we plan to (a) identify practical PTIME cases in various applications, (b) develop efficient heuristic algorithms with certain performance guarantees, and (c) conduct incremental analysis when data or copy functions are updated, which is expected to result in a lower complexity than its batch counterpart when the area affected by the updates is small, as commonly found in practice.

A third issue concerns the interaction between data consistency and data currency. There is an intimate connection between these two central issues of data quality. Indeed, identifying the current value of an entity helps resolve data inconsistencies, and conversely, repairing data helps remove obsolete data. While these processes should logically be unified, we are not aware of any previous work on this topic. A promising approach to tackling this is to develop a uniform logical framework that captures stale values and inconsistencies. This is possible since data inconsistencies are typically detected and fixed by using integrity constraints such as denial constraints [Bertossi 2006; Chomicki 2007] and conditional functional dependencies [Fan et al. 2008], while data currency is also specified in terms of denial constraints. These allow us to strike on data currency and consistency in a unified process.

It should be remarked that the current value of an entity derived from a database may still not be the *true value* of the entity. Indeed, information in a real-life database is often incomplete, with missing tuples and missing values. The chances are that when we derive the current value of an entity, the true values of some attributes of the entity are not collected in the database at all. This highlights the need for studying data currency and complete information together. We intend to tackle this issue by extending the logical framework aforementioned, to check information completeness relative to master data [Fan and Geerts 2011]. Indeed, relative information completeness is also specified in terms of a class of containment constraints, which can be readily incorporated into our framework. Nevertheless, the analyses of data currency in the presence of incomplete and inconsistent data are expected to be more intricate.

Finally, we have so far assumed that for each entity, one can identify tuples pertaining to it based on entity resolution [Elmagarmid et al. 2007]. It is possible that one can unify the process of entity resolution and the process of determining current values. Indeed, recent work has shown that temporal information helps improve the accuracy of entity resolution [Li et al. 2011]. Conversely, accurate matches via entity resolution help us determine the current values of entities. This issue deserves a full treatment.

REFERENCES

ABITEBOUL, S., HULL, R., AND VIANU, V. 1995. *Foundations of Databases*. Addison-Wesley.

BERTI-EQUILLE, L., SARMA, A. D., DONG, X., MARIAN, A., AND SRIVASTAVA, D. 2009. Sailing the information ocean with awareness of currents: Discovery and application of source dependence. In *Proceedings of the 4th Biennial Conference on Innovative Data Systems Research (CIDR)*. Online Proceedings.

BERTOSSI, L. 2006. Consistent query answering in databases. *SIGMOD Rec. 35,* 2, 68–76.

BODIRSKY, M. AND KÁRA, J. 2010. The complexity of temporal constraint satisfaction problems. *Journal of the ACM 57,* 2.

BUNEMAN, P., CHENEY, J., TAN, W., AND VANSUMMEREN, S. 2008. Curated databases. In *Proceedings of the 27th Symposium on Principles of Database Systems (PODS)*. 1–12.

CHENEY, J., CHITICARIU, L., AND TAN, W. C. 2009. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases 1,* 4, 379–474.

CHOMICKI, J. 2007. Consistent query answering: Five easy pieces. In *Proceedings of the 11th International Conference on Database Theory (ICDT)*. 1–17.

CHOMICKI, J. AND TOMAN, D. 2005. Time in database systems. In *Handbook of Temporal Reasoning in Artificial Intelligence*, M. Fisher, D. Gabbay, and L. Vila, Eds. Elsevier.

CLIFFORD, J., DYRESON, C. E., ISAKOWITZ, T., JENSEN, C. S., AND SNODGRASS, R. T. 1997. On the semantics of "now" in databases. *TODS 22,* 2, 171–214.

CODD, E. F. 1979. Extending the database relational model to capture more meaning. *TODS 4,* 4, 397–434.

DEUTSCH, A., NASH, A., AND REMMEL, J. B. 2008. The chase revisited. In *Proceedings of the 27th symposium on Principles of database systems (PODS)*. 149–158.

DONG, X., BERTI-EQUILLE, L., HU, Y., AND SRIVASTAVA, D. 2010. Global detection of complex copying relationships between sources. *PVLDB 3,* 1, 1358–1369.

DONG, X., BERTI-EQUILLE, L., AND SRIVASTAVA, D. 2009. Truth discovery and copying detection in a dynamic world. *PVLDB 2,* 1, 562–573.

DYRESON, C. E., JENSEN, C. S., AND SNODGRASS, R. T. 2009. Now in temporal databases. In *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds. Springer.

ECKERSON, W. W. 2002. Data quality and the bottom line: Achieving business success through a commitment to high quality data. The Data Warehousing Institute.

ELMAGARMID, A. K., IPEIROTIS, P. G., AND VERYKIOS, V. S. 2007. Duplicate record detection: A survey. *TKDE 19,* 1, 1–16.

FAN, W. AND GEERTS, F. 2011. Relative information completeness. *TODS 35,* 4.

FAN, W., GEERTS, F., JIA, X., AND KEMENTSIETSIDIS, A. 2008. Conditional functional dependencies for capturing data inconsistencies. *TODS 33,* 1.

FAN, W., GEERTS, F., LI, J., AND XIONG, M. 2011. Discovering conditional functional dependencies. *TKDE 23,* 5, 683–698.

FAN, W., GEERTS, F., AND WIJSEN, J. 2011. Determining the currency of data. In *Proceedings of the 30th symposium on Principles of database systems (PODS)*. 71–82.

GAREY, M. AND JOHNSON, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

GRAHNE, G. 1991. *The Problem of Incomplete Information in Relational Databases*. Springer.

GROHE, M. AND SCHWANDTNER, G. 2009. The complexity of datalog on linear orders. *Logical Methods in Computer Science 5,* 1.

IMIELIŃSKI, T. AND LIPSKI, JR, W. 1984. Incomplete information in relational databases. *Journal of the ACM 31,* 4, 761–791.

KNOWLEDGE INTEGRITY. 2003. Two sides to data decay. DM Review.

KOLAITIS, P. G. 2005. Schema mappings, data exchange, and metadata management. In *Proceedings of the 24th Symposium on Principles of Database Systems (PODS)*. 61–75.

KOUBARAKIS, M. 1994. Database models for infinite and indefinite temporal information. *Inf. Syst. 19,* 2, 141–173.

KOUBARAKIS, M. 1997. The complexity of query evaluation in indefinite temporal constraint databases. *Theor. Comput. Sci. 171,* 1-2, 25–60.

LENZERINI, M. 2002. Data integration: A theoretical perspective. In *Proceedings of the 21st Symposium on Principles of Database Systems (PODS)*. 233–246.

LI, P., DONG, X. L., MAURICIO, A., AND SRIVASTAVA, D. 2011. Linking temporal records. *PVLDB 4,* 11, 956–967.

PAPADIMITRIOU, C. H. 1994. *Computational Complexity*. Addison-Wesley.

SCHWALB, E. AND VILA, L. 1998. Temporal constraints: A survey. *Constraints 3,* 2-3, 129–149.

SNODGRASS, R. T. 1999. *Developing Time-Oriented Database Applications in SQL*. Morgan Kaufmann.

STOCKMEYER, L. J. 1976. The polynomial-time hierarchy. *Theor. Comput. Sci. 3,* 1, 1–22.

VAN DER MEYDEN, R. 1997. The complexity of querying indefinite data about linearly ordered domains. *JCSS 54,* 1, 113–135.

VAN DER MEYDEN, R. 1998. Logical approaches to incomplete information: A survey. In *Logics for Databases and Information Systems*, J. Chomicki and G. Saake, Eds. Kluwer.

VIANU, V. 1987. Dynamic functional dependencies and database aging. *Journal of the ACM 34,* 1, 28–59.

ZHANG, H., DIAO, Y., AND IMMERMAN, N. 2010. Recognizing patterns in streams with imprecise timestamps. *PVLDB 3,* 1, 244–255.