Correction to "An Equivalence between Relational Database Dependencies and a Fragment of Propositional Logic"

Y. SAGIV

Hebrew University, Jerusalem, Israel

C. DELOBEL

University of Grenoble, Grenoble, France

D. S. PARKER, JR.

University of California, Los Angeles, California

AND

RONALD FAGIN

IBM Almaden Research Center, San Jose, California

According to the definition of satisfaction of Boolean dependencies, Theorem 15 is not true for Boolean dependencies with negation. (A *positive* Boolean dependency is built using the Boolean connectives $\wedge$, $\vee$, and $\Rightarrow$; a general Boolean dependency (with negation) may use also the Boolean connective $\neg$.) Actually, the definition of satisfaction is not meaningful for Boolean dependencies with negation, since many are never satisfied. We show how the definition of satisfaction should be changed in order to make Boolean dependencies with negation meaningful and correct the error.

We associate with each relation r a set $\alpha(r)$ of *truth assignments*, as follows. For each pair of distinct tuples of r, the set $\alpha(r)$ contains the truth assignment that maps an attribute A to **true** if the two tuples are equal on A, and to **false** if the two tuples have different values for A. A Boolean dependency σ is satisfied by a relation r if σ (i.e., the corresponding Boolean formula) satisfies every truth assignment of $\alpha(r)$.

The original definition given in the paper is equivalent to having $\alpha(r)$ also include the truth assignment that is generated by pairs in which both tuples are really the same tuple of r, that is, to having $\alpha(r)$ also always include the truth assignment τ mapping all attributes to **true**. Under that definition, however, many Boolean dependencies with negation are never satisfied and, hence, are meaningless. More precisely, according to the original definition, a Boolean dependency is satisfied by

Authors' addresses: Y. Sagiv, Institute of Mathematics and Computer Science, Hebrew University of Jerusalem, Givat Ram 91904 Jerusalem, Israel; C. Delobel, Computer Laboratory, University of Grenoble, Boite Postale 53, 38041 Grenoble Cedex, Grenoble, France; D. S. Parker, Jr., Computer Science Department, School of Engineering and Applied Science, University of California at Los Angeles, Los Angeles, CA 90024; Ronald Fagin, IBM Research, Almaden Research Center, Dept. K53/802, 650 Harry Road, San Jose, CA 95120-6099.

Journal of the Association for Computing Machinery, Vol. 34, No. 4, October 1987, pp. 1016–1018.

some relation precisely if its corresponding Boolean formula is **true** under τ. We note that, if we define a *positive* Boolean formula analogously to a positive Boolean dependency (i.e., to be one that is built using the Boolean connectives $\wedge$, $\vee$, and $\Rightarrow$), this latter condition (that a Boolean formula is **true** under τ) holds precisely if the formula is equivalent to a positive Boolean formula.

Under the definition that is being proposed now, it is possible to describe "real-world" properties of data by Boolean dependencies with negation. For example, the Boolean dependency *NAME* $\Rightarrow$ $\neg$*SS#* expresses the fact that two distinct persons with the same name must have different Social Security numbers. This dependency can be specified for a relation scheme if, in any relation for this scheme, each tuple represents a distinct person. Clearly, according to the original definition, *NAME* $\Rightarrow$ $\neg$*SS#* is not satisfied by any relation and thus does not express the above statement (i.e., the statement that two distinct persons with the same name must have distinct Social Security numbers). Moreover, according to the original definition, it is impossible to express that statement by Boolean dependencies.

Since relations do not have duplicate tuples, there is one Boolean dependency that is satisfied by every relation, namely, the Boolean dependency stating that any two distinct tuples cannot be equal in all the columns. Formally, this Boolean dependency is $\neg(A_1 \wedge A_2 \wedge \cdots \wedge A_n)$, where $A_1, \ldots, A_n$ are all the attributes of U. We denote this dependency by ϕ. The correct version of Theorem 15 is the following one.

THEOREM 15. *Assume that Σ is a set of Boolean dependencies and σ a single Boolean dependency. Let $\boldsymbol{\Sigma}$ and $\boldsymbol{\sigma}$ be, respectively, the corresponding set of propositional formulas and single propositional formula. The following are equivalent:*

(1) *σ is a consequence of Σ.*
(2) *σ is a consequence of Σ in the world of 2-tuple relations.*
(3) *$\boldsymbol{\sigma}$ is a logical consequence of $\boldsymbol{\Sigma} \cup \{\boldsymbol{\phi}\}$.* □

Actually, with our new definition of satisfaction of Boolean dependencies, the original version of Theorem 15 is also true provided that we also consider relations with duplicate tuples and not just proper relations (where, in the definition of satisfaction, two copies of the same tuple are considered distinct). There are several good reasons for considering relations with duplicate tuples. First, by the Boolean dependency ϕ, it is possible to express the condition that a relation is *proper*, that is, does not have duplicate tuples. Second, relations with duplicate tuples do occur in practical database systems. For example, when a relation is projected onto a subset of its attributes, or when the union of two relations is performed, duplicate tuples may not be removed unless the user explicitly requests so. Third, every type of dependency studied in the literature so far (except for Boolean dependencies) has the following property: A dependency is satisfied by a relation r if and only if it is satisfied by the relation obtained from r by removing duplicate tuples. Therefore, the existing theory of dependencies (e.g., axiomatization of dependencies, algorithms for testing implications, normal forms) remains unchanged when relations with duplicate tuples are also considered. Thus, by allowing relations to have duplicate tuples, we can develop tools that might be useful in practical situations (and are also consistent with the existing theory of dependencies). Further, it is likely that these tools would also be useful when only proper relations are considered, since it is easy to express the condition characterizing proper relations as a Boolean dependency, as we have seen. In particular (the correct version above of) Theorem 15 is a straightforward corollary of the following theorem, which is just the original version of Theorem 15, but with the added assumption that relations may have duplicate tuples and with our new definition of satisfaction of Boolean dependencies.

THEOREM. *Assume that relations may have duplicate tuples, and suppose that* Σ *is a set of Boolean dependencies and* σ *a single Boolean dependency. Let* $\boldsymbol{\Sigma}$ *and* $\boldsymbol{\sigma}$ *be, respectively, the corresponding set of propositional formulas and single propositional formula. The following are equivalent:*

(1) σ *is a consequence of* Σ.
(2) σ *is a consequence of* Σ *in the world of 2-tuple relations.*
(3) $\boldsymbol{\sigma}$ *is a logical consequence of* $\boldsymbol{\Sigma}$.

In summary, we have corrected Theorem 15 by

(1) changing the definition of satisfaction of Boolean dependencies into a more meaningful definition, and
(2) either modifying condition (3) of Theorem 15 or allowing relations to have duplicate tuples.

ACKNOWLEDGMENTS. We thank Joel Berman and Wim Blok for pointing out the error in the original version of Theorem 15 and for the observation that the original version is correct for positive Boolean dependencies [1].

REFERENCES

1. BERMAN, J., AND BLOK, W. J. Positive Boolean dependencies. Report. Department of Mathematics, Statistics, and Computer Science, Univ. of Illinois, Chicago, Ill. 1985.