An Information-Theoretic Approach to Normal Forms for Relational and XML Data

Marcelo Arenas
University of Toronto
marenas@cs.toronto.edu

Leonid Libkin
University of Toronto
libkin@cs.toronto.edu

Abstract

Normalization as a way of producing good database designs is a well-understood topic. However, the same problem of distinguishing well-designed databases from poorly designed ones arises in other data models, in particular, XML. While in the relational world the criteria for being well-designed are usually very intuitive and clear to state, they become more obscure when one moves to more complex data models.

Our goal is to provide a set of tools for testing when a condition on a database design, specified by a *normal form*, corresponds to a good design. We use techniques of information theory, and define a measure of information content of elements in a database with respect to a set of constraints. We first test this measure in the relational context, providing information-theoretic justification for familiar normal forms such as BCNF, 4NF, PJ/NF, 5NFR, DK/NF. We then show that the same measure applies in the XML context, which gives us a characterization of a recently introduced XML normal form called XNF. Finally, we look at information-theoretic criteria for justifying normalization algorithms.

1 Introduction

What constitutes a good database design? This question has been studied extensively, with well-known solutions presented in practically all database texts. But what is it that makes a database design good? This question is usually addressed at a much less formal level. For instance, we know that BCNF is an example of a good design, and we usually say that this is because BCNF eliminates update anomalies. Most of the time this is sufficient, given the simplicity of the relational model and our good intuition about it.

Several papers [15, 30, 20] attempted a more formal evaluation of normal forms, by relating it to the elimination of update anomalies. Another criterion is the existence of algorithms that produce good designs: for example, we know that every database scheme can be losslessly decomposed into one in BCNF, but some constraints may be lost along the way.

The previous work was specific for the relational model. As new data formats such as XML are becoming critically important, classical database theory problems have to be revisited in the new context [28, 26]. However, there is as yet no consensus on how to address the problem of well-designed data in the XML setting [12, 3].

It is problematic to evaluate XML normal forms based on update anomalies; while some proposals for update languages exist [27], no XML update language has been standardized. Likewise, using the existence of good decomposition algorithms as a criterion is problematic: for example, to formulate losslessness, one needs to fix a small set of operations in some language, that would play the same role for XML as relational algebra for relations. Stating dependency preservation

and testing normal forms is even more problematic: while in the relational world we have well-understood procedures for doing this, for XML we do not even know if implication of functional dependencies is decidable.

This suggests that one needs a different approach to the justification of normal forms and good designs. Such an approach must be applicable to new data models *before* the issues of query/update/constraint languages for them are completely understood and resolved. Therefore, such an approach must be based on some intrinsic characteristics of the data, as opposed to query/update languages for a particular data model. In this paper we suggest such an approach based on information-theoretic concepts, more specifically, on measuring the information content of the data. Our goal here is twofold. First, we present information-theoretic measures of "goodness" of a design, and test them in the relational world. To be applicable in other contexts, we expect these measures to characterize familiar normal forms. Second, we apply them in the XML context, and show that they justify a normal form XNF proposed in [3]. We also use our measures to reason about normalization algorithms, by showing that standard decomposition algorithms never decrease the information content of any piece of data in a database/document.

The rest of the paper is organized as follows. In Section 2 we give the notations, and review the basics of information theory (entropy and conditional entropy). Section 3 is an "appetizer" for the main part of the paper: we present a particularly simple information-theoretic way of measuring the information content of a database, and show how it characterizes BCNF and 4NF. The measure, however, is too coarse, and, furthermore, cannot be used to reason about normalization algorithms. In Section 4 we present our main information-theoretic measure of the information content of a database. Unlike the measure studied before [18, 8, 10, 19], our measure takes into account both database instance and schema constraints, and defines the content with respect to a set of constraints. A well-designed database is one in which the content of each datum is the maximum possible. We use this measure to characterize BCNF and 4NF as the best way to design schemas under FDs and MVDs, and to justify normal forms involving JDs (PJ/NF, 5NFR) and other types of integrity constraints (DK/NF). In Section 5, we show that the main measure of Section 4 straightforwardly extends to the XML setting, giving us a definition of well-designed XML specifications. We prove that for constraints given by FDs, well-designed XML specifications are precisely those in XNF. In Section 6, we use the measures of Sections 4 and 5 to reason about normalization algorithms, by showing that good normalization algorithms do not decrease the information content of each datum at every step. Finally, Section 7 presents the conclusions and some ideas for future work.

2 Notations

2.1 Schemas and Instances

A database schema S is a finite set of relation names, with a set of attributes, denoted by $sort(R)$, associated with each $R \in S$. We shall identify $sort(R)$ of cardinality m with $\{1, \ldots, m\}$. Throughout the paper, we assume that the domain of each attribute is $\mathbb{N}^+$, the set of positive integers. An instance I of schema S assigns to each symbol $R \in S$ with $m = |sort(R)|$ a relation $I(R)$ which is a finite set of m-tuples over $\mathbb{N}^+$. By $adom(I)$ we mean the active domain of I, that is, the set of all elements of $\mathbb{N}^+$ that occur in I. The size of $I(R)$ is defined as $\|I(R)\| = |sort(R)| \cdot |I(R)|$, and the size of I is $\|I\| = \sum_{R \in S} \|I(R)\|$. If I is an instance of S, the set of *positions* in I, denoted by $Pos(I)$, is the set $\{(R, t, A) \mid R \in S,\ t \in I(R) \text{ and } A \in sort(R)\}$. Note that $|Pos(I)| = \|I\|$.

We shall deal with *integrity constraints* which are first-order sentences over S. Given a set Σ

of integrity constraints, Σ^+ denotes the set of all constraints implied by it, that is, constraints φ such that for every instance I, $I \models \Sigma$ implies $I \models \varphi$. We define $inst(S, \Sigma)$ as the set of all database instances of S satisfying Σ and $inst_k(S, \Sigma)$ as $\{I \in inst(S, \Sigma) \mid adom(I) \subseteq [1, k]\}$, where $[1, k] = \{1, \ldots, k\}$.

2.2 Constraints and Normal Forms.

Here we briefly review the most common normal forms BCNF, 4NF, PJ/NF, 5NFR and DK/NF. For more information, the reader is referred to [6, 17, 1, 7]. The most widely used among these are BCNF and 4NF, defined in terms of functional dependencies (FD) and multivalued dependencies (MVD), respectively. We shall use the standard notations $X \to Y$ and $X \twoheadrightarrow Y$ for FDs and MVDs. Given a set Σ of FDs over S, (S, Σ) is in BCNF if for every nontrivial FD $X \to Y \in \Sigma^+$, X is a key (that is, if $X \to Y$ is defined over R, then $X \to sort(R) \in \Sigma^+$). If Σ is a set of FDs and MVDs over S, then 4NF is defined analogously [13]: for every nontrivial MVD $X \twoheadrightarrow Y \in \Sigma^+$, X must be a key. Recall that in the case of FDs nontrivial means $Y \not\subseteq X$, and in the case of MVDs nontrivial means $Y \not\subseteq X$ and $X \cup Y \subsetneqq sort(R)$.

The normal forms PJ/NF (projection-join normal form) [14] and 5NFR [29] deal with FDs and join dependencies (JDs). Recall that a JD over $R \in S$ is an expression of the form $\bowtie[X_1, \ldots, X_n]$, where $X_1 \cup \cdots \cup X_n = sort(R)$. A database instance I of S satisfies $\bowtie[X_1, \ldots, X_n]$, if $I(R) = \pi_{X_1}(I(R)) \bowtie \cdots \bowtie \pi_{X_n}(I(R))$. Given a set Σ of FDs and JDs over S, (S, Σ) is in PJ/NF if $\Delta \models \Sigma$, where Δ is the set of key dependencies in Σ^+ (that is, dependencies of the form $X \to sort(R)$ for $X \subseteq sort(R)$). In other words, every instance of S that satisfies all the keys in Σ^+ must satisfy Σ as well. PJ/NF is an extension of both 4NF and BCNF. Since an MVD $X \twoheadrightarrow Y$ over R is a JD $\bowtie[XY, X(sort(R) - Y)]$, when only FDs and MVDs are present in Σ, the definition of PJ/NF coincides with 4NF. If no MVDs are present at all, it reduces to the definition of BCNF [14].

An alternative normal form for FDs and JDs was introduced in [29], which is based on the original definitions of BCNF and 4NF. Given a set of FDs and JDs Σ over S, a JD $\varphi = \bowtie[X_1, \ldots, X_n]$ in Σ is strong-reduced if for every $i \in [1, n]$, $\bowtie[X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n]$ is not in Σ^+ or $X_1 \cup \cdots \cup X_{i-1} \cup X_{i+1} \cup \cdots \cup X_n \subsetneqq sort(R)$. (S, Σ) is in 5NFR (reduced 5th normal form) if for every nontrivial, strong-reduced join dependency $\bowtie[X_1, \ldots, X_n] \in \Sigma^+$ and every $i \in [1, n]$, X_i is a key. PJ/NF is strictly stronger than 5NFR.

The "ultimate" normal form for relational databases was introduced in [15]. This normal form was defined in terms of key dependencies and domain dependencies. In our setting, where domain dependencies are not considered, it says the following. Given *any* set of integrity constraints Σ over S, (S, Σ) is in DK/NF (domain-key normal form) if Σ is implied by the set of key dependencies in Σ^+.

2.3 Basics of Information Theory

The main concept of information theory is that of entropy, which measures the amount of information provided by a certain event. Assume that an event can have n different outcomes s_1, ..., s_n, each with probability p_i, $i \leq n$. How much information is gained by knowing that s_i occurred? This is clearly a function of p_i. Suppose g measures this information; then it must be continuous and decreasing function with domain $(0, 1]$ (the higher the probability, the less information gained) and $g(1) = 0$ (no information is gained if the outcome is known in advance). Furthermore, g is additive: if outcomes are independent, the amount of information gained by knowing two successive outcomes must be the sum of the two individuals amounts, that is, $g(p_i \cdot p_j) = g(p_i) + g(p_j)$. The

A	B	C
1	2	3
1	2	4

(a)

A	B	C
1	1	2
2	3	4

(b)

A	B	C
1	2	3
1	2	4
1	2	5

(c)

Figure 1: Database instances.

only function satisfying these conditions is $g(x) = -c \ln x$, where c is an arbitrary positive constant [25]. It is customary to use base 2 logarithms: $g(x) = -\log x$.

The *entropy* of a probability distribution represents the average amount of information gained by knowing that a particular event occurred. Let $\mathcal{A} = (\{s_1, \ldots, s_n\}, P_{\mathcal{A}})$ be a probability space. If $p_i = P_{\mathcal{A}}(s_i)$, then the entropy of $\mathcal{A}$, denoted by $H(\mathcal{A})$, is defined to be

$$H(\mathcal{A}) \;=\; \sum_{i=1}^{n} p_i \log \frac{1}{p_i} \;=\; -\sum_{i=1}^{n} p_i \log p_i.$$

Observe that some of the probabilities in the space $\mathcal{A}$ can be zero. For that case, we adopt the convention that $0 \log \frac{1}{0} = 0$, since $\lim_{x \to 0} x \log \frac{1}{x} = 0$. It is known that $0 \leq H(\mathcal{A}) \leq \log n$, with $H(\mathcal{A}) = \log n$ only for the uniform distribution $P_{\mathcal{A}}(s_i) = 1/n$ [9].

We shall also use *conditional entropy*. Assume that we are given two probability spaces $\mathcal{A} = (\{s_1, \ldots, s_n\}, P_{\mathcal{A}})$, $\mathcal{B} = (\{s'_1, \ldots, s'_m\}, P_{\mathcal{B}})$ and, furthermore, we know probabilities $P(s'_j, s_i)$ of all the events (s'_j, s_i) (that is, $P_{\mathcal{A}}$ and $P_{\mathcal{B}}$ need not be independent). Then the conditional entropy of $\mathcal{B}$ given $\mathcal{A}$, denoted by $H(\mathcal{B} \mid \mathcal{A})$, gives the average amount of information provided by $\mathcal{B}$ if $\mathcal{A}$ is known [9]. It is defined using conditional probabilities $P(s'_j \mid s_i) = P(s'_j, s_i)/P_{\mathcal{A}}(s_i)$:

$$H(\mathcal{B} \mid \mathcal{A}) = \sum_{i=1}^{n} \left(P_{\mathcal{A}}(s_i) \sum_{j=1}^{m} P(s'_j \mid s_i) \log \frac{1}{P(s'_j \mid s_i)} \right).$$

3 Information Theory and Normal Forms: an Appetizer

We will now see a particularly simple way to provide information-theoretic characterization of normal forms. Although it is very easy to present, it has a number of shortcomings, and a more elaborate measure will be presented in the next section.

Violating a normal form, e.g., BCNF, implies having redundancies. For example, if $S = \{R(A, B, C)\}$ and $\Sigma = \{A \to B\}$, then (S, Σ) is not in BCNF (A is not a key) and some instances can contain redundant information: in Figure 1 (a), the value of the gray cell must be equal to the value below it. We do not need to store this value as it can be inferred from the remaining values and the constraints.

We now use the concept of entropy to measure the information content of every position in an instance of S. The basic idea is as follows: we measure how much information we gain if we lose the value in a given position, and then someone restores it (either to the original, or to some other value, not necessarily from the active domain). For instance, if we lose the value in the gray cell in Figure 1 (a), we gain zero information if it gets restored, since we know from the rest of the instance and the constraints that it equals 2. Formally, let $I \in inst_k(S, \Sigma)$ (that is, $adom(I) \subseteq [1, k]$) and let

$p \in Pos(I)$ be a position in I. For any value a, let $I_{p \leftarrow a}$ be a database instance constructed from I by replacing the value in position p by a. We define a probability space $\mathcal{E}^k_\Sigma(I,p) = ([1,k+1], P)$ and use its entropy as the measure of information in p (we define it on $[1,k+1]$ to guarantee that there is at least one value outside of the active domain). The function P is given by:

$$P(a) = \begin{cases} 0 & I_{p \leftarrow a} \not\models \Sigma, \\ 1/|\{b \mid I_{p \leftarrow b} \models \Sigma\}| & \text{otherwise.} \end{cases}$$

In other words, let m be the number of $b \in [1,k+1]$ such that $I_{p \leftarrow b} \models \Sigma$ (note that $m > 0$ since $I \models \Sigma$). For each such b, $P(b) = 1/m$, and elsewhere $P = 0$. For example, for the instance in Figure 1 (a) if p is the position of the gray cell, then the probability distribution is as follows: $P(2) = 1$ and $P(a) = 0$, for all other $a \in [1,k+1]$. Thus, the entropy of $\mathcal{E}^k_\Sigma(I,p)$ for position p is zero, as we expect. More generally, we can show the following.

Theorem 1 *Let Σ be a set of FDs (or FDs and MVDs) over a schema S. Then (S,Σ) is in BCNF (or 4NF, resp.) if and only if for every $k > 1$, $I \in inst_k(S,\Sigma)$ and $p \in Pos(I)$,*

$$H(\mathcal{E}^k_\Sigma(I,p)) > 0.$$

Proof: We give the proof for the case of FDs; for FDs and MVDs the proof is almost identical.

($\Rightarrow$) Assume that (S,Σ) is in BCNF. Fix $k > 0$, $I \in inst_k(S,\Sigma)$ and $p \in Pos(I)$. Assume that a is the p-th element in I. We show that $I_{p \leftarrow k+1} \models \Sigma$, from which we conclude that $H(\mathcal{E}^k_\Sigma(I,p)) > 0$, since $\mathcal{E}^k_\Sigma(I,p)$ is uniformly distributed, and $P(a), P(k+1) \neq 0$.

Towards a contradiction, assume that $I_{p \leftarrow k+1} \not\models \Sigma$. Then there exist $R \in S$, $t'_1, t'_2 \in I_{p \leftarrow k+1}(R)$ and $X \to A \in \Sigma^+$ such that $t'_1[X] = t'_2[X]$ and $t'_1[A] \neq t'_2[A]$. Assume that t'_1, t'_2 were generated from tuples $t_1, t_2 \in I(R)$ (hence $t_1 \neq t_2$), respectively. Note that $t'_1[X] = t_1[X]$ (if $t_1[X] \neq t'_1[X]$, then $t'_1[B] = k+1$ for some $B \in X$; given that $k+1 \notin adom(I)$, only one position in $I_{p \leftarrow k+1}$ mentions this value and, therefore, $t'_1[X] \neq t'_2[X]$, a contradiction). Similarly, $t'_2[X] = t_2[X]$ and, therefore, $t_1[X] = t_2[X]$. Given that (S,Σ) is in BCNF, X must be a key in R. Hence, $t_1 = t_2$, since $I \models \Sigma$, which is a contradiction.

($\Leftarrow$) Assume that (S,Σ) is not in BCNF. We show that there exists $k > 0$, $I \in inst_k(S,\Sigma)$ and $p \in Pos(I)$ such that $H(\mathcal{E}^k_\Sigma(I,p)) = 0$. Since (S,Σ) is not in BCNF, there exist $R \in S$ and $X \to A \in \Sigma^+$ such that $A \notin X$, $X \cup \{A\} \subsetneq sort(R)$ and X is not a key in R. Thus, there exists a database instance I of S such that $I \models \Sigma$ and $I \not\models X \to sort(R)$. We can assume that $I(R)$ contains only two tuples, say t_1, t_2. Let k be the greatest value in I, $i = t_1[A]$ and p be the position of $t_1[A]$ in I. It is easy to see that $I \in inst_k(S,\Sigma)$ and $P(j) = 0$, for every $j \neq i$ in $[1,k+1]$, since $t_1[A]$ must be equal to $t_2[A] = i$. Therefore, $H(\mathcal{E}^k_\Sigma(I,p)) = 0$. □

Thus, a schema is in BCNF or 4NF iff for every instance, each position carries non-zero amount of information. This is a clean characterization of BCNF and 4NF, but the measure $H(\mathcal{E}^k_\Sigma(I,p))$ is not accurate enough for a number of reasons. For example, let $\Sigma_1 = \{A \to B\}$ and $\Sigma_2 = \{A \twoheadrightarrow B\}$. The instance I in Figure 1 (a) satisfies Σ_1 and Σ_2. Let p be the position of the gray cell in I. Then $H(\mathcal{E}^k_{\Sigma_1}(I,p)) = H(\mathcal{E}^k_{\Sigma_2}(I,p)) = 0$. But intuitively, the information content of p must be higher under Σ_2 than Σ_1, since Σ_1 says that the value in p must be equal to the value below it, and Σ_2 says that this should only happen if the values of the C-attribute are distinct.

Next, consider I_1 and I_2 shown in Figures 1 (a) and (c), respectively. Let $\Sigma = \{A \to B\}$, and let p_1 and p_2 denote the positions of the gray cells in I_1 and I_2. Then $H(\mathcal{E}^k_\Sigma(I_1,p_1)) = H(\mathcal{E}^k_\Sigma(I_2,p_2)) = 0$. But again we would like them to have different values, as the amount of redundancy is higher

A	B	C
6	5	4
3	2	1

(a) An enumeration of I

A	B	C
1	7	3
1	2	4

(b) $I_{(7,\bar{a}_1)} = \sigma_1(I_{(7,\bar{a}_1)})$

A	B	C
v_6	7	3
1	2	v_1

(c) $I_{(7,\bar{a}_2)}$

A	B	C
8	7	3
1	2	4

(d) $\sigma_2(I_{(7,\bar{a}_2)})$

Figure 2: Defining $\textsc{Inf}^k_I(p \mid \Sigma)$.

in I_2 than in I_1. Finally, let $S = R(A, B)$, $\Sigma = \{\emptyset \twoheadrightarrow A\}$, and $I = \{1, 2\} \times \{3, 4\} \in \mathit{inst}(S, \Sigma)$. For each position, the entropy would be zero. However, consider both positions in attribute A corresponding to the value 1. If they both disappear, then we know that no matter how they are restored, the values must be the same. The measure presented in this section cannot possibly talk about inter-dependencies of this kind.

In the next section we will present a measure that overcomes these problems.

4 A General Definition of Well-Designed Data

Let S be a schema, Σ a set of constraints, and $I \in \mathit{inst}(S, \Sigma)$ an instance with $\|I\| = n$. Recall that $\mathit{Pos}(I)$ is the set of positions in I, that is, $\{(R, t, A) \mid R \in S,\ t \in I(R) \text{ and } A \in \mathit{sort}(R)\}$. Our goal is to define a function $\textsc{Inf}_I(p \mid \Sigma)$, the information content of a position $p \in \mathit{Pos}(I)$ with respect to the set of constraints Σ. For a general definition of well-designed data, we want to say that this measure has the maximum possible value. This is a bit problematic for the case of an infinite domain ($\mathbb{N}^+$), since we only know what the maximum value of entropy is for a discrete distribution over k elements: $\log k$. To overcome this, we define, for each $k > 0$, a function $\textsc{Inf}^k_I(p \mid \Sigma)$ that would only apply to instances whose active domain is contained in $[1, k]$, and then consider the ratio $\textsc{Inf}^k_I(p \mid \Sigma)/\log k$. This ratio tells us how close the given position p is to having the maximum possible information content, for databases with active domain in $[1, k]$. As our final measure $\textsc{Inf}_I(p \mid \Sigma)$ we then take the limit of this sequence as k goes to infinity.

Informally, $\textsc{Inf}^k_I(p \mid \Sigma)$ is defined as follows. Let $X \subseteq \mathit{Pos}(I) - \{p\}$. Suppose the values in those positions X are lost, and then someone restores them from the set $[1, k]$; we measure how much information about the value in p this gives us. This measure is defined as the entropy of a suitably chosen distribution. Then $\textsc{Inf}^k_I(p \mid \Sigma)$ is the average such entropy over all sets $X \subseteq \mathit{Pos}(I) - \{p\}$. Note that this is much more involved than the definition of the previous section, as it takes into account all possible interactions between different positions in an instance and the constraints.

We now present this measure formally. An *enumeration* of I with $\|I\| = n$, $n > 0$, is a bijection f_I between $\mathit{Pos}(I)$ and $[1, n]$. From now on, we assume that every instance has an associated enumeration[1]. We say that the position of $(R, t, A) \in \mathit{Pos}(I)$ is p in I if the enumeration of I assigns p to (R, t, A), and if R is clear from the context, we say that the position of $t[A]$ is p. We normally associate positions with their rank in the enumeration f_I.

Fix a position $p \in \mathit{Pos}(I)$. As the first step, we need to describe all possible ways of removing values in a set of positions X, different from p. To do this, we shall be placing variables from a set $\{v_i \mid i \geq 1\}$ in positions where values are to be removed, where v_i can occur only in position i. Furthermore, we assume that each set of positions is equally likely to be removed. To model this, let $\Omega(I, p)$ be the set of all 2^{n-1} vectors $(a_1, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n)$ such that for every $i \in [1, n] - \{p\}$,

[1] The choice of a particular enumeration will not affect the measures we define.

a_i is either v_i or the value in the i-th position of I. A probability space $\mathcal{A}(I,p) = (\Omega(I,p), P)$ is defined by taking P to be the uniform distribution.

Example 1: Let I be the database instance shown in Figure 1 (a). An enumeration of the positions in I is shown in Figure 2 (a). Assume that p is the position of the gray cell shown in Figure 1 (a), that is, $p = 5$. Then $\bar{a}_1 = (4, 2, 1, 3, 1)$ and $\bar{a}_2 = (v_1, 2, 1, 3, v_6)$ are among the 32 vectors in $\Omega(I,p)$. For each of these vectors, we define P as $\frac{1}{32}$. □

Our measure $\textsc{Inf}_I^k(p \mid \Sigma)$, for $I \in inst_k(S,\Sigma)$, will be defined as the conditional entropy of a distribution on $[1,k]$, given the above distribution on $\Omega(I,p)$. For that, we define conditional probabilities $P(a \mid \bar{a})$ that characterize how likely a is to occur in position p, if some values are removed from I according to the tuple $\bar{a}$ from $\Omega(I,p)$ [2]. We need a couple of technical definitions first. If $\bar{a} = (a_i)_{i \neq p}$ is a vector in $\Omega(I,p)$ and $a > 0$, then $I_{(a,\bar{a})}$ is a table obtained from I by putting a in position p, and a_i in position $i, i \neq p$. If $k > 0$, then a *substitution* $\sigma : \bar{a} \to [1,k]$ assigns a value from $[1,k]$ to each a_i which is a variable, and leaves other a_is intact. We can extend σ to $I_{(a,\bar{a})}$ and thus talk about $\sigma(I_{(a,\bar{a})})$.

Example 2: (example 1 continued) Let $k = 8$ and σ_1 be an arbitrary substitution from $\bar{a}_1$ to $[1,8]$. Note that σ_1 is the identity substitution, since $\bar{a}_1$ contains no variables. Figure 2 (b) shows $I_{(7,\bar{a}_1)}$, which is equal to $\sigma_1(I_{(7,\bar{a}_1)})$. Let σ_2 be a substitution from $\bar{a}_2$ to $[1,8]$ defined as follows: $\sigma(v_1) = 4$ and $\sigma(v_6) = 8$. Figure 2 (c) shows $I_{(7,\bar{a}_2)}$ and Figure 2 (d) shows the database instance generated by applying σ_2 to $I_{(7,\bar{a}_2)}$. □

If Σ is a set of constraints over S, then $SAT_\Sigma^k(I_{(a,\bar{a})})$ is defined as the set of all substitutions $\sigma : \bar{a} \to [1,k]$ such that $\sigma(I_{(a,\bar{a})}) \models \Sigma$ and $\|\sigma(I_{(a,\bar{a})})\| = \|I\|$ (the latter ensures that no two tuples collapse as the result of applying σ). With this, we define $P(a \mid \bar{a})$ as:

$$P(a \mid \bar{a}) \quad = \quad \frac{|SAT_\Sigma^k(I_{(a,\bar{a})})|}{\sum_{b \in [1,k]} |SAT_\Sigma^k(I_{(b,\bar{a})})|}.$$

We remark that this corresponds to conditional probabilities with respect to a distribution P' on $[1,k] \times \Omega(I,p)$ defined by $P'(a,\bar{a}) = P(a \mid \bar{a}) \cdot (1/2^{n-1})$, and that P' is indeed a probability distribution for every $I \in inst_k(S,\Sigma)$ and $p \in Pos(I)$.

Example 3: (example 2 continued) Assume that $\Sigma = \{A \to B\}$. Given that the only substitution σ from $\bar{a}_1$ to $[1,8]$ is the identity, for every $a \in [1,8]$, $a \neq 2$, $\sigma(I_{(a,\bar{a}_1)}) \not\models \Sigma$, and, therefore, $SAT_\Sigma^8(I_{(a,\bar{a}_1)}) = \emptyset$. Thus, $P(2 \mid \bar{a}_1) = 1$ since $\sigma(I_{(2,\bar{a}_1)}) \models \Sigma$. This value reflects the intuition that if the value in the gray cell of the instance shown in Figure 1 (a) is removed, then it can be inferred from the remaining values and the FD $A \to B$.

There are 64 substitutions with domain $\bar{a}_2$ and range $[1,8]$. A substitution σ is in $SAT_\Sigma^8(I_{(7,\bar{a}_2)})$ if and only if $\sigma(v_6) \neq 1$, and, therefore, $|SAT_\Sigma^8(I_{(7,\bar{a}_2)})| = 56$. The same can be proved for every $a \in [1,8]$, $a \neq 2$. On the other hand, the only substitution that is not in $SAT_\Sigma^8(I_{(2,\bar{a}_2)})$ is $\sigma(v_1) = 3$ and $\sigma(v_6) = 1$, since $\sigma(I_{(2,\bar{a}_2)})$ contains only one tuple. Thus, $|SAT_\Sigma^8(I_{(2,\bar{a}_2)})| = 63$ and, therefore,

$$P(a \mid \bar{a}_2) = \begin{cases} \frac{63}{455} & \text{if } a = 2, \\ \\ \frac{56}{455} & \text{otherwise.} \end{cases}$$

□

[2] We use the same letter P here, but this will never lead to confusion. Furthermore, all probability distributions depend on I, p, k and Σ, but we omit them as parameters of P since they will always be clear from the context.

We define a probability space $\mathcal{B}^k_\Sigma(I,p) = ([1,k], P)$ where

$$P(a) \quad = \quad \frac{1}{2^{n-1}} \sum_{\bar{a} \in \Omega(I,p)} P(a \mid \bar{a}) \ ,$$

and, again, omit I, p, k and Σ as parameters, and overload the letter P since this will never lead to confusion.

The measure of the amount of information in position p, $\textsc{Inf}^k_I(p \mid \Sigma)$, is the conditional entropy of $\mathcal{B}^k_\Sigma(I,p)$ given $\mathcal{A}(I,p)$, that is, the average information provided by p, given all possible ways of removing values in the instance I:

$$\textsc{Inf}^k_I(p \mid \Sigma) \stackrel{\text{def}}{=} H(\mathcal{B}^k_\Sigma(I,p) \mid \mathcal{A}(I,p)) = \sum_{\bar{a} \in \Omega(I,p)} \left(P(\bar{a}) \sum_{a \in [1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \right).$$

Note that for $\bar{a} \in \Omega(I,p)$, $\sum_{a \in [1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})}$ measures the amount of information in position p, given a set of constraints Σ and some missing values in I, represented by the variables in $\bar{a}$. Thus, $\textsc{Inf}^k_I(p \mid \Sigma)$ is the average such amount over all $\bar{a} \in \Omega(I,p)$. Furthermore, from the definition of conditional entropy, $0 \leq \textsc{Inf}^k_I(p \mid \Sigma) \leq \log k$, and the measure $\textsc{Inf}^k_I(p \mid \Sigma)$ depends on the domain size k. We now consider the ratio of $\textsc{Inf}^k_I(p \mid \Sigma)$ and the maximum entropy $\log k$. It turns out that this sequence converges:

Lemma 1 *If Σ is a set of first-order constraints over a schema S, then for every $I \in inst(S,\Sigma)$ and $p \in Pos(I)$, $\lim_{k\to\infty} \textsc{Inf}^k_I(p \mid \Sigma)/\log k$ exists.*

The proof of this lemma is given in appendix A.1. In fact, Lemma 1 shows that such a limit exists for any set of *generic* constraints, that is, constraints that do not depend on the domain. This finally gives us the definition of $\textsc{Inf}_I(p \mid \Sigma)$.

Definition 1 *For $I \in inst(S,\Sigma)$ and $p \in Pos(I)$, the measure $\textsc{Inf}_I(p \mid \Sigma)$ is defined as*

$$\lim_{k\to\infty} \frac{\textsc{Inf}^k_I(p \mid \Sigma)}{\log k} \ .$$

$\textsc{Inf}_I(p \mid \Sigma)$ measures how much information is contained in position p, and $0 \leq \textsc{Inf}_I(p \mid \Sigma) \leq 1$. A well-designed schema should not have an instance with a position that has less than maximum information:

Definition 2 *A database specification (S,Σ) is* well-designed *if for every $I \in inst(S,\Sigma)$ and every $p \in Pos(I)$, $\textsc{Inf}_I(p \mid \Sigma) = 1$.*

Example 4: Let S be a database schema $\{R(A, B, C)\}$. Let $\Sigma_1 = \{A \to BC\}$. Figure 1 (b) shows an instance I of S satisfying Σ_1 and Figure 3 (a) shows the value of $\textsc{Inf}^k_I(p \mid \Sigma_1)$ for $k = 5, 6, 7$, where p is the position of the gray cell. As expected, the value of $\textsc{Inf}^k_I(p \mid \Sigma_1)$ is maximal, since (S,Σ_1) is in BCNF. Indeed, given that we have to preserve the number of tuples, the A-values must be distinct, hence all possibilities for selecting B and C are open.

The next two examples show that the measure $\textsc{Inf}^k_I(p \mid \Sigma)$ can distinguish cases that were indistinguishable with the measure of Section 3. Let $\Sigma_2 = \{A \to B\}$ and $\Sigma_2' = \{A \twoheadrightarrow B\}$. Figure 1 (a) shows an instance I of S satisfying both Σ_2 and Σ_2'. Figure 3 (b) shows the value of $\textsc{Inf}^k_I(p \mid \Sigma_2)$ and $\textsc{Inf}^k_I(p \mid \Sigma_2')$ for $k = 5, 6, 7$. As expected, the values are smaller for Σ_2. Finally,

k	$A \to BC$	$\log k$
5	2.3219	2.3219
6	2.5850	2.5850
7	2.8074	2.8074

(a)

k	$A \to B$	$A \to\!\!\to B$
5	2.0299	2.2180
6	2.2608	2.4637
7	2.4558	2.6708

(b)

k	I_1	I_2
5	2.0299	1.8092
6	2.2608	2.0167
7	2.4558	2.1914

(c)

Figure 3: Value of conditional entropy.

let $\Sigma_3 = \{A \to B\}$. Figures 1 (a) and 1 (c) show instances I_1, I_2 of S satisfying Σ_3. We expect the information content of the gray cell to be smaller in I_2 than in I_1, but the measure used in Section 3 could not distinguish them. Figure 3 (c) shows the values of $\textsc{Inf}^k_{I_1}(p \mid \Sigma_3)$ and $\textsc{Inf}^k_{I_2}(p \mid \Sigma_3)$ for $k = 5, 6, 7$. As expected, the values are smaller for I_2. In fact, $\textsc{Inf}_{I_1}(p \mid \Sigma_3) = 0.875$ and $\textsc{Inf}_{I_2}(p \mid \Sigma_3) = 0.78125$. □

4.1 Basic Properties

It is clear from the definitions that $\textsc{Inf}_I(p \mid \Sigma)$ does not depend on a particular enumeration of positions. Two other basic properties that we can expect from the measure of information content are as follows: first, it should not depend on a particular representation of constraints, and second, a schema without constraints must be well-designed (as there is nothing to tell us that it is not). Both are indeed true.

Proposition 1

1) Let Σ_1 and Σ_2 be two sets of constraints over a schema S. If they are equivalent (that is, $\Sigma_1^+ = \Sigma_2^+$), then for any instance I satisfying Σ_1 and any $p \in Pos(I)$, $\textsc{Inf}_I(p \mid \Sigma_1) = \textsc{Inf}_I(p \mid \Sigma_2)$.

2) If $\Sigma = \emptyset$, then (S, Σ) is well-designed.

Proof:

1) Follows from the fact that for every instance I of S, $I \models \Sigma_1$ iff $I \models \Sigma_2$. Hence, for every $a \in [1, k]$ and $\bar{a} \in \Omega(I, p)$, $SAT^k_{\Sigma_1}(I_{(a,\bar{a})}) = SAT^k_{\Sigma_2}(I_{(a,\bar{a})})$ and, therefore, $H(\mathcal{B}^k_{\Sigma_1}(I, p) \mid \mathcal{A}(I, p)) = H(\mathcal{B}^k_{\Sigma_2}(I, p) \mid \mathcal{A}(I, p))$.

2) Follows from part 2) of Proposition 2, to be proved below. Since for every $I \in inst(S, \Sigma)$, $p \in Pos(I)$ and $a \in \mathbb{N}^+ - adom(I)$, we have $I_{p \leftarrow a} \models \Sigma$, this implies that (S, Σ) is well-designed.

□

In the following proposition we show a very useful structural criterion for $\textsc{Inf}_I(p \mid \Sigma) = 1$, namely that a schema (S, Σ) is well-designed if and only if one position of an arbitrary $I \in inst(S, \Sigma)$ can always be assigned a fresh value. Also in this proposition, we use this criterion to show that $\textsc{Inf}^k_I(p \mid \Sigma)$ cannot exhibit sub-logarithmic growth, that is, if $\lim_{k\to\infty} \textsc{Inf}^k_I(p \mid \Sigma)/\log k = 1$, then $\lim_{k\to\infty}[\log k - \textsc{Inf}^k_I(p \mid \Sigma)] = 0$.

Proposition 2 *Let S be a schema and Σ a set of constraints over S. Then the following are equivalent.*

1) (S, Σ) *is well-designed.*

2) For every $I \in inst(S, \Sigma)$, $p \in Pos(I)$ *and* $a \in \mathbb{N}^+ - adom(I)$, $I_{p \leftarrow a} \models \Sigma$.

3) For every $I \in inst(S, \Sigma)$ *and* $p \in Pos(I)$, $\lim_{k \to \infty} [\log k - \textsc{Inf}_I^k(p \mid \Sigma)] = 0$.

The following lemma will be used in the proof of this proposition and in several other proofs.

Lemma 2 *Fix* $n, m > 0$, *an* n*-element set* A *and a probability space* $\mathcal{A}$ *on* A *with the uniform distribution* $P_{\mathcal{A}}$. *Assume that for each* $k > 0$, *we have a probability space on* $[1, k]$ *called* $\mathcal{B}_k$ *and a joint distribution* $P_{\mathcal{B}_k, \mathcal{A}}$ *on* $[1, k] \times A$ *such that for some* $a_0 \in A$, *and for all* $k > 0$, *the conditional probability* $P(i \mid a_0) = P_{\mathcal{B}_k, \mathcal{A}}(i, a_0)/P_{\mathcal{A}}(a_0) = 0$, *for at least* $k - m$ *elements of* $[1, k]$. *Then for every* $k > m^2$*:*

$$\frac{H(\mathcal{B}_k \mid \mathcal{A})}{\log k} \;<\; 1 - \frac{1}{2n}.$$

In particular, if $\lim_{k \to \infty} H(\mathcal{B}_k \mid \mathcal{A})/\log k$ *exists, then* $\lim_{k \to \infty} H(\mathcal{B}_k \mid \mathcal{A})/\log k \;<\; 1$.

Proof: First, assume that $m > 1$. Let $k > m^2$ and $M = \{i \in [1, k] \mid P(i \mid a_0) > 0\}$. Observe that $|M| \leq m$. Then

$$\begin{aligned}
\frac{H(\mathcal{B}_k \mid \mathcal{A})}{\log k} &= \frac{1}{\log k}\left[\sum_{a \in A} \frac{1}{n} \sum_{i \in [1,k]} P(i \mid a) \log \frac{1}{P(i \mid a)}\right] \\
&= \frac{1}{n \log k}\left[\left(\sum_{a \in A - \{a_0\}} \sum_{i \in [1,k]} P(i \mid a) \log \frac{1}{P(i \mid a)}\right) + \left(\sum_{i \in [1,k]} P(i \mid a_0) \log \frac{1}{P(i \mid a_0)}\right)\right] \\
&= \frac{1}{n \log k}\left[\left(\sum_{a \in A - \{a_0\}} \sum_{i \in [1,k]} P(i \mid a) \log \frac{1}{P(i \mid a)}\right) + \left(\sum_{i \in M} P(i \mid a_0) \log \frac{1}{P(i \mid a_0)}\right)\right] \\
&\leq \frac{1}{n \log k}\left[\left(\sum_{a \in A - \{a_0\}} \log k\right) + \log m\right] \qquad (1) \\
&= \frac{1}{n \log k}\left[(n-1) \log k + \log m\right] \\
&= 1 - \frac{1}{n} + \frac{\log m}{n \log k} \;<\; 1 - \frac{1}{n} + \frac{\log m}{n \log m^2} \;=\; 1 - \frac{1}{n} + \frac{1}{2n} \;=\; 1 - \frac{1}{2n}.
\end{aligned}$$

Now, assume that $m = 1$. In this case, $\log m$ in equation (1) is equal to 0 and, therefore, the previous sequence of formulas show that $H(\mathcal{B}_k \mid \mathcal{A})/\log k \leq 1 - \frac{1}{n} < 1 - \frac{1}{2n}$. □

Proof of Proposition 2: We will prove the chain of implications 3) ⇒ 1) ⇒ 2) ⇒ 3).

The implication 3) ⇒ 1) is straightforward. Next we show 1) ⇒ 2). Towards a contradiction, assume that there exists $I \in inst(S, \Sigma)$, $p \in Pos(I)$ and $a \in \mathbb{N}^+ - adom(I)$ such that $I_{p \leftarrow a} \not\models \Sigma$. Let $k > 0$ be such that $adom(I) \cup \{a\} \subseteq [1, k]$. By Claim 1 (see Appendix), for every $b \in [1, k] - adom(I)$, $I_{p \leftarrow b} \not\models \Sigma$. Thus, for every $a \in [1, k] - adom(I)$, $P(a \mid \bar{a}_0) = 0$, where $\bar{a}_0$ is the tuple in $\Omega(I, p)$ containing no variables. Therefore, applying Lemma 2 with $n = 2^{\|I\|-1}$ and $m = |adom(I)|$, we conclude that for $k > m^2$:

$$\frac{\textsc{Inf}_I^k(p \mid \Sigma)}{\log k} \;=\; \frac{H(\mathcal{B}_\Sigma^k(I, p) \mid \mathcal{A}(I, p))}{\log k} \;<\; 1 - \frac{1}{2 \cdot 2^{\|I\|-1}}.$$

Since $\textsc{Inf}_I(p \mid \Sigma) = \lim_{k\to\infty} \textsc{Inf}_I^k(p \mid \Sigma)/\log k$ exists by Lemma 1, we conclude that $\textsc{Inf}_I(p \mid \Sigma) < 1$ and thus (S, Σ) is not well-designed, a contradiction.

Next, we show 2) $\Rightarrow$ 3). Let $I \in inst(I, \Sigma)$ and $p \in Pos(I)$. Let $\|I\| = n$ and let $k > n$ be such that $I \in inst_k(S, \Sigma)$. First, we prove that for every $a \in [1, k] - adom(I)$ and $\bar{a} \in \Omega(I, p)$,

$$|SAT_\Sigma^k(I_{(a,\bar{a})})| \quad \geq \quad (k-n)^{|var(\bar{a})|} \tag{2}$$

where $var(\bar{a})$ is the set of variables in $\bar{a}$. We do it by induction on $|var(\bar{a})|$ [3]. We do it by induction on $|var(\bar{a})|$. Assume that $|var(\bar{a})| = 0$. Then given that $I_{p\leftarrow a} \models \Sigma$, we conclude that $|SAT_\Sigma^k(I_{(a,\bar{a})})| = 1$. Now assume that (2) is true for every tuple in $\Omega(I, p)$ containing at most m variables, and let $|var(\bar{a})| = m + 1$. Suppose that $\bar{a} = (a_1, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n)$ and $a_i = v_i$, for some $i \in [1, p-1] \cup [p+1, n]$. Let $I' = I_{p\leftarrow a}$. By the assumption, $I' \models \Sigma$, and hence for every $b \in [1, k] - adom(I')$ we have $I'_{i\leftarrow b} \models \Sigma$. Thus, given that $|[1, k] - adom(I')| \geq k - n$ and for every $b_1, b_2 \in [1, k] - adom(I')$, $|SAT_\Sigma^k(I'_{(a,\bar{b}_1)})| = |SAT_\Sigma^k(I'_{(a,\bar{b}_2)})|$, where $\bar{b}_j$ $(j = 1, 2)$ is a tuple constructed from $\bar{a}$ by replacing v_i by b_j, we conclude that if $\bar{b}$ is a tuple constructed from $\bar{a}$ by replacing v_i by an arbitrary $b \in [1, k] - adom(I')$, then $|SAT_\Sigma^k(I_{(a,\bar{a})})| \geq (k-n) \cdot |SAT_\Sigma^k(I'_{(a,\bar{b})})|$, since $|adom(I')| \leq n$. By the induction hypothesis, $|SAT_\Sigma^k(I'_{(a,\bar{b})})| \geq (k-n)^{|var(\bar{b})|} = (k-n)^{|var(\bar{a})|-1}$ and, therefore, $|SAT_\Sigma^k(I_{(a,\bar{a})})| \geq (k-n)^{|var(\bar{a})|}$, proving (2).

Now we show that $\lim_{k\to\infty}[\log k - \textsc{Inf}_I^k(p \mid \Sigma)] = 0$. For every $k \geq 1$ such that $adom(I) \subseteq [1, k]$, $\log k \geq \textsc{Inf}_I^k(p \mid \Sigma)$ and, therefore, $\lim_{k\to\infty}[\log k - \textsc{Inf}_I^k(p \mid \Sigma)] \geq 0$. Hence, to prove the theorem we will show that

$$\lim_{k\to\infty}[\log k - \textsc{Inf}_I^k(p \mid \Sigma)] \quad \leq \quad 0. \tag{3}$$

Let $k \geq 1$ be such that $adom(I) \subseteq [1, k]$. Assume that $k > n$. Let $a \in [1, k] - adom(I)$ and $\bar{a} \in \Omega(I, p)$. Since $\sum_{b\in[1,k]} |SAT_\Sigma^k(I_{(b,\bar{a})})| \leq k^{|var(\bar{a})|+1}$, using (2), we get

$$P(a \mid \bar{a}) \;\geq\; \frac{(k-n)^{|var(\bar{a})|}}{k^{|var(\bar{a})|+1}} \;=\; \frac{1}{k}\,(1 - \frac{n}{k})^{|var(\bar{a})|}. \tag{4}$$

By Claim 1 (see Appendix), for every $a, b \in [1, k] - adom(I)$ and every $\bar{a} \in \Omega(I, p)$, $P(a \mid \bar{a}) = P(b \mid \bar{a})$. Thus, for every $a \in [1, k] - adom(I)$ and every $\bar{a} \in \Omega(I, p)$,

$$P(a \mid \bar{a}) \;\leq\; 1/(k - |adom(I)|) \;\leq\; 1/(k-n). \tag{5}$$

In order to prove (3), we need to establish a lower bound for $\textsc{Inf}_I^k(p \mid \Sigma)$. We do this by using (4) and (5) as follows: Given the term $P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})}$, we use (4) and (5) to replace $P(a \mid \bar{a})$ and

[3] This induction relies on the following simple idea: If $a \notin adom(I)$, then $I_{p\leftarrow a} \models \Sigma$ and, therefore, one can replace values in positions of $\bar{a}$ one by one, provided that each position gets a fresh value.

$\log \frac{1}{P(a|\bar{a})}$ by smaller terms, respectively. More precisely,

$$\begin{aligned}
\textsc{Inf}_I^k(p \mid \Sigma) &= \sum_{\bar{a}\in\Omega(I,p)} \left(P(\bar{a}) \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \right) \\
&\geq \frac{1}{2^{n-1}} \sum_{a\in[1,k]-adom(I)} \sum_{\bar{a}\in\Omega(I,p)} \frac{1}{k} \, (1-\frac{n}{k})^{|var(\bar{a})|} \log(k-n) \\
&= \frac{1}{2^{n-1}} \, \log(k-n) \, \frac{1}{k} \sum_{a\in[1,k]-adom(I)} \sum_{i=0}^{n-1} \binom{n-1}{i} (1-\frac{n}{k})^i \\
&= \frac{1}{2^{n-1}} \, \log(k-n) \, \frac{1}{k} \sum_{a\in[1,k]-adom(I)} ((1-\frac{n}{k})+1)^{n-1} \\
&\geq \frac{1}{2^{n-1}} \, \log(k-n) \, \frac{1}{k} \, (k-n) \, (2-\frac{n}{k})^{n-1} \\
&\geq \frac{1}{2^{n-1}} \, \log(k-n) \, \frac{1}{k} \, (k-n) \, (2-\frac{2n}{k})^{n-1} \\
&= \frac{1}{2^{n-1}} \, \log(k-n) \, (1-\frac{n}{k}) \, 2^{n-1} \, (1-\frac{n}{k})^{n-1} \\
&= \log(k-n) \, (1-\frac{n}{k})^n.
\end{aligned}$$

Therefore, $\log k - \textsc{Inf}_I^k(p \mid \Sigma) \leq \log k - \log(k-n) \, (1-\frac{n}{k})^n$. Since $\lim_{k\to\infty}[\log k - \log(k-n) \, (1-\frac{n}{k})^n] = 0$ we conclude that (3) holds. This completes the proof of Proposition 2 □

A natural question at this point is whether the problem of checking if a relational schema is well-designed is decidable. It is not surprising that for arbitrary first-order constraints, the problem is undecidable:

Proposition 3 *The problem of verifying whether a relational schema containing first-order constraints is well-designed is undecidable.*

Proof: It is known that the problem of verifying whether a first-order sentence φ of the form $\exists\bar{x}\forall\bar{y}\,\psi(\bar{x},\bar{y})$, where $\psi(\bar{x},\bar{y})$ is an arbitrary first-order formula, is finitely satisfiable is undecidable. Denote this decision problem by $\mathcal{P}_{\exists\forall}$.

We will reduce $\mathcal{P}_{\exists\forall}$ to the complement of our problem. Let φ be a formula of the form shown above. Assume that φ is defined over a relational schema $\{R_1,\ldots,R_n\}$ and $|\bar{x}| = m > 0$, and let S be a relational schema $\{U_1, U_2, R_1, \ldots, R_n\}$, where U_1, U_2 are m-ary predicates. Furthermore, define a set of constraints Σ over S as follows:

$$\Sigma = \{\forall\bar{x}\,(U_1(\bar{x}) \leftrightarrow U_2(\bar{x})),\ \forall\bar{x}\,(U_1(\bar{x}) \rightarrow \forall\bar{y}\,\psi(\bar{x},\bar{y}))\}. \tag{6}$$

It suffices to show that $\varphi \in \mathcal{P}_{\exists\forall}$ if and only if (S,Σ) is not well-designed.

($\Rightarrow$) Assume that $\varphi \in \mathcal{P}_{\exists\forall}$ and let I_0 be an instance of $\{R_1,\ldots,R_n\}$ satisfying φ. Define $I \in inst(S,\Sigma)$ as follows: $I(R_i) = I_0(R_i)$, for every $i \in [1,n]$, and $I(U_1) = I(U_2) = \{\bar{a}\}$, where $\bar{a}$ is an m-tuple in I_0 such that $I_0 \models \forall\bar{y}\,\psi(\bar{a},\bar{y})$. Let $a \in \mathbb{N}^+ - adom(I)$ and p be an arbitrary position in $I(U_1)$. Then $I_{p\leftarrow a} \not\models \forall\bar{x}\,(U_1(\bar{x}) \leftrightarrow U_2(\bar{x}))$ and, therefore, (S,Σ) is not well-designed by Proposition 2.

($\Leftarrow$) Assume that $\varphi \notin \mathcal{P}_{\exists\forall}$. Then for every nonempty instance $I \in inst(S,\Sigma)$, $I(U_1) = I(U_2) = \emptyset$ and $I(R_i) \neq \emptyset$, for some $i \in [1,n]$. But for every position p of a value in $I(R_j)$ ($j \in [1,n]$) and

every $a \in \mathbb{N}^+ - adom(I)$, $I_{p \leftarrow a} \models \Sigma$ since $I(U_1)$ and $I(U_2)$ are empty. We conclude that (S, Σ) is well-designed by Proposition 2. $\square$

However, integrity constraints used in database schema design are most commonly *universal*, that is, of the form $\forall \bar{x}\, \psi(\bar{x})$, where $\psi(\bar{x})$ is a quantifier-free formula. FDs, MVDs and JDs are universal constraints as well as more elaborated dependencies such as equality generating dependencies and full tuple generating dependencies [1]. For universal constraints, the problem of testing if a relational schema is well-designed is decidable. In fact,

Proposition 4 *The problem of deciding whether a schema containing only universal constraints is well-designed is co-NEXPTIME-complete. Furthermore, if for a fixed m, each relation in S has at most m attributes, then the problem is Π_2^p-complete.*

To prove this proposition, first we have to prove a lemma. In this lemma we use the following terminology. A first-order constraint φ is a Σ_n-*sentence* if φ is of the form $Q_1 x_1 Q_2 x_2 \cdots Q_m x_m \psi$, where (1) $Q_i \in \{\forall, \exists\}$ $(i \in [1, m])$; (2) ψ is a quantifier-free formula; (3) the string of quantifiers $Q_1 Q_2 \cdots Q_m$ consists of n consecutive blocks, all quantifiers in the same block are the same and no adjacent blocks have the same quantifiers; and (4) the first block contains existential quantifiers. Moreover, Π_n-sentences are defined analogously, but requiring that the first block contains universal quantifiers.

Lemma 3 *Let S be a relational schema and Σ be a set of $\Sigma_n \cup \Pi_n$-sentences over S, $n \geq 1$. Then there exists a relational schema $S' \supseteq S$ and a Π_{n+1}-sentence φ over S' such that (S, Σ) is well-designed iff $\varphi \in \Sigma^+$. Moreover, $\|\varphi\|$ is $O(\|(S, \Sigma)\|^2)$.*

Proof: Assume that $S = \{R_1^{m_1}, \ldots, R_n^{m_n}\}$, where m_i is the arity of R_i $(i \in [1, n])$. Define a relational schema S' as $S \cup \{R_{i,j}^{m_i} \mid i \in [1, n] \text{ and } j \in [1, m_i]\} \cup \{U^1\}$. To define φ, first we define sentence ψ as the conjunction of the following formulas.

- $\bigvee_{i=1}^{n} \exists x_1 \cdots \exists x_{m_i} R_i(x_1, \ldots, x_{m_i})$. For some $i \in [1, n]$, relation R_i is not empty.
- $\exists x\, (U(x) \wedge \forall y\, (U(y) \rightarrow x = y))$. U contains exactly one element.
- For every $i \in [1, n]$,

$$\forall x \forall y_1 \cdots \forall y_{m_i - 1}\, (U(x) \rightarrow \bigwedge_{j=1}^{m_i} \neg R_i(y_1, \ldots, y_{j-1}, x, y_j, \ldots, y_{m_i - 1})).$$

  That is, the element contained in U is not contained in the active domain of relation R_i, for every $i \in [1, n]$.

- For every $i \in [1, n]$,

$$(\forall x_1 \cdots \forall x_{m_i}\, \neg R_i(x_1, \ldots, x_{m_i})) \rightarrow (\bigwedge_{j=1}^{m_i} \forall y_1 \cdots \forall y_{m_i}\, \neg R_{i,j}(y_1, \ldots, y_{m_i})).$$

  If R_i is empty, then $R_{i,j}$ is empty, for every $j \in [1, m_i]$.

- For every $i \in [1,n]$ and every $j \in [1,m_i]$,

$$\exists u_1 \cdots \exists u_{m_i}\, R_i(u_1,\ldots,u_{m_i}) \rightarrow$$
$$\exists x \exists x' \exists y_1 \cdots \exists y_{j-1} \exists y_{j+1} \cdots \exists y_{m_i}\, (R_i(y_1,\ldots,y_{j-1},x,y_{j+1},\ldots,y_{m_i}) \wedge$$
$$\neg R_{i,j}(y_1,\ldots,y_{j-1},x,y_{j+1},\ldots,y_{m_i}) \wedge$$
$$R_{i,j}(y_1,\ldots,y_{j-1},x',y_{j+1},\ldots,y_{m_i}) \wedge U(x') \wedge$$
$$\forall z_1 \cdots \forall z_{m_i}\, ((z_j \neq x \wedge z_j \neq x') \vee \bigvee_{k=1,k\neq j}^{m_i} z_k \neq y_k \rightarrow$$
$$(R_i(z_1,\ldots,z_{m_i}) \leftrightarrow R_{i,j}(z_1,\ldots,z_{m_i})))).$$

  If R_i is not empty, then there exists a tuple t in R_i and a tuple t' in $R_{i,j}$ such that t' is not in R_i, t is not in $R_{i,j}$ and t, t' contain exactly the same values, except for the element in the j-th column where t' contains a value that is in relation U. Furthermore, every other tuple is in R_i if and only if is in $R_{i,j}$.

Given $i \in [1,n]$ and $j \in [1,m_i]$, we denote by $\Sigma[R_i/R_{i,j}]$ the set of first-order constraints generated from Σ by replacing every occurrence of R_i by $R_{i,j}$. We define sentence φ as follows:

$$\psi \rightarrow \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m_i} \Sigma[R_i/R_{i,j}]. \tag{7}$$

Notice that ψ is a Σ_2-sentence and, therefore, φ is a Π_{n+1}-sentence, since $n \geq 1$. To finish the proof, we have to show that (S,Σ) is well-designed if and only if $\varphi \in \Sigma^+$.

($\Leftarrow$) Assume that (S,Σ) is not well-designed. Then by Proposition 2, there exists $I \in inst(S,\Sigma)$, $p \in Pos(I)$ and $a \in \mathbb{N}^+ - adom(I)$ such that $I_{p\leftarrow a} \not\models \Sigma$. Assume that p is the position of some element in the j_0-th column of R_{i_0} ($i_0 \in [1,n]$, $j_0 \in [1,m_{i_0}]$). Then we define an instance I' of S' as follows. For every $i \in [1,n]$, $I'(R_i) = I(R_i)$, $I(U) = \{a\}$ and $I'(R_{i_0,j_0}) = I_{p\leftarrow a}(R_{i_0})$. Furthermore, for every $i \in [1,n]$ and $j \in [1,m_i]$, with $i \neq i_0$ or $j \neq j_0$, if $I(R_i)$ is empty, then $I'(R_{i,j})$ is also empty, else $I'(R_{i,j})$ is constructed by replacing an arbitrary element in the j-th column of $I(R_i)$ by a. Then $I' \models \Sigma$, since $I \models \Sigma$ and $I'(R_i) = I(R_i)$ for every $i \in [1,n]$. $I' \models \psi$ since (1) $I'(R_{i_0})$ is not empty ($I(R_{i_0})$ is not empty); (2) $I'(U) = \{a\}$ and $a \notin adom(I)$; (3) for every $i \in [1,n]$, if $I'(R_i)$ is empty, then $I'(R_{i,j})$ is empty, for every $j \in [1,m_i]$; and (4) for every $i \in [1,n]$, $j \in [1,m_i]$, if $I'(R_i)$ is not empty, then $I'(R_{i,j})$ differs from $I'(R_i)$ by exactly one value, which is in U. Finally, $I' \not\models \Sigma[R_{i_0}/R_{i_0,j_0}]$, since $I'(R_{i_0,j_0}) = I_{p\leftarrow a}(R_{i_0})$ and $I_{p\leftarrow a} \not\models \Sigma$. We conclude that $I' \not\models \varphi$ and, therefore, $\varphi \notin \Sigma^+$.

($\Rightarrow$) Assume that $\varphi \notin \Sigma^+$. Then there exists a database instance I' of S', $i_0 \in [1,n]$ and $j_0 \in [1,m_{i_0}]$ such that $I' \models \Sigma$, $I' \models \psi$ and $I' \not\models \Sigma[R_{i_0}/R_{i_0,j_0}]$. We note that $I'(R_{i_0})$ is not empty (if $I'(R_{i_0})$ is empty, then $I'(R_{i_0,j_0})$ is empty ($I' \models \psi$) and, therefore, $I'(R_{i_0,j_0}) = I'(R_{i_0})$ and $I' \models \Sigma[R_{i_0}/R_{i_0,j_0}]$, since $I' \models \Sigma$, a contradiction). Define an instance I of S as follows. For every $i \in [1,n]$, $I(R_i) = I'(R_i)$. Let a be the element in $I'(U)$ and let p be the position in I of the element that has to be changed to obtain $I'(R_{i_0,j_0})$ from $I(R_{i_0})$. Then (1) I is not empty, since $I' \models \psi$; (2) $I \models \Sigma$, since $I' \models \Sigma$ and $I(R_i) = I'(R_i)$, for every $i \in [1,n]$; and (3) $I_{p\leftarrow a} \not\models \Sigma$, since $I' \not\models \Sigma[R_{i_0}/R_{i_0,j_0}]$. Given that $a \in \mathbb{N}^+ - adom(I)$, since $I' \models \psi$, by Proposition 2 we conclude that (S,Σ) is not well-designed. □

Σ_2-sentences correspond to the Schönfinkel-Bernays fragment of first-order logic. It is known that the problem of verifying if a Schönfinkel-Bernays formula has a finite model is NEXPTIME-complete

[24] and becomes Σ_2^p-complete if every relation has at most m attributes, where m is a fixed constant. Thus, from Lemma 3 we obtain the following corollary and the proof of Proposition 4.

Corollary 1 *The problem of deciding whether a schema containing only $\Sigma_1 \cup \Pi_1$-sentences is well-designed belongs to co-NEXPTIME.*

Proof of Proposition 4: We consider only the case of unbounded-arity relations, being the case of fixed-arity relations similar. The membership part of the proposition is a particular case of Corollary 1. The hardness part of the proposition follows from the following observation. If in the reduction of Proposition 3 the formula φ is of the form $\exists\bar{x}\forall\bar{y}\,\psi(\bar{x},\bar{y})$, where ψ is quantifier-free, then the set of constraints Σ defined in (6) is universal. Thus, the same reduction of Proposition 3 shows that the problem of deciding whether a Σ_2-sentence is finitely satisfiable is reducible to the problem of deciding whether a schema containing only universal constraints is well-designed. □

For specific kinds of constraints, e.g., FDs, MVDs, lower complexity bounds will follow from the results in the next section.

4.2 Justification of Relational Normal Forms

We now apply the criterion of being well-designed to various relational normal forms. We show that all of them lead to well-designed specifications, and some precisely characterize the well-designed specifications that can be obtained with a class of constraints.

We start by finding constraints that always give rise to well-designed schemas. Recall that a *typed unirelational equality generating dependency* [1] is a constraint of the form:

$$\forall\ (R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{y}),$$

where $\forall$ represents the universal closure of a formula, $\bar{x}, \bar{y} \subseteq \bar{x}_1 \cup \cdots \cup \bar{x}_m$ and there is an assignment of variables to columns such that each variable occurs only in one column and each equality atom involves a pair of variables assigned to the same column. An *extended key* is a typed unirelational equality generating dependency of the form:

$$\forall\ (R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x}_i = \bar{x}_j),$$

where $i, j \in [1, m]$. Note that every key is an extended key.

Proposition 5 *If S is a schema and Σ a set of extended keys over S, then (S, Σ) is well-designed.*

Before proving this proposition we introduce one definition that will be used in several proofs. Let $I \in inst(S, \Sigma)$, $p \in Pos(I)$, $a \in [1, k]$ and $\bar{a} \in \Omega(I, p)$. Given a substitution $\sigma : \bar{a} \rightarrow [1, k]$ and $R \in S$, we say that a tuple $t' \in \sigma(I_{(a,\bar{a})})(R)$ is *generated* by a tuple $t \in I(R)$ by means of a tuple $t^* \in I_{(a,\bar{a})}$ if $\sigma(t^*) = t'$ and t^* can be obtained from t by replacing each value in it by the element of $(a, \bar{a})$ in the same position. We say $t' \in \sigma(I_{(a,\bar{a})})(R)$ is generated by a tuple $t \in I(R)$ if it is generated by t by means of some $t^* \in I_{(a,\bar{a})}$.

Proof of Proposition 5: To prove the proposition, we now use part 2) of Proposition 2. Let $I \in inst(S, \Sigma)$, $p \in Pos(I)$ and $a \in \mathbb{N}^+ - adom(I)$. We have to show that $I_{p \leftarrow a} \models \Sigma$.

Assume to the contrary that $I_{p \leftarrow a} \not\models \Sigma$. Then there exists $R \in S$ and an extended key $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x}_i = \bar{x}_j) \in \Sigma$ such that $I_{p \leftarrow a} \not\models \forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x}_i = \bar{x}_j)$. Thus,

there exists a substitution $\rho' : \bar{x}_1 \cup \cdots \cup \bar{x}_m \to [1,k]$ such that $\rho'(\bar{x}_l) = t'_l$ and $t'_l \in I_{p \leftarrow a}(R)$, for every $l \in [1,m]$, and $t'_i \neq t'_j$. Define a substitution $\rho : \bar{x}_1 \cup \cdots \cup \bar{x}_m \to [1,k]$ as follows. Let b be the value in the p-th position of I. Then

$$\rho(x) = \begin{cases} \rho'(x) & \rho'(x) \neq a \\ b & \text{Otherwise} \end{cases}$$

Let $\rho(\bar{x}_l) = t_l$, for every $l \in [1,n]$. It is straightforward to verify that t'_1, ..., t'_n are generated from t_1, ..., t_n, respectively. Given that $I \models \Sigma$, $t_i = t_j$ and, therefore, $t'_i = t'_j$. This contradiction proves the proposition. □

Corollary 2 *A relational specification (S, Σ) in DK/NF is well-designed.*

In the rest of this section, we also denote join dependencies by first-order sentences. More precisely, a join dependency over a relation R is a first-order sentence of the form:

$$\forall\, (R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \to R(\bar{x})),$$

where $\forall$ represents the universal closure of a formula, $\bar{x} \subseteq \bar{x}_1 \cup \cdots \cup \bar{x}_m$, every variable not in $\bar{x}$ occurs in precisely one $\bar{x}_i$ ($i \in [1,m]$) and there is an assignment of variables to columns such that each variable occurs only in one column. For example, join dependency $\bowtie[AB, BC]$ over a relation $R(A,B,C)$ can be denoted by

$$\forall x \forall y \forall z \forall u_1 \forall u_2\, (R(x,y,u_1) \wedge R(u_2,y,z) \to R(x,y,z)).$$

Next, we characterize well-designed schemas with FDs and JDs.

Theorem 2 *Let Σ be a set of FDs and JDs over a relational schema S. (S, Σ) is well-designed if and only if for every $R \in S$ and every nontrivial join dependency $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \to R(\bar{x}))$ in Σ^+, there exists $M \subseteq \{1, \ldots, m\}$ such that:*

1. *$\bar{x} \subseteq \bigcup_{i \in M} \bar{x}_i$.*
2. *For every $i, j \in M$, $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \to \bar{x}_i = \bar{x}_j) \in \Sigma^+$.*

In the proof of Theorem 2 we shall use chase for FDs and JDs [21] which we now briefly review for the sake of completeness. A tableau is a set of rows with one column for each attribute in some universe U. The rows are composed of distinguished and non-distinguished variables. Each variable may appear in only one column and only one distinguished variable may appear in one column. Let the non-distinguished variables be x_1, ..., x_m. The chase of T with respect to a set Σ of FDs and JDs is based on the successive application of the following two rules:

FD rule: Let σ be a functional dependency in Σ of the form $X \to A$, where A is a single attribute, and let $u, v \in T$ be such that $u[X] = v[X]$ and $u[A] \neq v[A]$. The result of applying the FD σ to T is a new tableau T' defined as follows. If one of the variables $u[A], v[A]$ is distinguished, then all the occurrences of the other one are renamed to that variable. If both are non-distinguished, then all the occurrences of the variable with the larger subscript are renamed to the variable with the smaller subscript.

JD rule: Let σ be a join dependency of the form $\bowtie[X_1, \ldots, X_n]$ and let u be a tuple not in T. If there are $u_1, \ldots, u_n \in T$ such that $u_i[X_i] = u[X_i]$ for every $i \in [1,n]$, then the result of applying the JD σ over T is $T \cup \{u\}$.

A chasing sequence of T by Σ is a sequence of tableaux $T = T_0, T_1, T_2, \ldots$, such that for each $i \geq 0$, T_{i+1} is the result of applying some dependency in Σ to T_i. It is known that any such sequence terminate and the resulting tableau does not depend on a particular sequence [21]; we denote this tableau by $\mathit{Chase}(T, \Sigma)$.

Every application of either the "FD rule" or the "JD rule" naturally defines a substitution of variables by variables (in the latter, this substitution is the identity). The substitution defined by the chase is obtained as the composition of the substitutions for each step of the chase. This substitution enables us to map each original variable (tuple) in T to a variable (tuple) in $\mathit{Chase}(T, \Sigma)$.

Given a set of FDs and JDs $\Sigma \cup \{\sigma\}$, it was shown in [21] that the chase can be used for checking whether $\Sigma \models \sigma$. The idea is to construct a tableau T_σ, compute $\mathit{Chase}(T_\sigma, \Sigma)$ and verify whether some condition is satisfied. If σ is an FD $X \rightarrow A$, then T_σ has two rows: one contains only distinguished variables, and the other one contains distinguished variables in all the X-columns and non-distinguished variables elsewhere. Then $\Sigma \models \sigma$ iff $\mathit{Chase}(T_\sigma, \Sigma)$ has only one distinguished variable in the A-column [21]. Moreover, if σ is a JD $\bowtie[X_1, \ldots, X_n]$, then T_σ has n rows. For every $i \in [1, n]$, the i-th row contains distinguished variables in the X_i-columns and non-distinguished variables in the remaining columns. Furthermore, every non-distinguished variable in T_σ appears exactly once. Then $\Sigma \models \sigma$ iff $\mathit{Chase}(T_\sigma, \Sigma)$ contains a row of all distinguished variables [21].

Chase, and all the results shown above, can be generalized in a natural manner to the case of more expressive constraints like typed equality generating dependencies (see [1]).

We now move to the proof of Theorem 2. We need two lemmas first.

Lemma 4 *Let Σ be a set of FDs and JDs over a relational schema S and $R \in S$. Assume that Σ contains a JD $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x}))$ such that $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{x}_i) \notin \Sigma^+$, for every $i \in [1, m]$. Then there exists $I \in \mathit{inst}(S, \Sigma)$ and $p \in \mathit{Pos}(I)$ such that $\textsc{Inf}_I(p \mid \Sigma) < 1$.*

Proof: Let T be a tableau containing tuples $\{\bar{x}_1, \ldots, \bar{x}_m\}$, and let $\bar{x}$ be the distinguished variables. Let ρ be a one-to-one function with the domain $\bar{x}_1 \cup \cdots \cup \bar{x}_m$ and the range contained in $\mathbb{N}^+$. Define $I = \rho(\mathit{Chase}(T, \Sigma))$. Assume that θ is the composition of the substitutions used in the chase. Let $t_j = \rho(\theta(\bar{x}_j))$, for every $j \in [1, m]$, and $t = \rho(\theta(\bar{x}))$. Given that $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{x}_i) \notin \Sigma^+$, for every $i \in [1, m]$, we conclude that $t \neq t_j$, for every $j \in [1, m]$. Let $A \in \mathit{sort}(R)$, p be the position of $t[A]$ in I and k such that $\mathit{adom}(I) \subseteq [1, k]$. Since $I \models \Sigma$ and I contains t_1, ..., t_m, the JD $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x})) \in \Sigma$ implies that I must contain t. Thus, changing any value in t generates an instance that does not satisfy Σ. Hence, for every $a \in [1, k] - \{t[A]\}$, $P(a \mid \bar{a}_0) = 0$, where $\bar{a}_0$ is the tuple in $\Omega(I, p)$ containing no variables. Applying Lemma 2 we conclude that $H(\mathcal{B}^k_\Sigma(I, p) \mid \mathcal{A}(I, p))/\log k < c$ for some constant $c < 1$, for all sufficiently large k, and thus by Lemma 1, $\textsc{Inf}_I(p \mid \Sigma) = \lim_{k \to \infty} \textsc{Inf}^k_I(p \mid \Sigma)/\log k < 1$. $\square$

Given a set Σ of FDs and JDs over a relational schema S and a JD $\varphi \in \Sigma$ of the form $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x}))$, define an equivalence relation $\sim_\varphi$ on tuples of variables as follows. For every $i, j \in [1, m]$, $\bar{x}_i \sim_\varphi \bar{x}_j$ if $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x}_i = \bar{x}_j) \in \Sigma^+$. Let $[i]_\varphi$ be the equivalence class of $\bar{x}_i$, for every $i \in [1, m]$, and let $\mathit{var}([i]_\varphi)$ be the set of variables contained in all the tuples $\bar{x}_j \in [i]_\varphi$.

Lemma 5 *Let Σ be a set of FDs and JDs over a relational schema S and $R \in S$. Assume that Σ contains a JD φ of the form $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x}))$ such that $\bar{x} \not\subseteq \mathit{var}([i]_\varphi)$, for every $i \in [1, m]$. Then there exists $I \in \mathit{inst}(S, \Sigma)$ and $p \in \mathit{Pos}(I)$ such that $\textsc{Inf}_I(p \mid \Sigma) < 1$.*

Proof: If $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{x}_i) \notin \Sigma^+$, for every $i \in [1, m]$, then by Lemma 4 there exists $I \in \mathit{inst}(S, \Sigma)$ and $p \in \mathit{Pos}(I)$ such that $\textsc{Inf}_I(p \mid \Sigma) < 1$. Thus, we may assume that there

exists $i \in [1,m]$ such that $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{x}_i) \in \Sigma^+$. By the hypothesis, there exists $l \in [1, |\bar{x}|]$ and a variable x in the l-th column of $\bar{x}$ such that $x \notin var([i]_\varphi)$. Let u be the variable in the l-th column of $\bar{x}_i$ and U_i the set of variables in the l-column of all the tuples $\bar{x}_j$ ($j \in [1,m]$) such that $\bar{x}_i \sim_\varphi \bar{x}_j$.

Let T be a tableau $\{\bar{x}_1, \ldots, \bar{x}_m\}$, with $\bar{x}_i$ as distinguished variables. In $Chase(T, \Sigma)$, all the tuples in the equivalence class of $\bar{x}_i$ (and no other) are identified with this tuple. Denote the l-th component of tuple $\bar{x}_j$ by $\bar{x}_j^l$ (and similarly for other tuples).

Let ρ be a one-to-one function with the domain $\bar{x}_1 \cup \cdots \cup \bar{x}_m$ and the range contained in $\mathbb{N}^+$ and $I = \rho(Chase(T, \Sigma))$. Assume that θ is the composition of the substitutions used in the chase. Let $t_j = \rho(\theta(\bar{x}_j))$ be a tuple in I, for every $j \in [1,m]$. Note that $\rho(\theta(\bar{x}_i)) = \rho(\bar{x}_i)$ since $\bar{x}_i$ is a tuple of distinguished variables. Additionally, since I satisfies $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow \bar{x} = \bar{x}_i)$, it must be the case that $\rho(\theta(\bar{x})) = \rho(\bar{x}_i)$.

Let p be the position in I of t_i^l. The value in this position is $\rho(u)$. We will show that for every $a \in [1,k] - \{\rho(u)\}$, $P(a \mid \bar{a}_0) = 0$, where $\bar{a}_0$ is a tuple in $\Omega(I,p)$ containing no variables.

Denote by t'_j the tuple of $I_{(a,\bar{a}_0)}$ that corresponds to t_j in I. Note that $t'_j = t_j$ for all j such that $\bar{x}_j$ is not in $[i]_\varphi$. When $\bar{x}_j$ is in $[i]_\varphi$, t'_j differs from t_j only in that the value in its l-th column is a rather than $\rho(u)$. Assume that $I_{(a,\bar{a}_0)}$ satisfies Σ. Then it satisfies, in particular, $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x}))$. Recall that in this JD, every variable not in $\bar{x}$ occurs in a unique $\bar{x}_j$. We give a substitution from the variable tuples $\bar{x}_1, \ldots, \bar{x}_m$ to the tuples $t'_1, \ldots, t'_m$, respectively. Let $\rho' : \bar{x}_1 \cup \cdots \cup \bar{x}_m \rightarrow [1,k]$ be a substitution defined as follows. For every $y \in \bar{x}_1 \cup \cdots \cup \bar{x}_m$,

$$\rho'(y) = \begin{cases} \rho(\theta(y)) & \text{if } \ y \notin U_i \\ a & \text{otherwise.} \end{cases}$$

We claim that for every $j \in [1,m]$, $\rho'(\bar{x}_j) = t'_j$. Clearly, we only need to consider the l-th column. Indeed, if $\bar{x}_j$ is in $[i]_\varphi$, then t'_j is t_j, except in the l-column, where t_j contains the value a, since $\bar{x}_j^l$ is in U_i. Thus, $\rho'(\bar{x}_j) = t'_j$. If $\bar{x}_j$ is not in $[i]_\varphi$, then $\bar{x}_j^l$ is either x, or a variable that occurs only in $\bar{x}_j$. In either case, it is not in U_i. Thus, $\rho'(\bar{x}_j) = t'_j$. Since $I_{(a,\bar{a}_0)}$ is assumed to satisfy JD $\forall(R(\bar{x}_1) \wedge \cdots \wedge R(\bar{x}_m) \rightarrow R(\bar{x}))$, it must contain $\rho'(\bar{x})$. However, since x is not in U_i, $\rho'(\bar{x}) = \rho(\theta(\bar{x})) = \rho(\bar{x}_i) = t_i$ in I, which is not in $I_{(a,\bar{a}_0)}$, a contradiction.

We conclude that for every $a \in [1,k] - \{\rho(u)\}$, $P(a \mid \bar{a}_0) = 0$. Hence, by Lemma 2, $\textsc{Inf}_I^k(p \mid \Sigma)/\log k < c$ for some constant $c < 1$, for all sufficiently large k, and then by Lemma 1, $\textsc{Inf}_I(p \mid \Sigma) = \lim_{k\to\infty} \textsc{Inf}_I^k(p \mid \Sigma)/\log k < 1$. This proves the lemma. □

Theorem 2 is a corollary of Proposition 5 and Lemma 5. We note that this theorem justifies various normal forms proposed for JDs and FDs [14, 29].

Corollary 3 *Let Σ be a set of FDs and JDs over a relational schema S. If (S, Σ) is in PJ/NF or 5NFR, then it is well-designed.*

However, neither of these normal forms characterizes precisely the notion of being well-defined:

Proposition 6 *There exists a schema S and a set of JDs and FDs Σ such that (S, Σ) is well-designed, but it violates all of the following: DK/NF, PJ/NF, 5NFR.*

Proof: Let $S = \{R(A,B,C)\}$ and $\Sigma = \{AB \rightarrow C,\ AC \rightarrow B,\ \bowtie[AB, AC, BC]\}$. This specification is not in DK/NF and PJ/NF since the set of keys implied by Σ is $\{AB \rightarrow ABC,\ AC \rightarrow ABC,$

$ABC \rightarrow ABC\}$ and this set does not imply $\bowtie[AB, AC, BC]$. Furthermore, this specification is not in 5NFR since $\bowtie[AB, AC, BC]$ is a strong-reduced join dependency and BC is not a key in Σ.

Join dependency $\bowtie[AB, AC, BC]$ corresponds to the following first order sentence:

$$\forall x \forall y \forall z \forall u_1 \forall u_2 \forall u_3 \, (R(x, y, u_1) \wedge R(x, u_2, z) \ \wedge R(u_3, y, z) \rightarrow R(x, y, z)).$$

From Theorem 2, we conclude that (S, Σ) is well designed since Σ implies the sentence

$$\forall x \forall y \forall z \forall u_1 \forall u_2 \forall u_3 (R(x, y, u_1) \wedge R(x, u_2, z) \ \wedge R(u_3, y, z) \rightarrow y = u_2 \wedge z = u_1).$$

and $(x, y, z) \subseteq (x, y, u_1) \cup (x, u_2, z)$. □

By restricting Theorem 2 to the case of specifications containing only FDs and MVDs (or only FDs), we obtain the equivalence between well-designed databases and 4NF (respectively, BCNF).

Theorem 3 *Let Σ be a set of integrity constraints over a relational schema S.*

1. *If Σ contains only FDs and MVDs, then (S, Σ) is well-designed if and only if it is in 4NF.*
2. *If Σ contains only FDs, then (S, Σ) is well-designed if and only if it is in BCNF.*

5 Normalizing XML data

In this section we give an overview of the XML normal form called XNF, and show that the notion of being well-designed straightforwardly extends from relations to XML. Furthermore, if all constraints are specified as functional dependencies, this notion precisely characterizes XNF.

5.1 Overview of XML Constraints and Normalization

5.1.1 DTDs and XML trees

We shall use a somewhat simplified model of XML trees in order to keep the notation simple. We assume a countably infinite set of labels L, a countably infinite set of attributes A (we shall use the notation $@l_1, @l_2$, etc for attributes to distinguish them from labels), and a countably infinite set V of values of attributes. Furthermore, we do not consider `PCDATA` elements in XML trees since they can always be represented by attributes.

A DTD (Document Type Definition) D is a 4-tuple (L_0, P, R, r) where L_0 is a finite subset of L, P is a set of rules $a \rightarrow P_a$ for each $a \in L_0$, where P_a is a regular expression over $L_0 - \{r\}$, R assigns to each $a \in L_0$ a finite subset of A (possibly empty; $R(a)$ is the set of attributes of a), and $r \in L_0$ (the root).

Example 5: The DTD below is a part of DBLP [11] that stores conference data.

```
<!ELEMENT db (conf*)>
<!ELEMENT conf (issue+)>
  <!ATTLIST conf
            title CDATA #REQUIRED>
<!ELEMENT issue (inproceedings+)>
<!ELEMENT inproceedings EMPTY>
  <!ATTLIST inproceedings
            author CDATA #REQUIRED
            title CDATA #REQUIRED
            pages CDATA #REQUIRED
            year CDATA #REQUIRED>
```

This DTD is represented as (L_0, P, R, r), where $r = db$, $L_0 = \{db, conf, issue, inproceedings\}$, $P = \{db \rightarrow conf^*, conf \rightarrow issue^+, issue \rightarrow inproceedings^+, inproceedings \rightarrow \epsilon\}$, $R(conf) = \{@title\}$, $R(inproceedings) = \{@author, @title, @pages, @year\}$ and $R(db) = R(issue) = \emptyset$. □

An XML tree is a finite rooted directed tree $T = (N, E)$ where N is the set of nodes and E is the set of edges, together with the labeling function $\lambda : N \rightarrow L$ and partial attribute value functions $\rho_{@l} : N \rightarrow V$ for each $@l \in A$. We furthermore assume that for every node x in N, its children $x_1, \ldots, x_n$ are ordered and $\rho_{@l}(x)$ is defined for a finite set of attributes $@l$. We say that T conforms to DTD $D = (L_0, P, R, r)$, written as $T \models D$, if the root of T is labeled r, for every $x \in N$ with $\lambda(x) = a$, the word $\lambda(x_1) \cdots \lambda(x_n)$ that consists of the labels of its children belongs to the language denoted by P_a, and for every $x \in N$ with $\lambda(x) = a$, $@l \in R(a)$ if and only if the function $\rho_{@l}$ is defined on x (and thus provides the value of attribute $@l$).

5.1.2 Functional Dependencies for XML

To present a functional dependency language for XML we need to introduce some terminology. Recall that L and A are countably infinite sets of labels and attributes, respectively. Then an *element path* q is a word in L^*, and an *attribute path* is a word of the form $q.@l$, where $q \in L^*$ and $@l \in A$. An element path q is consistent with a DTD D if there is a tree $T \models D$ that contains a node reachable by q (in particular, all such paths must have r as the first letter); if in addition the nodes reachable by q have attribute $@l$, then the attribute path $q.@l$ is consistent with D. The set of all paths (element or attribute) consistent with D is denoted by $paths(D)$. This set is finite for a non-recursive D and infinite if D is recursive.

A *functional dependency* over DTD D [3] is an expression of the form $\{q_1, \ldots, q_n\} \rightarrow q$, where $n \geq 1$ and $q, q_1, \ldots, q_n \in paths(D)$. To define the notion of satisfaction for FDs, we use a relational representation of XML trees from [3]. Given $T \models D$, a *tree tuple* in T is a mapping $t : paths(D) \rightarrow N \cup V \cup \{\perp\}$ such that if q is an element path whose last letter is a and $t(q) \neq \perp$, then

- $t(q) \in N$ and its label, $\lambda(t(q))$, is a;
- if q' is a prefix of q, then $t(q') \neq \perp$ and the node $t(q')$ lies on the path from the root to $t(q)$ in T;
- if $@l$ is defined for $t(q)$ and its value is $v \in V$, then $t(q.@l) = v$.

Intuitively, a tree tuple assigns nodes or attribute values or nulls ($\perp$) to paths in a consistent manner. A tree tuple is maximal if it cannot be extended to another one by changing some nulls to values from $N \cup V$. The set of maximal tree tuples is denoted by $tuples_D(T)$. Now we say that FD $\{q_1, \ldots, q_n\} \rightarrow q$ is true in T if for any $t_1, t_2 \in tuples_D(T)$, whenever $t_1(q_i) = t_2(q_i) \neq \perp$ for all $i \leq n$, then $t_1(q) = t_2(q)$ holds.

Example 6: Let D be the DTD from Example 5. Among the set Σ of FDs over this DTD are:

$$db.conf.@title \rightarrow db.conf,$$
$$db.conf.issue \rightarrow db.conf.issue.inproceedings.@year.$$

The first functional dependency specifies that two distinct conferences must have distinct titles. The second one specifies that any two *inproceedings* children of the same *issue* must have the same value of *@year*. □

5.1.3 XNF: An XML Normal Form.

Suppose we are given a DTD D and a set Σ of FDs over D. The set of all FDs implied by (D, Σ) is denoted by $(D, \Sigma)^+$, this is, $(D, \Sigma)^+$ is the set of all FD $X \to Y$ over D such that for every XML tree T conforming to D and satisfying Σ, $T \models X \to Y$. An FD is called *trivial* if it belongs to $(D, \emptyset)^+$, that is, it is implied by the DTD alone. For example, $q \to r$, where r is the root, or $q \to q.@l$, are trivial FDs.

We say that (D, Σ) is in *XML Normal Form (XNF)* [3] if for any nontrivial FD $X \to q.@l$ in $(D, \Sigma)^+$, the FD $X \to q$ is in $(D, \Sigma)^+$ as well. Intuitively, a violation of XNF means that there is some redundancy in the document: we may have many nodes reachable by path q but all of them will have the same value of attribute $@l$ (provided they agree on X).

Example 7: The DBLP example 5 seen earlier may contain redundant information: year is stored multiple times for the same issue of a conference. This XML specification is *not* in XNF since

$$db.conf.issue \;\to\; db.conf.issue.inproceedings$$

is not in $(D, \Sigma)^+$. This suggests making *@year* an attribute of *issue*, and indeed, such a revised specification can easily be shown to be in XNF. □

5.2 Well-designed XML data

We do not need to introduce a new notion of being well-designed specifically for XML: the definition that we formulated in Section 4 for relational data will apply. We only have to define the notion of positions in a tree, and then reuse the relational definition. For relational databases, positions correspond to the "shape" of relations, and each position contains a value. Likewise, for XML, positions will correspond to the shape (that is more complex, since documents are modeled as trees), and they must have values associated with them. Consequently, we formally define the set of positions $Pos(T)$ in a tree $T = (N, E)$ as $\{(x, @l) \mid x \in N,\ @l \in R(\lambda(x))\}$. As before, we assume that there is an enumeration of positions (a bijection between $Pos(T)$ and $\{1, \ldots, n\}$ where $n = |Pos(T)|$) and we shall associate positions with their numbers in the enumeration. We define $adom(T)$ as the set of all values of attributes in T and $T_{p \leftarrow a}$ as an XML tree constructed from T by replacing the value in position p by a.

As in the relational case, we take the domain of values V of the attributes to be $\mathbb{N}^+$. Let Σ be a set of FDs over a DTD D and $k > 0$. Define $inst(D, \Sigma)$ as the set of all XML trees that conform to D and satisfy Σ and $inst_k(D, \Sigma)$ as its restriction to trees T with $adom(T) \subseteq [1, k]$. Now fix $T \in inst_k(D, \Sigma)$ and $p \in Pos(T)$. With the above definitions, we define the probability spaces $\mathcal{A}(T, p)$ and $\mathcal{B}^k_\Sigma(T, p)$ exactly as we defined $\mathcal{A}(I, p)$ and $\mathcal{B}^k_\Sigma(I, p)$ for a relational instance I. That is, $\Omega(T, p)$ is the set of all tuples $\bar{a}$ of the form $(a_1, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n)$ such that every a_i is either a variable, or the value T has in the corresponding position, $SAT^k_\Sigma(T_{(a,\bar{a})})$ as the set of all possible ways to assign values from $[1, k]$ to variables in $\bar{a}$ that result in a tree satisfying Σ, and the rest of the definition repeats the relational case one verbatim, substituting T for I.

We use the above definitions to define $\textsc{Inf}^k_T(p \mid \Sigma)$ as the entropy of $\mathcal{B}^k_\Sigma(T, p)$ given $\mathcal{A}(T, p)$:

$$\textsc{Inf}^k_T(p \mid \Sigma) \;\stackrel{\text{def}}{=}\; H(\mathcal{B}^k_\Sigma(T, p) \mid \mathcal{A}(T, p)) \;.$$

As in the relational case, we can show that the limit

$$\lim_{k \to \infty} \frac{\textsc{Inf}^k_T(p \mid \Sigma)}{\log k}$$

exists, and we denote it by $\textsc{Inf}_T(p \mid \Sigma)$. Following the relational case, we introduce

Definition 3 *An XML specification* (D, Σ) *is* well-designed *if for every* $T \in inst(D, \Sigma)$ *and every* $p \in Pos(T)$, $\textsc{Inf}_T(p \mid \Sigma) = 1$.

Note that the information-theoretic definition of well-designed schema presented in Section 4 for relational data proved to be extremely robust, as it extended straightforwardly to a different data model: we only needed a new definition of $Pos(T)$ to use in place of $Pos(I)$, and $Pos(T)$ is simply an enumeration of all the places in a document where attribute values occur. As in the relational case, it is possible to show that well-designed XML and XNF coincide. Furthermore, it is also possible to establish a useful structural criterion for $\textsc{Inf}_T(p \mid \Sigma) = 1$, namely that an XML specification (D, Σ) is well-designed if and only if one position of an arbitrary $T \in inst(D, \Sigma)$ can always be assigned a fresh value.

Theorem 4 *Let* D *be a DTD and* Σ *a set of FDs over* D. *Then the following are equivalent.*

1) (D, Σ) *is well-designed.*

2) (D, Σ) *is in XNF.*

3) For every $T \in inst(D, \Sigma)$, $p \in Pos(T)$ *and* $a \in \mathbb{N}^+ - adom(T)$, $T_{p \leftarrow a} \models \Sigma$.

The proof of the theorem follows rather closely the proof of Proposition 2, by replacing relational concepts by their XML counterparts.

Proof of Theorem 4: We will prove the chain of implications 1) $\Rightarrow$ 2) $\Rightarrow$ 3) $\Rightarrow$ 1).

1) $\Rightarrow$ 2) Assume that (D, Σ) is not in XNF. We will show that there exists $T \in inst(D, \Sigma)$ and $p \in Pos(T)$ such that $\textsc{Inf}_T(p \mid \Sigma) < 1$.

Given that (D, Σ) is not in XNF, there exists a nontrivial FD $X \rightarrow q.@l \in (D, \Sigma)^+$ such that $X \rightarrow q \notin (D, \Sigma)^+$. Thus, there is $T \in inst(D, \Sigma)$ containing tree tuples t_1, t_2 such that $t_1(q') = t_2(q')$ and $t_1(q') \neq \perp$, for every $q' \in X$, and $t_1(q) \neq t_2(q)$. We may assume that $t_1(q) \neq \perp$ and $t_2(q) \neq \perp$ (if $t_1(q) = \perp$ or $t_2(q) = \perp$, then $t_1(q.@l) \neq t_2(q.@l)$, which would contradict $T \models \Sigma$). Let $x = t_1(q)$, p be the position of $(x, @l)$ in T and $a = t_1(q.@l)$. Let $\bar{a}_0$ be the vector in $\Omega(T, p)$ containing no variables. Given that $t_1(q) \neq t_2(q)$ and none of these values is $\perp$, for every $b \in [1, k] - \{a\}$, $T_{(b, \bar{a}_0)} \not\models \Sigma$. Thus, for every $b \in [1, k] - \{a\}$, $P(b \mid \bar{a}_0) = 0$. Now a straightforward application of Lemma 2 implies

$$\textsc{Inf}_T(p \mid \Sigma) \;=\; \lim_{k \to \infty} \textsc{Inf}_T^k(p \mid \Sigma) / \log k \;<\; 1.$$

This concludes the proof.

2) $\Rightarrow$ 3) Let (D, Σ) be an XML specification in XNF, $T \in inst(D, \Sigma)$, $p \in Pos(T)$ and $a \in \mathbb{N}^+ - adom(T)$. We prove that $T_{p \leftarrow a} \models \Sigma$.

Assume, to the contrary, that $T_{p \leftarrow a} \not\models \Sigma$. Then there exists a FD $X \rightarrow q \in \Sigma$ such that $T_{p \leftarrow a} \not\models X \rightarrow q$. Thus, there exists $t_1', t_2' \in tuples_D(T_{p \leftarrow a})$ such that $t_1'(q') = t_2'(q')$ and $t_1'(q') \neq \perp$, for every $q' \in X$, and $t_1'(q) \neq t_2'(q)$. Assume that these tuples were generated from tuples $t_1, t_2 \in tuples_D(T)$. Given that $a \in \mathbb{N}^+ - adom(T)$, $t_1(q') = t_2(q')$ and $t_1(q') \neq \perp$, for every $q' \in X$, and, therefore, $t_1(q) = t_2(q)$, since $T \models \Sigma$. If q is an element path, then $t_1'(q) = t_1(q)$ and $t_2'(q) = t_2(q)$, since $T_{p \leftarrow a}$ is constructed from T by modifying only the values of attributes. Thus, $t_1'(q) = t_2'(q)$, a contradiction. Assume that q is an attribute path of the form $q_1.@l$. In this case, $X \rightarrow q_1.@l$ is a nontrivial FD in Σ and, therefore, $X \rightarrow q_1 \in (D, \Sigma)^+$, since (D, Σ) is in XNF. We conclude

that $t_1(q_1) = t_2(q_1)$. Given that q_1 is an element path, as in the previous case we conclude that $t'_1(q_1) = t'_2(q_1)$. Hence, $t'_1(q_1.@l) = t'_2(q_1.@l)$, again a contradiction.

3) $\Rightarrow$ 1) Let $T \in inst(D, \Sigma)$ and $p \in Pos(T)$. We have to prove that $\textsc{Inf}_T(p \mid \Sigma) = 1$. To show this, it suffices to prove that

$$\lim_{k\to\infty} \frac{\textsc{Inf}^k_T(p \mid \Sigma)}{\log k} \quad \geq \quad 1. \tag{8}$$

Let $n = |Pos(T)|$ and $k > 2n$ such that $T \in inst_k(D, \Sigma)$. If $\bar{a} \in \Omega(T, p)$ and $var(\bar{a})$ is the set of variables mentioned in $\bar{a}$, then for every $a \in [1, k] - adom(T)$,

$$|SAT^k_\Sigma(T_{(a,\bar{a})})| \quad \geq \quad (k - 2n)^{|var(\bar{a})|}$$

since by hypothesis one can replace values in positions of $\bar{a}$ one by one, provided that each position gets a fresh value. Thus, given that $\sum_{b\in[1,k]} |SAT^k_\Sigma(T_{(b,\bar{a})})| \leq k^{|var(\bar{a})|+1}$, for every $a \in [1, k] - adom(T)$ and every $\bar{a} \in \Omega(T, p)$, we have:

$$P(a \mid \bar{a}) \;\geq\; \frac{(k - 2n)^{|var(\bar{a})|}}{k^{|var(\bar{a})|+1}} \;=\; \frac{1}{k}\,(1 - \frac{2n}{k})^{|var(\bar{a})|}. \tag{9}$$

Functional dependencies are generic constraints. Thus, for every $a, b \in [1, k] - adom(T)$ and every $\bar{a} \in \Omega(T, p)$, $P(a \mid \bar{a}) = P(b \mid \bar{a})$. Hence, for every $a \in [1, k] - adom(T)$ and every $\bar{a} \in \Omega(T, p)$:

$$P(a \mid \bar{a}) \;\leq\; \frac{1}{k - |adom(T)|} \;\leq\; \frac{1}{k - n}. \tag{10}$$

In order to prove (8), we need to establish a lower bound for $\textsc{Inf}^k_T(p \mid \Sigma)$. We do this by using (9) and (10) as follows: Given the term $P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})}$, we use (9) and (10) to replace $P(a \mid \bar{a})$ and $\log \frac{1}{P(a|\bar{a})}$ by smaller terms, respectively. More precisely,

$$\begin{aligned}
\textsc{Inf}^k_T(p \mid \Sigma) &= \sum_{\bar{a}\in\Omega(T,p)} \left(P(\bar{a}) \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \right) \\
&\geq \frac{1}{2^{n-1}} \sum_{a\in[1,k]-adom(T)} \sum_{\bar{a}\in\Omega(T,p)} \frac{1}{k}\,(1 - \frac{2n}{k})^{|var(\bar{a})|} \log(k - n) \\
&= \frac{1}{2^{n-1}}\;\log(k - n)\;\frac{1}{k} \sum_{a\in[1,k]-adom(I)} \sum_{i=0}^{n-1} \binom{n-1}{i} (1 - \frac{2n}{k})^i \\
&= \frac{1}{2^{n-1}}\;\log(k - n)\;\frac{1}{k} \sum_{a\in[1,k]-adom(I)} ((1 - \frac{2n}{k}) + 1)^{n-1} \\
&\geq \frac{1}{2^{n-1}}\;\log(k - n)\;\frac{1}{k}\;(k - n)\;(2 - \frac{2n}{k})^{n-1} \\
&= \frac{1}{2^{n-1}}\;\log(k - n)\;(1 - \frac{n}{k})\;2^{n-1}\;(1 - \frac{n}{k})^{n-1} \\
&= \log(k - n)\;(1 - \frac{n}{k})^n.
\end{aligned}$$

Therefore, $\frac{\textsc{Inf}^k_T(p|\Sigma)}{\log k} \geq \frac{\log(k-n)}{\log k}(1 - \frac{n}{k})^n$. Since $\lim_{k\to\infty} \frac{\log(k-n)}{\log k}(1 - \frac{n}{k})^n = 1$, (8) follows. This concludes the proof. □

The theory of XML constraints and normal forms is not nearly as advanced as its relational counterparts, but we demonstrated here that the definition of well-designed schemas works well for the existing normal form based on FDs; thus, it can be used to test other design criteria for XML when they are proposed.

6 Normalization algorithms

We now show how the information-theoretic measure of Section 4 can be used for reasoning about normalization algorithms at the instance level. For this section, we assume that Σ is a set of FDs, both for the relational and the XML cases. The results shown here state that after each step of a decomposition algorithm, the amount of information in each position does not decrease.

6.1 Relational Databases

Let I' be the result of applying one step of a normalization algorithm to I. In order to compare the amount of information in these instances, we need to show how to associate positions in I and I'. Since we only consider here functional dependencies, we deal with BCNF, and standard BCNF decomposition algorithms use steps of the following kind: pick a relation R with the set of attributes W, and let W be the disjoint union of X, Y, Z, such that $X \to Y \in \Sigma^+$. Then an instance $I = I(R)$ of R gets decomposed into $I_{XY} = \pi_{XY}(I)$ and $I_{XZ} = \pi_{XZ}(I)$, with the sets of FDs Σ_{XY} and Σ_{XZ}, where Σ_U stands for $\{C \to D \in \Sigma^+ \mid CD \subseteq U \subseteq W\}$. This decomposition gives rise to two partial maps $\pi_{XY} : Pos(I) \to Pos(I_{XY})$ and $\pi_{XZ} : Pos(I) \to Pos(I_{XZ})$. If p is the position of $t[A]$ for some $A \in XY$, then $\pi_{XY}(p)$ is defined, and equals the position of $\pi_{XY}(t)[A]$ in I_{XY}; the mapping π_{XZ} is defined analogously. Note that π_{XY} and π_{XZ} can map different positions in I to the same position of I_{XY} or I_{XZ}.

We now show that the amount of information in each position does not decrease in the normalization process.

Theorem 5 *Let (X, Y, Z) partition the attributes of R, and let $X \to Y \in \Sigma^+$. Let $I \in inst(R, \Sigma)$ and $p \in Pos(I)$. If U is either XY or XZ and π_U is defined on p, then $\textsc{Inf}_I(p \mid \Sigma) \leq \textsc{Inf}_{I_U}(\pi_U(p) \mid \Sigma_U)$.*

To prove this theorem, first we need to prove two lemmas.

Lemma 6 *Let Σ be a set of FDs over a relational schema S, $I \in inst(S, \Sigma)$, $p \in Pos(I)$ and $\bar{a} \in \Omega(I, p)$. Then $\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})}$ is either 0 or 1.*

Proof: Given in Appendix A.2. □

Let R be a relation schema such that $sort(R) = X \cup Y \cup Z$, where X, Y and Z are nonempty pairwise disjoint sets of attributes. Let Σ be a set of FDs over R and $I \in inst(R, \Sigma)$. Assume that $X \to Y \in \Sigma^+$. Define R' as a relation schema such that $sort(R') = X \cup Y$, $\Sigma' = \Sigma_{XY}$, and let I' be $\pi_{XY}(I)$. Note that $I' \in inst(R', \Sigma')$. We use Lemma 6 to show the following.

Lemma 7 *Let $t_0 \in I$, $t'_0 = \pi_{XY}(t_0)$ and $A \in X \cup Y$. If $t_0[A]$ is the p-th element in I and $t'_0[A]$ is the p'-th element in I', then $\textsc{Inf}_I(p \mid \Sigma) \leq \textsc{Inf}_{I'}(p' \mid \Sigma')$.*

Proof: Assume that $\|I\| = n$, $X \cup Y = \{A_1, \ldots, A_m\}$ and $\{t[X] \mid t \in I\}$ contains l tuples $\{\bar{c}_1, \ldots, \bar{c}_l\}$. For every $i \in [1, l]$, choose a tuple $t_i \in I$ such that $t_i[X] = \bar{c}_i$. Without loss of generality, assume that $t_0 = t_l$, $A = A_m$ and $t_i[A_j]$ is the $((i-1)m + j)$-th element in I. Thus, $t_1[A_1]$ is the first element in I, $t_1[A_m]$ is the m-th element in I and $t_l[A_m]$ is the lm-th element in I. We note that $p = lm$.

For every $\bar{a} = (a_1, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n) \in \Omega(I, p)$, define $\bar{a}^* = (a_1, \ldots, a_{p-1}, v_{p+1}, \ldots, v_n)$, that is, $\bar{a}^*$ is generated from $\bar{a}$ by replacing each a_i ($i \in [p+1, n]$) by a variable. Furthermore, define $\Omega^*(I, p)$ as $\{\bar{a} \in \Omega(I, p) \mid$ for every $i \in [p+1, n]$, a_i is a variable$\}$. It is easy to see that if $\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})} = 1$, then $\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}^*) \log \frac{1}{P(a|\bar{a}^*)} = 1$. Thus, by Lemma 6, for every $\bar{a} \in \Omega(I, p)$:

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \quad \leq \quad \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}^*) \log \frac{1}{P(a \mid \bar{a}^*)}.$$

Therefore,

$$\begin{aligned}
\textsc{Inf}_I(p \mid \Sigma) &= \lim_{k\to\infty} \frac{1}{\log k} \sum_{\bar{a}\in\Omega(I,p)} \frac{1}{2^{n-1}} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&= \frac{1}{2^{n-1}} \sum_{\bar{a}\in\Omega(I,p)} \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&\leq \frac{1}{2^{n-1}} 2^{n-p} \sum_{\bar{a}\in\Omega^*(I,p)} \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&= \frac{1}{2^{p-1}} \sum_{\bar{a}\in\Omega^*(I,p)} \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})}. \qquad (11)
\end{aligned}$$

Observe that $\|I'\| = lm$. Without loss of generality assume that $p' = lm = p$. Then for every $\bar{a} = (a_1, \ldots, a_{p-1}, a_{p+1}, \ldots, a_n) \in \Omega(I, p)$, define $\bar{a}' \in \Omega(I', p')$ as $(a_1, \ldots, a_{p'-1})$. As in the case of $\bar{a}^*$, it is easy to see that $\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})} \leq \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}') \log \frac{1}{P(a|\bar{a}')}$. Particularly, this property holds for every $\bar{a} \in \Omega^*(I, p)$. Thus, by (11) we conclude that

$$\begin{aligned}
\textsc{Inf}_{I'}(p' \mid \Sigma') &= \lim_{k\to\infty} \frac{1}{\log k} \sum_{\bar{a}\in\Omega(I',p')} \frac{1}{2^{p'-1}} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&= \frac{1}{2^{p'-1}} \sum_{\bar{a}\in\Omega(I',p')} \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&\geq \frac{1}{2^{p-1}} \sum_{\bar{a}\in\Omega^*(I,p)} \lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \\
&\geq \textsc{Inf}_I(p \mid \Sigma).
\end{aligned}$$

□

Proof of Theorem 5: First, we notice that adding new relations and constraints over them to a schema does not affect the information content of the old positions. Namely, let $S = \{R_1, \ldots, R_m\}$ be a relational schema, $\Sigma = \Sigma_1 \cup \cdots \cup \Sigma_m$ be a set of FDs over S such that Σ_i is a set of FDs over R_i ($i \in [1, m]$), $S' = \{R_1\}$, $\Sigma' = \Sigma_1$, $I \in inst(S, \Sigma)$ and $I' \in inst(S', \Sigma')$ such that

$I' = I(R_1)$. Furthermore, let p be a position in $I(R_1)$ and p' the corresponding position in I'. Then $\text{INF}_I(p \mid \Sigma) = \text{INF}_{I'}(p' \mid \Sigma')$. The theorem now is a direct consequence of this fact and Lemma 7. □

A decomposition algorithm is *effective* in I if for one of its basic steps, and for some p, the inequality in Theorem 5 is strict: that is, the amount of information increases. This notion leads to another characterization of BCNF.

Proposition 7 *(R, Σ) is in BCNF if and only if no decomposition algorithm is effective in (R, Σ).*

Proof: ($\Rightarrow$) If (R, Σ) is in BCNF, then for every $I \in inst(R, \Sigma)$ and $p \in Pos(I)$, $\text{INF}_I(p \mid \Sigma) = 1$. Thus, no decomposition algorithm can be effective on any $I \in inst(R, \Sigma)$.

($\Leftarrow$) Assume that (R, Σ) is not in BCNF. We will show that there exists a decomposition algorithm effective in (R, Σ).

Given that (R, Σ) is not in BCNF, we can find nonempty pairwise disjoint sets of attributes X, Y, Z such that $X \cup Y \cup Z = sort(R)$, $X \to Y \in \Sigma^+$, X is not a key and (XY, Σ_{XY}) is in BCNF. Let I be a database instance of R containing two tuples t_1, t_2 defined as follows. For every $A \in sort(R)$, $t_1[A] = 1$. If $X \to A \in \Sigma^+$, then $t_2[A] = 1$, otherwise $t_2[A] = 2$. It is easy to see that $I \in inst(R, \Sigma)$. Furthermore, for every $A \in Y$ and $p \in Pos(I)$ such that $t_1[A]$ (or $t_2[A]$) is the p-th element in I, $\text{INF}_I(p \mid \Sigma) < 1$ and $\text{INF}_{I_{XY}}(\pi_{XY}(p) \mid \Sigma_{XY}) = 1$ (since (XY, Σ_{XY}) is in BCNF). Therefore, $\text{INF}_I(p \mid \Sigma) < \text{INF}_{I_{XY}}(\pi_{XY}(p) \mid \Sigma_{XY})$. Thus, a decomposition algorithm that decomposes I into I_{XY} and I_{XZ} is effective in (R, Σ). □

6.2 XML data

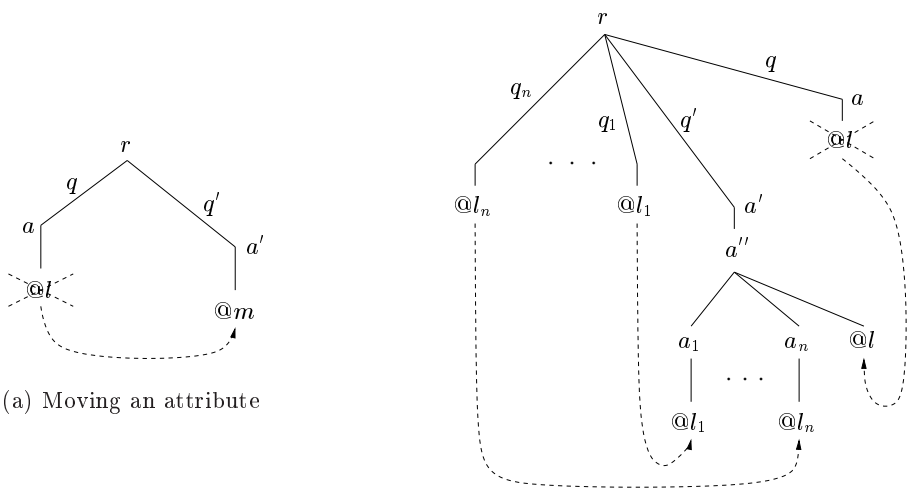
We now treat the XML case. We shall prove a result similar to Theorem 5. However, to state the result, we first need to review the normalization algorithm for XML data proposed in [3], and explain how each step of the algorithm induces a mapping between positions in two XML trees. Throughout the section, we assume that the DTDs are non-recursive and that all FDs contain at most one element path on the left-hand side. Furthermore, for presenting the algorithm and proving the result, we also make the following assumption: if $X \to q.@l$ is an FD that causes a violation of XNF, then every time that $q.@l$ is not null, every path in X is not null (it is shown in [4] how to eliminate this assumption).

To present the algorithm proposed in [3] we need to introduce some terminology. Given a DTD D and a set of FDs Σ, a nontrivial FD $X \to q.@l$ is called *anomalous*, over (D, Σ), if it violates XNF; that is, $X \to q.@l \in (D, \Sigma)^+$ but $X \to q \notin (D, \Sigma)^+$. The algorithm eliminates anomalous functional dependencies by using two basic steps: moving an attribute, and creating a new element type.

Moving attributes. Let $D = (L_0, P, R, r)$ be a DTD and Σ a set of FDs over D. Assume that (D, Σ) contains an anomalous FD $q' \to q.@l$, where q' is an element path. For instance, the DBLP database shown in example 7 contains an anomalous FD of this form:

$$db.conf.issue \to db.conf.issue.inproceedings.@year. \tag{12}$$

To eliminate the anomalous FD, we move the attribute $@l$ from the set of attributes of the last element a of q to the set of attributes of the last element a' of q', as shown in Figure 4 (a). For instance, to eliminate the anomalous functional dependency (12) we move the attribute *@year* from



(a) Moving an attribute

(b) Creating a new element type

Figure 4: Two transformations of the XML normalization algorithm.

the set of attributes of *inproceedings* to the set of attributes of *issue*. Formally, the new DTD $D[q.@l := q'.@m]$, where $@m$ is an attribute, is (L_0, P, R', r), where $R'(a') = R(a') \cup \{@m\}$, $R'(a) = R(a) - \{@l\}$ and $R'(b) = R(b)$ for each $b \in L_0 - \{a, a'\}$.

After transforming D into a new DTD $D[q.@l := q'.@m]$, a new set of functional dependencies is generated. Formally, the set of FDs $\Sigma[q.@l := q'.@m]$ over $D[q.@l := q'.@m]$ consists of all FDs $X \to Y \in (D, \Sigma)^+$ with $X \cup Y \subseteq \mathit{paths}(D[q.@l := q'.@m])$. Observe that the new set of FDs does not include the functional dependency $q \to q'.@l$.

Creating new element types. Let $D = (L_0, P, R, r)$ be a DTD and Σ a set of FDs over D. Assume that (D, Σ) contains an anomalous FD $\{q', q_1.@l_1, \ldots, q_n.@l_n\} \to q.@l$, where q' is an element path and $n \geq 1$. For example, consider the following DTD that describes a database containing courses in different universities:

```
<!ELEMENT db (univ*)>
<!ELEMENT univ (course*)>
<!ELEMENT course (student*)>
  <!ATTLIST course
            cno CDATA #REQUIRED
            title CDATA #REQUIRED>
<!ELEMENT student EMPTY>
  <!ATTLIST student
            sno CDATA #REQUIRED
            name CDATA #REQUIRED
            grade CDATA #REQUIRED>
```

For every course, we store its number (*@cno*), its title and the list of students taking the course. For each student taking a course, we store his/her number (*@sno*), name, and the grade in the

course. In this database we have the following functional dependencies:

$$\begin{aligned} \{db.univ,\ db.univ.course.@cno\} &\rightarrow db.univ.course, \\ \{db.univ,\ db.univ.course.student.@sno\} &\rightarrow db.univ.course.student.@name. \end{aligned} \tag{13}$$

The first FD says that two distinct courses of the same university must have distinct *@cno* numbers, the second one says that two students of the same university with the same *@sno* value must have the same *@name*. We observe that (13) is an anomalous FD of the form described above since $\{db.univ,\ db.univ.course.student.@sno\} \rightarrow db.univ.course.student$ is not in $(D, \Sigma)^+$.

To eliminate the anomalous FD, we construct a new DTD D' by creating a new element type a'' as a child of the last element a' of q', making $a_1, \ldots, a_n$ its children, $@l$ its attribute, and $@l_1, \ldots, @l_n$ attributes of $a_1, \ldots, a_n$, respectively. Furthermore, we remove $@l$ from the set of attributes of the last element a of q, as shown in Figure 4 (b). Formally, if $\{a'', a_1, \ldots, a_n\}$ are element types which are not in L_0, the new DTD, denoted by $D[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$, is (L_0', P', R', r), where $L_0' = L_0 \cup \{a'', a_1, \ldots, a_n\}$ and P', R' are defined as follows.

1) Assume that $a' \rightarrow P_{a'} \in P$. Then $P' = (P - \{a' \rightarrow P_{a'}\}) \cup \{a' \rightarrow (a'')^* P_{a'},\ a'' \rightarrow a_1^* \cdots a_n^*,\ a_1 \rightarrow \epsilon, \ldots, a_n \rightarrow \epsilon\}$.

2) $R'(a'') = \{@l\}$, $R'(a_i) = \{@l_i\}$, for each $i \in [1, n]$, $R'(a) = R(a) - \{@l\}$ and $R'(b) = R(b)$ for each $b \in L_0 - \{a\}$.

For instance, to eliminate the anomalous functional dependency (13), we create a new element type *info* as a child of *courses*, we remove *@name* as an attribute of *student* and we make it an attribute of *info*, we create an element type *number* as a child of *info* and we make *@sno* its attribute. We note that we do not remove *@sno* as an attribute of *student*.

After transforming D into a new DTD$D' = D[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$, a new set of functional dependencies is generated. Formally, $\Sigma[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$ is a set of FDs over D' defined as the union of the sets of constraints defined in 1), 2) and 3):

1) $X \rightarrow Y \in (D, \Sigma)^+$ with $X \cup Y \subseteq paths(D')$;

2) For each FD $X \rightarrow Y \in (D, \Sigma)^+$ with $X \cup Y \subseteq \{q', q_1, \ldots, q_n, q_1.@l_1, \ldots, q_n.@l_n, q.@l\}$, we include an FD obtained from it by changing q_i to $q'.a''.a_i$, $q_i.@l_i$ to $q.a''.a_i.@l_i$, and $q.@l$ to $q.a''.@l$;

3) $\{q',\ q'.a''.a_1.@l_1,\ \ldots,\ q'.a''.a_n.@l_n\} \rightarrow\ q'.a''$, and $\{q'.a'', q'.a''.a_i.@l_i\} \rightarrow q'.a''.a_i$ for $i \in [1, n]$.

The Algorithm. In Figure 5 is shown the normalization algorithm proposed in [3]. This algorithm applies the "moving attributes" and "creating new element types" transformations until the schema is in XNF. We note that the "creating new element types" transformation is not applied to an arbitrary anomalous FD, but rather to a *minimal* one. To understand the notion of minimality for XML FDs, we first introduce this notion for relational databases. Let R be a relation schema containing a set of attributes U and Σ a set of FDs over R. If (R, Σ) is not in BCNF, then there exist pairwise disjoint sets of attributes X, Y and Z such that $U = X \cup Y \cup Z$, $\Sigma \vdash X \rightarrow Y$ and $\Sigma \not\vdash X \rightarrow A$, for every $A \in Z$. In this case we say that $X \rightarrow Y$ is an anomalous FD. To eliminate this anomaly, a decomposition algorithm splits relation R into two relations: $S(X, Y)$ and $T(X, Z)$. A desirable property of the new schema is that S or T is in BCNF. We say that $X \rightarrow Y$ is a minimal anomalous FD if $S(X, Y)$ is in BCNF, that is, $S(X, Y)$ does not contain an anomalous

(1) If (D, Σ) is in XNF then return (D, Σ), otherwise go to step (2).
(2) If there is an anomalous FD $X \rightarrow q.@l$ and an element path q' in D such that $q' \in X$ and $q' \rightarrow X \in (D, \Sigma)^+$, then:
  (2.1) Choose a fresh attribute $@m$
  (2.2) $D := D[q.@l := q'.@m]$
  (2.3) $\Sigma := \Sigma[q.@l := q'.@m]$
  (2.4) Go to step (1)
(3) Choose a (D, Σ)-minimal anomalous FD $X \rightarrow q.@l$, where $X = \{q', q_1.@l_1, \ldots, q_n.@l_n\}$
  (3.1) Create fresh element types a'', a_1, ..., a_n
  (3.2) $D := D[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$
  (3.3) $\Sigma := \Sigma[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$
  (3.4) Go to step (1)

Figure 5: An XML normalization algorithm.

FD. This condition can be defined as follows: $X \rightarrow Y$ is *minimal* if there are no pairwise disjoint sets $X', Y' \subseteq U$ such that $X' \cup Y' \subsetneq X \cup Y$, $\Sigma \vdash X' \rightarrow Y'$ and $\Sigma \nvdash X' \rightarrow X \cup Y$.

In the XML context, the definition of minimality is similar in the sense that we expect the new element types a'', a_1, ..., a_n form a structure not containing anomalous elements. However, the definition of minimality is more complex to account for paths used in FDs. We say that $\{q, q_1.@l_1, \ldots, q_n.@l_n\} \rightarrow q_0.@l_0$ is (D, Σ)-minimal if there is no anomalous FD $X \rightarrow q_i.@l_i \in (D, \Sigma)^+$ such that $i \in [0, n]$ and X is a subset of $\{q, q_1, \ldots, q_n, q_0.@l_0, \ldots, q_n.@l_n\}$ such that $|X| \leq n$ and X contains at most one element path.

Now we prove that after each step of the normalization algorithm proposed in [3], the amount of information in each position does not decrease. Let (D, Σ) be an XML specification and $T \in inst(D, \Sigma)$. Assume that (D, Σ) is not in XNF. Let (D', Σ') be an XML specification obtained by executing one step of the normalization algorithm. Every step of this algorithm induces a natural transformation on XML documents. One of the properties of the algorithm is that for each normalization step that transforms $T \in inst(D, \Sigma)$ into $T' \in inst(D', \Sigma')$, one can find a map $\pi_{T',T} : Pos(T') \rightarrow 2^{Pos(T)}$ that associates each position in the new tree T' with one or more positions in the old tree T, as shown below.

1) Assume that $D' = D[q.@l := q'.@m]$ and, therefore, $q' \rightarrow q.@l$ is an anomalous FD in (D, Σ). In this case, an XML tree T' is constructed from T as follows. For every $t \in tuples_D(T)$, define a tree tuple t' by using the following rule: $t'(q'.@m) = t(q.@l)$ and for every $q'' \in paths(D) - \{q.@l\}$, $t'(q'') = t(q'')$. Then T' is an XML tree whose tree tuples are $\{t' \mid t \in tuples_D(T)\}$. Furthermore, positions in t' are associated to positions in t as follows: if $p' = (t'(q'), @m)$, then $\pi_{T',T}(p') = \{(t(q), @l)\}$; otherwise, $\pi_{T',T}(p') = \{p'\}$.

2) Assume that (D', Σ') was generated by considering a (D, Σ)-minimal anomalous FD $\{q', q_1.@l_1, \ldots, q_n.@l_n\} \rightarrow q.@l$. Thus, $D' = D[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$. In this case, an XML tree T' is constructed from T as follows. For every $t \in tuples_D(T)$, define a tree tuple t' by using the following rule: $t'(q'.a'')$ is a fresh node identifier, $t'(q'.a''.@l) = t(q.@l)$, $t'(q'.a''.a_i)$ is a fresh node identifier ($i \in [1, n]$), $t'(q.a''.q_i.@l_i) = t(q_i.@l_i)$ and for every $q'' \in paths(D) - \{q.@l\}$, $t'(q'') = t(q'')$. Then T' is an XML tree whose tree tuples are $\{t' \mid t \in tuples_D(T)\}$. Furthermore, positions in t' are associated to positions in t as

follows. If $p' = (t'(q'.a''), @l)$, then $\pi_{T',T}(p') = \{(t(q), @l)\}$. If $p' = (t'(q'.a''.a_i), @l_i)$, then $(t(q_i), @l_i) \in \pi_{T',T}(p')$ (note that in this case $\pi_{T',T}(p)$ may contain more than one position). For any other position p' in t', $\pi_{T',T}(p') = \{p'\}$.

Similarly to the relational case, we can now show the following.

Theorem 6 *Let T be a tree that conforms to a DTD D and satisfies a set of FDs Σ, and let $T' \in inst(D', \Sigma')$ result from T by applying one step of the normalization algorithm. Let $p' \in Pos(T')$. Then*

$$\textsc{Inf}_{T'}(p' \mid \Sigma') \quad \geq \quad \max_{p \in \pi_{T',T}(p')} \textsc{Inf}_T(p \mid \Sigma).$$

Proof: Let (D, Σ) be an XML specification and $T \in inst(D, \Sigma)$. Assume that (D, Σ) is not in XNF. Let (D', Σ') be an XML specification obtained by executing one step of the normalization algorithm. We have to prove that for every $p' \in Pos(T')$, $\textsc{Inf}_{T'}(p' \mid \Sigma') \geq \max_{p \in \pi_{T',T}(p')} \textsc{Inf}_T(p \mid \Sigma)$. This can be done in exactly the same way as the proof of Theorem 5. First, by using the same proof as for Lemma 6, we show that the same results holds for XML trees. Using this, we show the following:

1) Assume $D' = D[q.@l := q'.@m]$ and $q' \rightarrow q.@l$ is an anomalous FD over (D, Σ). Let a' be the last element of q' and $p' \in Pos(T')$. If p' is of the form $(x, @m)$, where $\lambda(x) = a'$, then $\textsc{Inf}_{T'}(p' \mid \Sigma') = 1$ and, therefore, the theorem trivially holds. Otherwise, $\pi_{T',T}(p') = \{p'\}$ and it can be shown that $\textsc{Inf}_{T'}(p' \mid \Sigma') \geq \textsc{Inf}_T(p' \mid \Sigma)$ by using the same proof as that of Lemma 7.

2) Assume that $D' = D[q.@l := q'.a''[a_1.@l_1, \ldots, a_n.@l_n, @l]]$ $\{q', q_1.@l_1, \ldots, q_n.@l_n\} \rightarrow q.@l$ is a (D, Σ)-minimal anomalous FD. Let $p' \in Pos(T')$. If p' is the position in T' of some value reachable from the root by following path $q'.a''.@l$ or $q'.a''.a_i.@l_i$, for some $i \in [1, n]$, then $\textsc{Inf}_{T'}(p' \mid \Sigma') = 1$ since $\{q', q_1.@l_1, \ldots, q_n.@l_n\} \rightarrow q.@l$ is (D, Σ)-minimal. Thus, in this case the theorem trivially holds. Otherwise, $\pi_{T',T}(p') = \{p'\}$ and again it can be shown that $\textsc{Inf}_{T'}(p' \mid \Sigma') \geq \textsc{Inf}_T(p' \mid \Sigma)$ by using the same proof as for Lemma 7.

This completes the proof of the theorem. □

Just like in the relational case, one can define effective steps of the algorithm as those in which the above inequality is strict for at least one position, and show that (D, Σ) is in XNF if and only if no decomposition algorithm is effective in (D, Σ).

7 Conclusions and Future Work

Our goal was to find criteria for good data design, based on the intrinsic properties of a data model rather than tools built on top of it, such as query and update languages. We were motivated by the justification of normal forms for XML, where usual criteria based on update anomalies or existence of lossless decompositions are not applicable until we have standard and universally acceptable query and update languages.

We proposed to use techniques from information theory, and measure the information content of elements in a database with respect to a set of constraints. We tested this approach in the relational case and showed that it works: that is, it characterizes the familiar normal forms such as BCNF and 4NF as precisely those corresponding to good designs, and justifies others, more complicated

ones, involving join dependencies. We then showed that the approach straightforwardly extends to the XML setting, and for the case of constraints given by functional dependencies, equates the normal form XNF of [3] with good designs. In general, the approach is very robust: although we do not show it here due to space limitations, it can be easily adapted to the nested relational model, where it justifies a normal form NNF [22, 23].

It would be interesting to characterize 3NF by using the measure developed in this paper. So far, a little bit is known about 3NF. For example, as in the case of BCNF, it is possible to prove that the synthesis approach for generating 3NF databases does not decrease the amount of information in each position. Furthermore, given that 3NF does not necessarily eliminate all redundancies, one can find 3NF databases where the amount of information in some positions is not maximal.

We would like to consider more complex XML constraints and characterize good designs they give rise to. We also would like to connect this approach with that of [16], where information capacities of two schemas can be compared based on the existence of queries in some standard language that translate between them. For two classes of well-designed schemas (those with no constraints, and with keys only), being information-capacity equivalent means being isomorphic [2, 16], and we would like to see if this connection extends beyond the classes of schemas studied in [2, 16].

Acknowledgment We thank Pablo Barceló and Michael Benedikt for helpful comments. We would also like to thank the anonymous referees for several very helpful comments.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] J. Albert, Y. Ioannidis, and R. Ramakrishnan. Equivalence of keyed relational schemas by conjunctive queries. *JCSS*, 58(3):512–534, 1999.

[3] M. Arenas and L. Libkin. A normal form for XML documents. In *PODS'02*, pages 85–96.

[4] M. Arenas and L. Libkin. A normal form for XML documents. To appear in *TODS*.

[5] C. Beeri. On the membership problem for functional and multivalued dependencies in relational databases. *TODS*, 5(3):241–259, 1980.

[6] C. Beeri, P. Bernstein, and N. Goodman. A sophisticate's introduction to database normalization theory. In *VLDB'78*, pages 113–124.

[7] J. Biskup. Achievements of relational database schema design theory revisited. In *Semantics in Databases*, LNCS 1358, pages 29–54. Springer-Verlag, 1995.

[8] R. Cavallo and M. Pittarelli. The theory of probabilistic databases. In *VLDB'87*, pages 71–81.

[9] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[10] M. Dalkilic and E. Robertson. Information dependencies. In *PODS'00*, pages 245–253.

[11] DBLP. `http://www.informatik.uni-trier.de/~ley/db/`.

[12] D. W. Embley and W. Y. Mok. Developing XML documents with guaranteed "good" properties. In *ER'01*, pages 426–441.

[13] R. Fagin. Multivalued dependencies and a new normal form for relational databases. *ACM TODS*, 2(3):262–278, 1977.

[14] R. Fagin. Normal forms and relational database operators. In *SIGMOD'79*, pages 153–160.

[15] R. Fagin. A normal form for relational databases that is based on domains and keys. *ACM TODS*, 6(3):387–415, 1981.

[16] R. Hull. Relative information capacity of simple relational database schemata. *SIAM J. Comput.*, 15(3):856–886, 1986.

[17] P. Kanellakis. *Elements of Relational Database Theory*, In *Handbook of TCS, vol. B*, pages 1075–1144. 1990.

[18] T. T. Lee. An information-theoretic analysis of relational databases - Part I: Data dependencies and information metric. *IEEE Trans. on Software Engineering*, 13(10):1049–1061, 1987.

[19] M. Levene and G. Loizou. Why is the snowflake schema a good data warehouse design? *Information Systems*, to appear.

[20] M. Levene and M. W. Vincent. Justification for inclusion dependency normal form. *IEEE TKDE*, 12(2):281–291, 2000.

[21] D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. *ACM TODS*, 4(4):455–469, 1979.

[22] W.Y. Mok, Y. K. Ng, D. Embley. A normal form for precisely characterizing redundancy in nested relations. *ACM TODS* 21 (1996), 77–106.

[23] Z. M. Özsoyoglu, L.-Y. Yuan. A new normal form for nested relations. *ACM TODS* 12(1): 111–136, 1987.

[24] C. H. Papadimitriou. *Computational Complexity* Addison-Wesley, 1994.

[25] C.E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423 (Part I), 623–656 (Part II), 1948.

[26] D. Suciu. On database theory and XML. *SIGMOD Record*, 30(3):39–45, 2001.

[27] I. Tatarinov, Z. Ives, A. Halevy, and D. Weld. Updating XML. In *SIGMOD'01*, pages 413–424.

[28] V. Vianu. A Web Odyssey: from Codd to XML. In *PODS'01*, pages 1–15.

[29] M. W. Vincent. A corrected 5NF definition for relational database design. *TCS*, 185(2):379–391, 1997.

[30] M. W. Vincent. Semantic foundations of 4NF in relational database design. *Acta Informatica*, 36(3):173–213, 1999.

A Proofs

A.1 Proof of Lemma 1

We start with the following simple but useful observation. The proof follows immediately from genericity.

Claim 1 *Let Σ be a set of generic integrity constraints over a relational schema S, $I \in inst_k(S,\Sigma)$ and $p \in Pos(I)$. Assume that $a, b \in [1,k] - adom(I)$. Then for every $\bar{a} \in \Omega(I,p)$, $|SAT^k_\Sigma(I_{(a,\bar{a})})| = |SAT^k_\Sigma(I_{(b,\bar{a})})|$.*

Next, we need the following.

Claim 2 *Let Σ be a set of integrity constraints over a relational schema S, $I \in inst(S,\Sigma)$, $p \in Pos(I)$ and $\bar{a} \in \Omega(I,p)$. Then for every $a \in \mathbb{N}^+$, there exists $k_0 \in \mathbb{N}^+$ and a polynomial $q_a(k)$ such that $|SAT^k_\Sigma(I_{(a,\bar{a})})| = q_a(k)$, for every $k > k_0$.*

Proof: Let the variables of $\bar{a}$ be $v_1, \ldots, v_l$. Fix $a > 0$, and let m be the maximum value in $adom(I) \cup \{a\}$. Define k_0 to be $m+l+1$. By genericity, $|SAT^{k_0}_\Sigma(I_{(a,\bar{a})})| = 0$ implies $|SAT^k_\Sigma(I_{(a,\bar{a})})| = 0$ for all $k > k_0$, so we assume there is at least one substitution in $SAT^{k_0}_\Sigma(I_{(a,\bar{a})})$.

We consider the set of all triples $\mathcal{P} = (X, \sigma_X, \Pi)$ where

- $X \subseteq \{v_1, \ldots, v_l\}$,
- $\sigma_X : X \to [1,m]$, and
- Π is a partition on $\{v_1, \ldots, v_l\} - X$.

Given $\sigma \in SAT^k_\Sigma(I_{(a,\bar{a})})$, we write $\sigma \sim \mathcal{P}$ if for every $i \in X$, $\sigma(v_i) = \sigma_X(v_i)$, for every $i \notin X$, $\sigma(v_i) \notin [1,m]$, and for every $i, j \notin X$, $\sigma(v_i) = \sigma(v_j)$ iff i and j are in the same block of Π. Observe that for every $\sigma \in SAT^k_\Sigma(I_{(a,\bar{a})})$, there exists exactly one triple $\mathcal{P}$ such that $\sigma \sim \mathcal{P}$.

Let $\sigma, \sigma' \sim \mathcal{P}$ be two substitutions. From the genericity of Σ we immediately see that $\sigma(I_{(a,\bar{a})}) \models \Sigma$ iff $\sigma'(I_{(a,\bar{a})}) \models \Sigma$. Furthermore, if σ collapses two rows in $I_{(a,\bar{a})}$, then so does σ' (since $\sigma(v_i) = \sigma(v_j)$ iff $\sigma'(v_i) = \sigma'(v_j)$). We conclude that $\sigma \in SAT^k_\Sigma(I_{(a,\bar{a})})$ iff $\sigma' \in SAT^k_\Sigma(I_{(a,\bar{a})})$.

The number of triples $\mathcal{P}$ depends on I, a and $\bar{a}$ but not on k. For each $\mathcal{P}$, either all σ with $\sigma \sim \mathcal{P}$ belong to $SAT^k_\Sigma(I_{(a,\bar{a})})$, or none belongs to $SAT^k_\Sigma(I_{(a,\bar{a})})$. Thus, it will suffice to show that for every $\mathcal{P}$, there exists a polynomial $q^{\mathcal{P}}_a(k)$ such that $|\{\sigma \in SAT^k_\Sigma(I_{(a,\bar{a})}) \mid \sigma \sim \mathcal{P}\}| = q^{\mathcal{P}}_a(k)$.

The case when no σ with $\sigma \sim \mathcal{P}$ belongs to $SAT^k_\Sigma(I_{(a,\bar{a})})$ is trivial: $q^{\mathcal{P}}_a(k) = 0$ for all k. Otherwise, let $\mathcal{P} = (X, \sigma_X, \Pi)$, and let $m_{\mathcal{P}}$ be the number of partition blocks of Π. The number of $\sigma \sim \mathcal{P}$ is then the number of ways to chose $m_{\mathcal{P}}$ distinct ordered elements in $[m+1, k]$, that is

$$q^{\mathcal{P}}_a(k) \quad = \quad \prod_{i=0}^{m_{\mathcal{P}}-1} (k - m - i).$$

Since m and $m_{\mathcal{P}}$ do not depend on k, this concludes the proof of the claim. □

Proof of Lemma 1: Let $I \in inst(S,\Sigma)$, $p \in Pos(I)$, and $\bar{a} \in \Omega(I,p)$. To prove this lemma it suffices to show that the following limit exists:

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a \in [1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})}. \tag{14}$$

By Claims 1 and 2, there exists $k_0 > 0$ and polynomials $q_a(k)$, for every $a \in adom(I)$, and $q(k)$ such that for every $k > k_0$:

1. $|SAT^k_\Sigma(I_{(a,\bar{a})})| = q_a(k)$, for every $a \in adom(I)$;
2. $|SAT^k_\Sigma(I_{(a,\bar{a})})| = q(k)$, for every $a \in [1,k] - adom(I)$.

Let $n = |adom(I)|$ and $r(k) = (k-n)q(k) + \sum_{a \in adom(I)} q_a(k)$. Then (14) is equal to

$$\lim_{k\to\infty} \frac{1}{\log k}\left[\sum_{a\in adom(I)} \left(\frac{q_a(k)}{r(k)} \log \frac{r(k)}{q_a(k)}\right) + (k-n)\frac{q(k)}{r(k)} \log \frac{r(k)}{q(k)}\right]. \tag{15}$$

We first show that

$$\lim_{k\to\infty} \frac{1}{\log k}\left[\sum_{a\in adom(I)} \frac{q_a(k)}{r(k)} \log \frac{r(k)}{q_a(k)}\right] \;=\; 0. \tag{16}$$

Note that $degree(r) \geq degree(q_a)$ for every $a \in adom(I)$. If $degree(r) > degree(q_a)$, then clearly $\lim_{k\to\infty} \frac{q_a(k)}{r(k)} \log \frac{r(k)}{q_a(k)} = 0$. If $degree(r) = degree(q_a)$, then $\lim_{k\to\infty} \frac{q_a(k)}{r(k)} \log \frac{r(k)}{q_a(k)}$ exists and equals some positive constant c_a; hence $\lim_{k\to\infty} \frac{1}{\log k} \frac{q_a(k)}{r(k)} \log \frac{r(k)}{q_a(k)} = 0$. Thus, (16) holds and (15) equals

$$\lim_{k\to\infty}\left[\frac{(k-n)}{\log k} \cdot \frac{q(k)}{r(k)} \cdot \log \frac{r(k)}{q(k)}\right]. \tag{17}$$

By the definition of r, $degree(r) \geq degree(q) + 1$. A simple calculation shows that for $degree(r) = degree(q) + 1$, (17) equals some positive constant that depends on the coefficients of q and r, and for $degree(r) > degree(q) + 1$, (17) equals 0. Hence, the limit (15) always exists, which completes the proof. □

A.2 Proof of Lemma 6

Assume that

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \;\neq\; 0. \tag{18}$$

We will show that this limit must be 1.

First note that by (18), there exists $k_0 > 0$ such that for every $k \geq k_0$ and $a \in [1,k] - adom(I)$, $|SAT^k_\Sigma(I_{(a,\bar{a})})| \geq 1$. If this were not true, then by Claim 1, for every $a \in \mathbb{N}^+ - adom(I)$, we would have $|SAT^k_\Sigma(I_{(a,\bar{a})})| = 0$ and, therefore, $\sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})} \leq \log |adom(I)|$. We conclude that

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \;\leq\; \lim_{k\to\infty} \frac{\log |adom(I)|}{\log k} \;=\; 0,$$

which contradicts (18).

To prove the lemma we need to introduce an equivalence relation on the elements of $\bar{a}$ and prove some basic properties about it. Assume that $\|I\| = n, n > 0$. Let $k \geq k_0$ be such that $adom(I) \subsetneq [1,k]$. Given $a_i, a_j \in \bar{a}$, we say that a_i and a_j are *linked* in $(a,\bar{a})$, written as $a_i \sim a_j$,

if for every substitution $\sigma : \bar{a} \to [1,k]$ such that $\sigma(I_{(a,\bar{a})}) \models \Sigma$, it is the case that $\sigma(a_i) = \sigma(a_j)$. Observe that if a_i, a_j are constants, then $a_i \sim a_j$ iff $a_i = a_j$. It is easy to see that $\sim$ is an equivalence relation on $\bar{a}$. We say that $a_i \in \bar{a}$ is *determined in* $(a, \bar{a})$ if for every pair of substitutions $\sigma_1, \sigma_2 : \bar{a} \to [1,k]$ such that $\sigma_1(I_{(a,\bar{a})}) \models \Sigma$ and $\sigma_2(I_{(a,\bar{a})}) \models \Sigma$, it is the case that $\sigma_1(a_i) = \sigma_2(a_i)$. Notice that if a_i is a constant, then a_i is determined in $(a, \bar{a})$. Furthermore, observe that if $a_i \sim a_j$ and a_i is determined in $(a, \bar{a})$, then a_j is determined in $(a, \bar{a})$. Thus, we can extend the definition for equivalence classes: $[a_i]_\sim$ is determined in $(a, \bar{a})$ if a_i is determined in $(a, \bar{a})$. We define $undet(a, \bar{a})$ as the set of all undetermined equivalence classes of $\sim$:

$$undet(a, \bar{a}) \quad = \quad \{[a_i]_\sim \mid a_i \in \bar{a} \text{ and } [a_i]_\sim \text{ is not determined}\}.$$

Claim 3

1) *For every $a \in adom(I)$ and $b \in [1,k] - adom(I)$, if there exists a substitution $\sigma : \bar{a} \to [1,k]$ such that $\sigma(I_{(b,\bar{a})}) \models \Sigma$, then $|undet(b, \bar{a})| \geq |undet(a, \bar{a})|$.*

2) *For every $a, b \in [1,k] - adom(I)$, $undet(b, \bar{a}) = undet(a, \bar{a})$.*

Proof: 1) Let $a \in adom(I)$ and $b \in [1,k] - adom(I)$. Assume that there exists a substitution $\sigma : \bar{a} \to [1,k]$ such that $\sigma(I_{(b,\bar{a})}) \models \Sigma$. It is easy to see that for every $a_i, a_j \in \bar{a}$, if a_i is determined in $(b, \bar{a})$, then a_i is determined in $(a, \bar{a})$, and if a_i, a_j are linked in $(b, \bar{a})$, then a_i, a_j are linked in $(a, \bar{a})$. Thus, $|undet(b, \bar{a})| \geq |undet(a, \bar{a})|$.

2) Trivial, by Claim 1. □

Claim 4 *Let $a \in [1,k] - adom(I)$. If $k > 2n$, then $|SAT^k_\Sigma(I_{(a,\bar{a})})| \geq (k - 2n)^{|undet(a,\bar{a})|}$.*

Proof: To prove this claim, we consider two cases.

First assume that $\bar{a}$ does not contain any variable. Then $|undet(a, \bar{a})| = 0$ and we have to prove that $|SAT^k_\Sigma(I_{(a,\bar{a})})| \geq 1$. For that, it suffices to show that $I_{(a,\bar{a})} \models \Sigma$. Towards a contradiction, assume that $I_{(a,\bar{a})} \not\models \Sigma$. Then by Claim 1, $|SAT^k_\Sigma(I_{(b,\bar{a})})| = 0$, for every $b \in \mathbb{N}^+ - adom(I)$, which contradicts the existence of k_0.

Second assume that $\bar{a}$ contains at least one variable. Let $\sigma_0 : \bar{a} \to [1,k]$ be a substitution such that $\sigma_0(I_{(a,\bar{a})}) \models \Sigma$ (such a substitution exists by assumption (18)). Let $\sigma : \bar{a} \to [1,k]$ be a substitution such that: (a) σ and σ_0 coincide in determined equivalence classes; (b) for every undetermined class $[a_i]_\sim$, σ assigns the same value in $[1,k] - (adom(I) \cup \{a\})$ to each element in this class; (c) for every pair of distinct undetermined classes $[a_i]_\sim$, $[a_j]_\sim$, $\sigma(a_i) \neq \sigma(a_j)$. Notice that such a function exists since $k > 2n$. Given that $\sigma_0(I_{(a,\bar{a})}) \models \Sigma$, we have $\sigma(I_{(a,\bar{a})}) \models \Sigma$. Thus, $|SAT^k_\Sigma(I_{(a,\bar{a})})|$ is greater than or equal to the number of substitutions with domain $\bar{a}$ and range contained in $[1,k]$ satisfying conditions (a), (b) and (c). Therefore, $|SAT^k_\Sigma(I_{(a,\bar{a})})| \geq (k - (n+1))(k - (n+2)) \cdots (k - (n + |undet(a,\bar{a})|)) \geq (k - 2n)^{|undet(a,\bar{a})|}$. This proves the claim. □

We will use this claim to prove that $\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})} = 1$. Let $k \geq k_0$ be such that $adom(I) \subseteq [1,k]$ and $k > 2n$. By Claim 4, for every $a \in [1,k] - adom(I)$, $|SAT^k_\Sigma(I_{(a,\bar{a})})| \geq (k - 2n)^{|undet(a,\bar{a})|}$. Furthermore, by Claim 3, for every $a \in [1,k] - adom(I)$:

$$\sum_{b\in[1,k]} |SAT^k_\Sigma(I_{(b,\bar{a})})| \;\leq\; \sum_{b\in[1,k]} k^{|undet(b,\bar{a})|} \;\leq\; k^{|undet(a,\bar{a})|+1}$$

Thus, for every $a \in [1,k] - adom(I)$:

$$P(a \mid \bar{a}) \;\geq\; \frac{(k-2n)^{|undet(a,\bar{a})|}}{k^{|undet(a,\bar{a})|+1}} \;=\; \frac{1}{k}\,(1-\frac{2n}{k})^{|undet(a,\bar{a})|}. \tag{19}$$

By Claim 1, for every $a, b \in [1,k] - adom(I)$, $P(a \mid \bar{a}) = P(b \mid \bar{a})$ and, therefore,

$$P(a \mid \bar{a}) \;\leq\; \frac{1}{k - |adom(I)|} \;\leq\; \frac{1}{k-n}. \tag{20}$$

Therefore, using (19) and (20) we conclude that:

$$\begin{aligned}\sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \;&\geq\; \sum_{a\in[1,k]-adom(I)} \frac{1}{k}\,(1-\frac{2n}{k})^{|undet(b,\bar{a})|} \log(k-n) \\ &\geq\; \log(k-n)\,(1-\frac{n}{k})(1-\frac{2n}{k})^{|undet(b,\bar{a})|},\end{aligned}$$

where b is an arbitrary element in $[1,k] - adom(I)$. Thus,

$$\frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \;\geq\; \frac{\log(k-n)}{\log k}\,(1-\frac{n}{k})(1-\frac{2n}{k})^{|undet(b,\bar{a})|}.$$

It is straightforward to prove that $\lim_{k\to\infty}[\frac{\log(k-n)}{\log k}\,(1-\frac{n}{k})(1-\frac{2n}{k})^{|undet(b,\bar{a})|}] = 1$. Thus,

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} \geq 1$$

and, therefore,

$$\lim_{k\to\infty} \frac{1}{\log k} \sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a \mid \bar{a})} = 1,$$

since $\sum_{a\in[1,k]} P(a \mid \bar{a}) \log \frac{1}{P(a|\bar{a})} \leq \log k$. This completes the proof of Lemma 6.