Data & Knowledge Engineering 68 (2009) 1128–1155

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Constraint acquisition for Entity-Relationship models

Sven Hartmann [a], Sebastian Link [b,*], Thu Trinh [a]

[a] Department of Informatics, Clausthal University of Technology, Germany
[b] School of Information Management, Victoria University of Wellington, New Zealand

ARTICLE INFO

Article history:
Received 15 May 2008
Received in revised form 4 June 2009
Accepted 4 June 2009
Available online 14 June 2009

Keywords:
Knowledge acquisition
Semantic constraint
Conceptual database
Propositional logic

ABSTRACT

We establish search algorithms from the area of propositional logic as invaluable tools for the semantic knowledge acquisition in the conceptual database design phase. The acquisition of such domain knowledge is crucial for the quality of the target database.

Integrity constraints are conditions that capture the semantics of the application domain under consideration. They restrict the databases to those that are considered meaningful to the application at hand. In practice, the decision of specifying a constraint is very important and extremely challenging.

We show how techniques from propositional logic can be utilised to offer decision support for specifying Boolean and multivalued dependencies between properties of entities and relationships in conceptual databases. In particular, we use a search version of *SAT*-solvers to semi-automatically generate sample databases for this class of dependencies in Entity-Relationship models. The sample databases enable design participants to judge, justify, convey and test their understanding of the semantics of the future database. Indeed, the decision by the participants to specify a dependency explicitly is reduced to their decision whether there is some sample database that they can accept as a future database instance.

1. Introduction

Database design is based on the assumption that the semantics of the underlying application domain has been completely captured by the present database model. However, the complete acquisition of such knowledge presents many challenges. During the design process or in the lifetime of a database, it happens all too often that the knowledge about the database proves to be incomplete. Therefore, the acquisition of semantic domain knowledge is absolutely crucial for the quality of a database.

Integrity constraints can classify database instances into those that are meaningful to the application domain at hand, and those that are not. Indeed, the instances that are not considered meaningful according to this classification, are excluded from the set of possible databases. In this sense, integrity constraints restrict the database instances to those that capture the application domain. A very simple example of such integrity constraints are domain constraints, e.g. the domain of an attribute *Gender* may restrict the values allowed to occur to *female* and *male*. A database that obeys this domain constraint must not have any entry for *Gender* that is *blue*, for instance.

The acquisition and specification of integrity constraints is far from being trivial. This task not only demands high abilities to abstract from the participants of the database design phase, but also tends to be rather complex. For the correct utilisation of semantic information an advanced understanding of logics is required. Sometimes designers misunderstand integrity

* Corresponding author.
E-mail addresses: sven.hartmann@tu-clausthal.de (S. Hartmann), sebastian.link@vuw.ac.nz (S. Link), thu.trinh@tu-clausthal.de (T. Trinh).

0169-023X/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2009.06.001

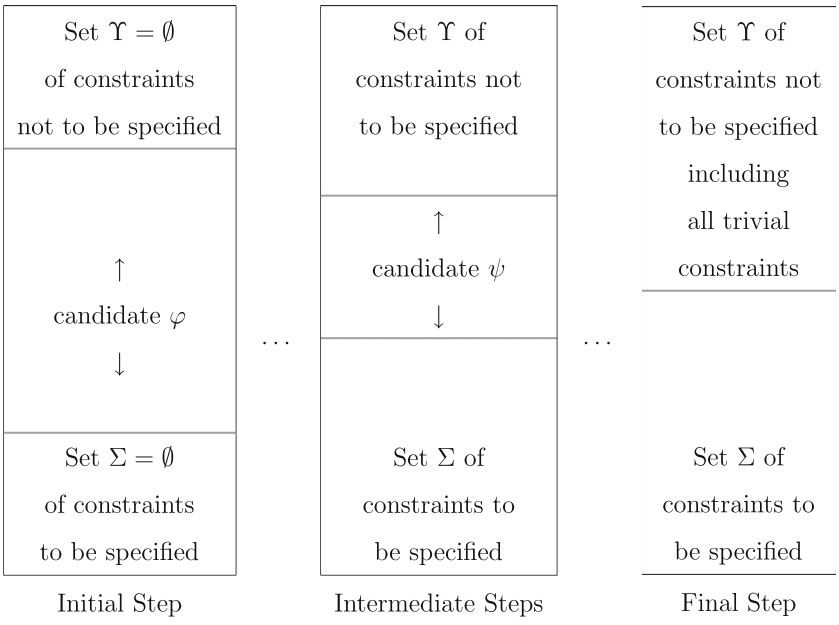

Fig. 1. Constraint acquisition by iterative inspection of candidates.

constraints and, consequently, interpret them badly. In addition, if designers and users work together it may be that they interpret the same constraint in different ways. Participants in the requirements acquisition process have their own skills, experience and knowledge. In view of these problems and the importance of the sound and complete gathering of semantic information, it is highly desirable to support the participants of the database design process in their task of semantic constraint acquisition.

Naturally, constraint acquisition can be an iterative process of inspection: some participant suggests the significance of some integrity constraint φ for the application domain (i.e., φ becomes a candidate constraint that might be specified), and according to the mutual perception φ will be added to the set Σ of constraints that will be specified explicitly or φ will be discarded (added to the set Υ of constraints that do not need to be specified). Any constraint φ that is implied by Σ has already been specified implicitly and can therefore be added to Υ. However, deciding implication is part of the process, and is generally a non-trivial task in itself. The process ends when the potential candidates have been exhausted. At this stage, the set Σ contains all those constraints that will be specified explicitly. In particular, Υ will include all trivial constraints, i.e., those constraints that are satisfied by every database instance. The iterative process is illustrated in Fig. 1. The crucial question is how to deal with those candidates φ that are not implied by Σ? In that situation, sample databases that satisfy all the constraints in Σ and violate φ could offer decision support for the participants of the design process. Such sample databases would enable the participants to better comprehend the consequences of not specifying φ.

The idea of utilising sample databases in the database design process is not new. *Armstrong databases* constitute an invaluable tool for the validation of semantic knowledge, and a user-friendly representation of integrity constraints [1–18]. A database instance is defined to be an Armstrong database for a constraint set Σ, if the database instance satisfies an integrity constraint φ precisely when φ is implied by Σ. In particular, every constraint φ not implied by Σ is violated by an Armstrong database for Σ. Hence, the consequences of not specifying φ can be identified somewhere as a violation in the Armstrong database. However, Armstrong databases do not necessarily give us an indication whether a candidate constraint should be specified. Instead, they only represent a single violation of the candidate constraint. Moreover, an Armstrong database must violate all the constraints not implied by Σ and, therefore, must contain as many tuples as necessary to exemplify all these violations. As a simple example, the minimum size of an Armstrong relation for an arbitrary system of candidate keys[1] over n attributes has lower bound $\frac{1}{n^2}\binom{n}{\lfloor \frac{n}{2} \rfloor}$ [5,14]. Hence, it may become rather difficult for the users and designers to focus on the specific candidate constraint under consideration. In this sense, Armstrong databases do not offer an ideal approach to constraint acquisition by inspecting candidate constraints. This is further supported by the fact that only few classes of integrity constraints actually do enjoy Armstrong databases, i.e., there is no guarantee that for members of a given class of constraints an Armstrong database exists.

In this paper, we introduce an approach towards constraint acquisition that supports the iterative process of inspecting candidate constraints as illustrated in Fig. 1: for each candidate φ we generate sample databases that satisfy Σ and violate φ,

[1] A candidate key is commonly defined as a minimal set of attributes on which no two distinct tuples of a relation coincide.