Journal of Computer and System Sciences 82 (2016) 856–877

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

On the finite and general implication problems of independence atoms and keys [☆]

Miika Hannula [a], Juha Kontinen [a,*], Sebastian Link [b]

[a] Department of Mathematics and Statistics, University of Helsinki, Finland
[b] Department of Computer Science, University of Auckland, New Zealand

ARTICLE INFO

Article history:
Received 3 June 2015
Received in revised form 18 November 2015
Accepted 15 February 2016
Available online 9 March 2016

Keywords:
Armstrong relation
Axiomatization
Dependence logic
Finite implication
Implication
Independence
Key

ABSTRACT

We investigate implication problems for keys and independence atoms in relational databases. For keys and unary independence atoms we show that finite implication is not finitely axiomatizable, and establish a finite axiomatization for general implication. The same axiomatization is also sound and complete for finite and general implication of unary keys and independence atoms, which coincide. We show that the general implication of keys and unary independence atoms and of unary keys and general independence atoms is decidable in polynomial time. For these two classes we also show how to construct Armstrong relations. Finally, we establish tractable conditions that are sufficient for certain classes of keys and independence atoms not to interact.

1. Introduction

We study two fundamental classes of integrity constraints in relational databases: Keys and independence atoms. Keys are one of the most important classes of integrity constraints as effective data processing largely depends on the identification of data records. Their importance is manifested in the de facto industry standard for data management, SQL, and they enjoy native support in every real-world database system. The ultimate goal in database normalization is to reduce the given set of integrity constraints to keys and domain constraints only, as this guarantees the absence of data redundancy from any future database instances that comply with these keys, and therefore allows database systems to process updates efficiently [17]. A relation r satisfies the key $\mathcal{K}(X)$ for a set X of attributes, if for all tuples $t_1, t_2 \in r$ it is true that $t_1 = t_2$ whenever t_1 and t_2 have matching values on all the attributes in X.

Independence atoms (IA) are less known in the database community, but have already been introduced under the term *cross product* by Paredaens in 1980 [58]. While different, independence atoms correspond to marginal probabilistic independence statements well-known in statistics and artificial intelligence. Marginal statements were investigated in depth by Geiger, Paz, and Pearl in 1991 [22]. Independence atoms occur naturally in data processing. A relation r satisfies the independence atom $X \perp Y$ between two sets X and Y of attributes, if for all tuples $t_1, t_2 \in r$ there is some tuple $t \in r$ which matches the values of t_1 on all attributes in X and matches the values of t_2 on all attributes in Y. In other words, in rela-

[☆] The first two authors were supported by grants 292767 and 264917 of the Academy of Finland. The research is supported by the Marsden Fund Council grant 3701176 from Government funding, administered by the Royal Society of New Zealand.

[*] Corresponding author.
E-mail addresses: miika.hannula@helsinki.fi (M. Hannula), juha.kontinen@helsinki.fi (J. Kontinen), s.link@auckland.ac.nz (S. Link).

http://dx.doi.org/10.1016/j.jcss.2016.02.007
0022-0000/© 2016 Elsevier Inc. All rights reserved.

tions that satisfy $X \perp Y$, the occurrence of X-values is independent of the occurrence of Y-values. An interesting special case are IAs of the form $X \perp X$ which is satisfied by a given relation if its projection on X contains at most one tuple. In other words, the relation is constant on X. For a simple example of a general independence atom, consider a database schema that stores information about the enrollment of students into a fixed course. The schema records for each enrolled *student* the *year* in which they completed a *prerequisite* course. Intuitively, every student must have completed every prerequisite for the course in some year. For this reason, for every value in the *student* column and every value in the *prerequisite* column there is some value in the *year* column such that these three values together form a tuple. That is, $\mathit{student} \perp \mathit{prerequiste}$ is a constraint that should hold on every meaningful relation over this schema. One of the most fundamental operators in relational algebra is the Cartesian product (or cross product), combining every tuple from one relation with every tuple from a second relation. In SQL, users must specify this database operation in the form of the `FROM` clause. For a minimal example consider two singleton attribute schemata *part* and *supplier* that we join.

part
engine

supplier
Mercedes
BMW

part	*supplier*
engine	Mercedes
engine	BMW

The definition of the Cartesian product entails that the resulting relation satisfies the independence atom $\mathit{part} \perp \mathit{supplier}$. It contains redundant data value occurrences in the sense that changing the value to *any* different value will result in the violation of some given constraint. For instance, changing the second occurrence of 'engine' to any other value in the example relation above will violate the independence atom $\mathit{part} \perp \mathit{supplier}$. Independence is therefore a major source of data redundancy, a property that largely determines which queries and updates can be processed efficiently [1,17,41,48,67]. Independence is thus a fundamental concept in database schema design, exhibited for example, by multivalued dependencies. A relation satisfies a multivalued dependency $X \twoheadrightarrow Y \perp R - XY$ over relation schema R if and only if the relation is the lossless join of its projections on XY and $X(R - XY)$. In other words, for each fixed X-value in the relation, the set of associated Y-values is independent of the set of associated $R - XY$-values. Multivalued dependencies correspond to saturated conditional independence atoms [23,26,51,55], and capture a large proportion of the integrity constraints specified in practice. They form the foundation for Fagin's Fourth Normal form [16]. Due to their fundamental importance in everyday data processing in practice, both keys and independence atoms have also received much research interest since the 1970s [6,13,14,17,22,30–32,34,42,49,50,52,58]. The core reasoning problems of data dependencies are their associated implication problems, with about 100 different classes studied so far [65]. Efficient solutions to these problems have important applications, for example, in database design, query and update processing, data cleaning, exchange, integration and security. Section 2 contains some showcases that illustrate the benefit of such solutions to the processing of updates and queries, as well as data privacy.

Given their importance for data processing in practice, given that keys and independence atoms naturally co-exist and given the long and fruitful history of research into relational data dependencies, it is rather surprising that keys and independence atoms have not been studied together. This is particularly true as more expressive classes of dependencies do not have feasible implication problems. In fact, keys are subsumed by numerical dependencies which do not enjoy a finite axiomatization [25], and independence atoms $Y \perp Z$ are subsumed by embedded multivalued dependencies $X \to Y \perp Z$ [12,16,60] as the special case where $X = \emptyset$, but whose implication problem is not finitely axiomatizable [62] and undecidable [35,36]. A relation satisfies an embedded multivalued dependency $X \to Y \perp Z$ if and only if the projection of the relation onto XYZ satisfies the multivalued dependency $X \to Y \perp Z$. While embedded multivalued dependencies are strongly related to probabilistic conditional independence statements [11], Studeny showed that their associated implication problems are different [63,64]. Studeny also showed that the implication problem of probabilistic conditional independence statements is not finitely axiomatizable, and the proof relies on a circular system of these statements [63,64]. These remarks show that independence is also a useful notion for probabilistic approaches to certain machine learning problems [4,15,55]. Nevertheless these approaches are different from the independence atoms we study here: We do not consider probabilities but are interested in the notion of independence as a class of data dependencies. There are also expressive classes of data dependencies whose implication problem can be decided efficiently. For example, the combined class of functional and multivalued dependencies enjoys an elegant finite axiomatization and is decidable in almost linear time [3,20]. These results can be extended to the general implication problem of functional, multivalued, and unary inclusion dependencies. On the other hand, the finite implication problem of functional, multivalued, and unary inclusion dependencies can be decided in cubic time in the input, and while it enjoys an elegant axiomatization it requires one cyclic inference rule for each positive integer [10]. Note that functional dependencies extend keys, but multivalued dependencies are full dependencies and cannot express many independence atoms, which are embedded dependencies. In fact, the intersection of multivalued dependencies and independence atoms consists of multivalued dependencies of the form $X \twoheadrightarrow Y \perp Z$ where $X = \emptyset$, or in other words, of independence atoms of the form $Y \perp Z$ where the underlying relation schema is the union of Y and Z. Keys and independence atoms in isolation enjoy efficient solutions to computational problems: Finite and general implication problems coincide, and are axiomatizable by finite sets of Horn rules [22,42,58,65]. They thus are excellent candidates to push the frontier of axiomatizable classes of data dependencies.

Motivated by real-world applications and the lack of previous research we initiate research on the interaction of key dependencies and independence atoms. As far as we are aware, keys and independence atoms together constitute the first