Equivalence of OLAP Dimension Schemas

Carlos A. Hurtado and Claudio Gutiérrez

Department of Computer Science
Universidad de Chile
{churtado,cgutierr}@dcc.uchile.cl

Abstract. Dimension schemas are abstract models of the data hierarchies that populate OLAP warehouses. Although there is abundant work on schema equivalence in a variety of data models, these works do not cover dimension schemas. In this paper we propose a notion of equivalence that allows to compare dimension schemas with respect to their information capacity. The proposed notion is intended to capture dimension schema equivalence in the context of OLAP schema restructuring. We offer characterizations of schema equivalence in terms of graph and schema isomorphisms, and present algorithms for testing it in well known classes of OLAP dimension schemas. Our results also permit to compare the expressiveness of different known classes of dimension schemas.

1 Introduction

OLAP dimensions are data hierarchies that populate data warehouses. These entities are hierarchically organized information that define the perspective upon which the data is viewed. As an example, in a data warehouse we may have dimensions describing products, stores and time, which may be used to visualize the facts generated by a sales process.

Figure 1 depicts a dimension that models financial services offered by a bank: accounts, credit cards and loans. On the left hand side of Figure 1, there is a graph called *hierarchy schema* which models the structure of the dimension. The vertices of this graph are called categories. On the right hand side, there is another graph, called *hierarchy domain*, whose vertices, called members, are grouped by categories and ordered by a child/parent relation. For example, in the dimension at hand, we may say that member $p1$ belongs to the category $Product$ and $p1$ has $d1$ as a parent in the category $Department$.

In the dimension at hand, some types of products, like personal loans and some sorts of accounts, are handled by branches, whereas others, like mortgage and corporate loans, are handled by departments. Only the products in branches are classified through the hierarchy path $Product$-$ProdType$-$ProdClass$-All. There is a manager in charge of each branch and department. Finally, it happens that the $Asia$ branch and all departments handle products in only one category; thus, their managers belongs to a member in $ProdClass$.

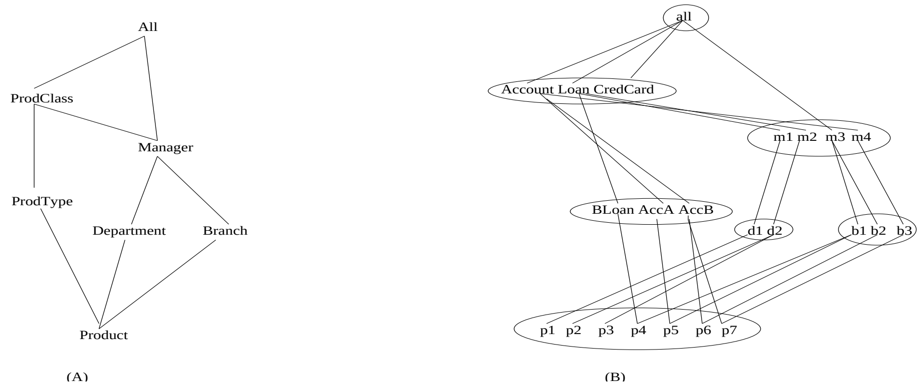

Fig. 1. The dimension Product: (A) hierarchy schema; (B) child/parent relation.

1.1 Dimension Schemas

A *dimension schema* is an abstract model of a dimension commonly used to support summarizability reasoning in OLAP applications [HM02], that is, to test whether aggregate views defined for some categories can be correctly derived from a set of precomputed views defined for other categories. A dimension schema, being an abstract representation of a dimension, represents the set of possible dimensions that conforms to it. This set reflects the *information capacity* of the schema. Thus when we perform reasoning on the schema, we infer properties of all the dimensions in the set.

A central drawback of traditional dimension schemas is that they do not account for structural heterogeneity. Such schemas model dimensions in which members in a category c should have a parent in every category directly above c, a condition we refer to as *homogeneity*. This restriction is unnatural since in many application domains the members of a category have parents (resp., ancestors) in different sets of categories. As an example, in the hierarchy domain of Figure 1 (B), some products are under branches while some others are under departments.

In previous work [HM02] we introduced semantically rich dimension schemas to support summarizability reasoning in heterogeneous dimensions. In our setting, dimension schemas are modeled as a hierarchy schema along with a set of integrity constraints, called *dimension constraints*. The hierarchy schema represents a set of links for the child/parent relation, that is, whenever we have a child/parent relationship between two members in two categories, the categories must be directly connected in the hierarchy schema. Dimension constraints are statements that specify legal paths allowed in the hierarchy domain. The constraints are used to place further restrictions to let the schema capture more precisely different sets of dimensions.

For example, we may require that all the products handled by some branch are not handled by departments, and vice versa. This is stated by the constraint saying that each product can have ancestors in either the path $\langle Product, Branch\rangle$ or the path $\langle Product, Department\rangle$, but not in both. Other constraints may express that the ancestor of some members rollup to members that form a particular path in the hierarchy schema. For example we model that "the manager of the Asia branch rolls up to a product class (because the manager handles products that belong to a single product class)" as "$\langle Branch = Asia\rangle \Leftrightarrow \langle Branch, Manager, ProdClass\rangle$". The expressions in brackets are atomic statements (called *atoms*). It turns out that Boolean combinations of atoms are needed to support summarizability reasoning [HM02].

Simple forms of these constraints characterize typical classes of OLAP schemas. For example, the condition of homogeneity of the edge $(ProdType, ProdClass)$ can be expressed with the constraint $\langle ProdType, ProdClass\rangle$, which asserts that each product type belongs to a product class. In this sense, the class of schemas with dimension constraints subsumes other classes of schemas in OLAP (see Section 2.4) such as the dimension schemas of Jagadish el al. [JLS99], called in this paper *canonical*, which partially solves the limitations of traditional OLAP models by allowing several bottom categories, but keeping the homogeneity restriction. Canonical schemas allow unbalancedness, that is, they can have several bottom categories. In this form, members in different bottom categories may have ancestors in different hierarchy paths in the schema.

Different classes of dimension schemas are classified in [Hur02] where for "traditional OLAP" we refer to the basic class of homogeneous schemas with a single bottom category (balanced schemas).

1.2 Problem Statement

Similarly to the case of general database schemas, two dimension schemas can be compared with respect to their information capacity. Schemas with the same information capacity can be used to simulate each other. In a typical modeling scenario the user starts with some schema and proceeds to restructure it. In the context of OLAP, it is very important that the restructuring process preserves schema equivalence because the schema is more useful for reasoning about data than it is just as a container of data. So we would like to keep the information on the schema as precise as possible to capture the set of instances as tight as possible.

The goal of this paper is to study dimension schema equivalence in the context of OLAP schema restructuring. Formal notions of schema equivalence are fundamental to sit restructuring techniques on solid grounds. For example, Miller et al. [MIR94] argue that the restructuring task may be addressed following two different strategies: (i) build a desired schema and then test whether it is equivalent to the original schema; (ii) use a set of primitives to transform the original schema into a desired schema. In both approaches, we need to define under which conditions two dimension schemas are equivalent. In the first approach we need algorithms for the equivalence test. The second approach requires a set of well defined dimension transformations. The central desirable properties of

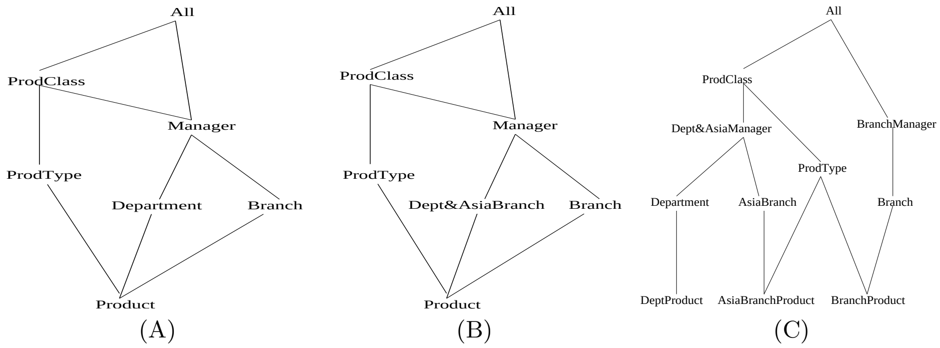

Fig. 2. Product hierarchy schemas.

such a set, soundness and completeness [Alb00], depend on the notion of schema equivalence used as well.

We cast the restructuring task as a process in which the structure of the dimension (i.e. its hierarchy schema) changes but its data hierarchy (hierarchical domain) does not.

Example 1. Consider the hierarchy schemas shown in Figure 2. The three hierarchy schemas can be used to model the hierarchy domain of Figure 1. Hierarchy schema (A) is the same as the one of Figure 1. Hierarchy schema (B) has Asia branch grouped with departments in a single category. Finally, in hierarchy schema (C), the products (bottom members) are split into three categories: $DeptProduct$, $AsiaBranchProduct$, and $BranchProduct$, the branches are split into $Branch$ and $AsiaBranch$, and the managers are split into the categories $Dep\&AsiaManager$ and $BranchManager$. Notice that the dimension having hierarchy schema (C) along with the hierarchy domain of Figure 1 is homogeneous.

Schema equivalence has been formalized by requiring the existence of a bijective mapping between the instances of two equivalent schema [Hul86]. Example 1 shows that a great deal of flexibility in OLAP dimension modeling can be captured by restructuring processes in which members are reorganized into different categories but their hierarchy domain does not change. Thus at the schema level, the correctness of a restructuring process may be formalized by requiring the existence of bijective instance mappings between the schemas which preserve hierarchy domains. As members are associated with facts in datacubes, this mapping restriction guarantees that the aggregate data are preserved through different dimension instances in the restructuring process, thus avoiding aggregate data re-computations, and keeping users to browse aggregate data using the same hierarchy domain.

Example 2. Consider the following dimension schemas. The dimension schema `productA` has the hierarchy schema of Figure 1 (A) along with the dimension

(a)	$\langle Product, Branch\rangle \oplus \langle Product, Department\rangle$
(b)	$\langle Product, Branch\rangle \Leftrightarrow \langle Product, ProdType\rangle$
(c)	$\langle Department, Manager, ProdClass\rangle$
(d)	$\langle Branch = Asia\rangle \Leftrightarrow \langle Branch, Manager, ProdClass\rangle$
(a')	$\langle Product, Dept\&AsiaBranch\rangle \oplus \langle Product, Branch\rangle$
(b')	$(\langle Product, Branch\rangle \vee \langle Product, .., Dept\&AsiaBranch = Asia\rangle \Leftrightarrow \langle Product, ProdType\rangle$
(c')	$\langle Dept\&AsiaBranch, Manager, ProdClass\rangle$
(e)	$\langle c, c'\rangle$ for each edge (c, c') in the hierarchy schema
(f)	$\langle AsiaBranch = Asia\rangle$

Fig. 3. Dimension Constraints for the product hierarchy schemas.

constraints (a)-(d) of Figure 3. The dimension schema `productB` has the hierarchy schema of Figure 1 (B) along with the dimension constraints (a')-(c') of Figure 3. Finally, the dimension schema `productC` has the hierarchy schema of Figure 1 (C) along with the constraint (f) of Figure 3, and a constraint $\langle c, c'\rangle$, for every edge (c, c') in the hierarchy schema. Observe that the products in schema `productC` are now split in three categories depending on where they roll up to (only to Department, to AsiaBranch and ProdType, etc.)

The constraints make the different schemas equivalent (although we have not proved this yet). For example, constraints (c) and (d) of `productA` translate to (c') in `productB`.

1.3 Related Work

There has been abundant work on OLAP dimension modeling over the past few years [CT97,HMV99,LAW98,PJE99,JLS99]. However, to the best of our knowledge, there are no studies regarding dimension schema equivalence. Several notions of schema equivalence for a variety of data models have been proposed. The most general notion of schema equivalence, *absolute equivalence* [Hul86] characterizes the minimum requirements that two schemas must satisfy in order for them to have the same information capacity. Absolute equivalence is formalized by requiring the existence of a bijection between the instances of the schemas. Any arbitrary mapping may be used to guarantee absolute equivalence. In addition, the mappings are not required to be finitely specifiable (they can be an infinite list of pairs of schema instances). A hierarchy of more restricted notions of equivalence has been proposed [Hul86]. For example: *internal equivalence* requires the existence of a bijection that neither creates nor destroy elements in the instances; *query equivalence* requires the mappings to be expressible in the query language of the data model. Other notions of equivalence and their testing have been studied for generic graph data models by Miller et al. [MIR94] and nested data models [VL00]. Our notion of equivalence places minimum restrictions to the instance mappings in order to let them capture all possible ways of grouping members into categories in a schema.

A detailed description of the relationship of dimension constraints and the other constraints for OLAP and other data models is presented by Hurtado [HGM03]. Next, we highlight the most important points in this respect. As explained in several papers (e.g., [JLS99,HMV99]) OLAP dimension may be modeled as a set of normalized tables, one for each category, containing the rollup mappings that start from the category, along with the the attributes of the category. Therefore dimension schemas may be formalized using a relational database setting. It is easily verified that dimension constraints are FOL constraints; therefore, our entire framework is a fragment of FOL. Abiteboul et al. [AV97] study a class of FOL constraints called *embedded constraints* that formalize a wide variety of constraints studied in the database literature. They essentially express that the presence of some tuples in the instance implies the presence of some other tuples in the instance or implies that certain tuple components are equal. Dimension constraints cannot be expressed with embedded constraints, since we cannot assert with them that "some tuples or some other tuples appear in the instance", which are needed to characterize summarizability. Dimension Constraints restrict data in a similar fashion to disjunctive existential constraints (dec's) [Gol81] (which are not embedded constraints). Disjunctive existential constraints are used to characterize the possible sets of non-null attributes that may occur in the tuples of a relation; conceptually, the possible objects that are mixed in the relation. Another class of constraints along the same lines is presented by Husemann et al. [HLV00]. These constraints can be easily represented with dimension schemas, and do not have the full expressiveness of the Boolean connectives needed for summarizability reasoning. *Path constraints* [AV97,BFS98] allow describing certain forms of heterogeneity in semistructured data. They characterize the existence of paths associated with sequences of labels in semistructured data. However, path constraints also lack the entire expressiveness needed to characterize summarizability, and to describe the sort of heterogeneity arising in OLAP applications. On the other hand, path constraints are interpreted over data which have fewer restrictions in their structure than OLAP dimensions, yielding to a different treatment and complexity of their inference.

1.4 Contributions

This paper presents the following contributions:

- A notion of equivalence, hierarchical equivalence, which allows comparison of dimension schemas with respect to their information capacity.
- A proof that hierarchical equivalence can be characterized in terms of graph and schema isomorphisms in two known classes of dimension schemas, called here canonical and balanced. The formal proof of this intuitive connection is non-obvious, as we show in Section 4. The result proves that canonical schemas are more expressive than balanced schemas, hence formally justifying the introduction of canonical schemas.

– A class of schemas –frozen schemas– that act as normal forms for dimensions schemas, in the sense that any dimension schema can be reduced via some well defined transformation to a unique (up to isomorphism) frozen schema. It is proved that hierarchy equivalence test for frozen dimension reduces to a simple form of schema isomorphism. This result leads to other important property of dimension schemas, namely, that heterogeneous dimension schemas can always be transformed into homogeneous schemas. We sketch an algorithm that performs such transformation in an efficient way, and study its complexity.
– Complexity bounds and a study of algorithmic aspects of hierarchical equivalence testing. In particular, we present a characterization of hierarchical equivalence in terms of mappings between minimal dimensions instance contained by the schemas. This leads to an algorithm for testing hierarchical equivalence. We show that the algorithm is more efficient than testing hierarchical equivalence by reducing the schemas to frozen schemas.

1.5 Outline

The remainder of the paper is organized as follows. In Section 2 we review the main concepts related to schemas and state the notation. Section 3 introduces hierarchical equivalence and show its relation with balanced schemas. Section 4 studies hierarchical equivalence of canonical schemas, and shows that in this context hierarchical equivalence corresponds exactly with graph isomorphism of the corresponding hierarchy schemas. In Section 5 we generalize this result to dimension schemas, that is allowing to compare different hierarchy schemas and constraints. The notion of frozen schema is introduced and studied, along with the algorithmic aspects of hierarchical equivalence are studied. Finally, in Section 6 we briefly conclude and outline further work. The complete proofs are presented in the full version of this paper [HG03].

2 Preliminaries

2.1 Basic Graph Terminology

A (directed) graph G is a pair of sets (V, E) where $E \subseteq V \times V$. Elements $v \in V$ are called *vertices* and pairs $(u, v) \in E$ (directed) *edges*; u and v are *adjacent* vertices. A *path* in G from v to w is a sequence of vertices $v = v_0, \ldots, v_n = w$ such that $(v_i, v_{i+1}) \in E$. We say that v *reaches* w. The *length* of a path is n. A *cycle* is a path with $v = w$. A *dag* is a directed acyclic graph. A *sink* in a dag is a distinguished vertex w reachable from every other vertex in the graph. A *source* in a dag is a distinguished vertex v from which every other vertex of the graph is reachable. A *shortcut* in a dag is a path of length > 1 between two adjacent vertices. Given a vertex v of G, an *upgraph* is the subgraph of G generated by v and all the vertices reachable from it.

Given two graphs $G = (V, E)$ and $G' = (V', E')$, a *graph morphism* is a function $\phi : V \to V'$ preserving edges, that is, $(u, v) \in E$ implies $(\phi(u), \phi(v)) \in E'$.

The morphism ϕ is called an *isomorphism (resp. monomorphism, epimorphism)* if ϕ as a function is bijective (resp. injective, onto).

2.2 Dimension Instance

Assume the existence of (possibly infinite) sets of categories $\mathbf{C}$, and of members $\mathbf{M}$.

Definition 1 (Hierarchy Schema). *A* hierarchy schema *is a* dag $H = (C, \nearrow)$, *where* $C \subseteq \mathbf{C}$, *having a distinguished category* $\mathtt{All} \in C$ *which is a sink.*

Definition 2 (Hierarchy Domain). *A Hierarchy domain is a* dag $h = (M, <)$ *where* $M \subset \mathbf{M}$, *having a distinguished member* $\mathtt{all} \in M$ *which is a sink, and without shortcuts.*

The last condition in Definition 2 (no shortcuts) avoids redundancies (transitive edges) in the representation of the data.

Given a child/parent relation $<$, its reflexive and transitive closure, denoted $\leq$, is called *rollup relation*, and is a partial order between members.

Definition 3 (Dimension Instance). *A* dimension instance d over a hierarchy schema $(C, \nearrow)$ *is a graph morphism* $d : (M, <) \rightarrow (C, \nearrow)$ *such that:*

1. $(M, <)$ *is a hierarchy domain;*
2. $d(\mathtt{all}) = \mathtt{All}$*;*
3. $x \leq y \wedge x \leq z$ *implies* $d(y) \neq d(z)$.

The fact that d is a graph morphism in Definition 3 states that whenever we have a child/parent relationship $m_1 < m_2$ between some pair of members $m_1 \in c_1$ and $m_2 \in c_2$, then there is an edge $c_1 \nearrow c_2$ in the hierarchy schema representing links between categories c_1 and c_2. Condition 3 of Definition 3 is a basic restriction in OLAP data modeling [HMV99,CT97,LAW98], and states that the rollup relation $\leq$ is functional (i.e., single valued) between every pair of categories. This motivates to introduce the *rollup mapping* between two categories c_1 and c_2 of a dimension d, denoted $\Gamma_{c_1}^{c_2} d$, which is the restriction of $\leq$ to $d^{-1}(c_1)$ and $d^{-1}(c_2)$.

2.3 Dimension Schema

Next we formalize the notions of dimension constraint and dimension schema.

Definition 4 (Dimension Constraint). *Let* $H = (C, \nearrow)$ *be a hierarchy schema,* $c \in C$, $K \subseteq \mathbf{M}$. *The language of constraints (with root c) has the following atoms:*

1. *Path atoms:* $\langle c, c_1, \cdots, c_n \rangle$, *where* $cc_1 \cdots c_n$ *is a path in* H*;*
2. *Equality atoms:* $\langle c, .., c' = k \rangle$, *where* c' *is such that there is a path from* c *to* c', *and* $k \in K$.

A dimension constraint with root c *is a Boolean combination* ϕ *of atoms of the above kind.*

Dimension constraints consider the usual connectives $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$, and $\oplus$ for exclusive disjunction. In addition, $\perp$ and $\top$ will denote the false and the true proposition, respectively.

Definition 5 (Semantics of Constraints). *Let $d : (M, <) \rightarrow (C, \nearrow)$ be a dimension instance, and ϕ a constraint with root c. Then $d \models \phi$ if and only if*

for all $m \in d^{-1}(c)$, $d \models \phi[c/m]$,

where $d \models \phi[c/m]$ is defined recursively as follows:

1. $d \models \langle c, c_1, \ldots, c_n\rangle[c/m]$ iff there is a path $mx_1 \cdots x_n$ in $(M, <)$ with $d(x_i) \in c_i$.

2. $d \models \langle c, .., c' = k\rangle[c/m]$ iff $d(k) \in c'$ and $m \leq k$.

3. $d \models (\phi \wedge \psi)[c/m]$ iff $d \models \phi[c/m]$ and $d \models \psi[c/m]$. Similarly for $\vee$ and the other Boolean connectives.

Given a hierarchy schema H and two sets of constraints Σ, Σ' over H, we say that Σ is equivalent to Σ', if for all dimension instances d over H it holds: $d \models \Sigma$ iff $d \models \Sigma'$.

Now we are ready to introduce the concept of Dimension Schema. The following definition extends Definition 3 in the presence of constraints.

Definition 6 (Dimension Schema). *A* dimension schema *is a pair (H, Σ) where H is a hierarchy schema and Σ is a set of constraints.*

A dimension instance d over a dimension schema $D = (H, \Sigma)$ *is a dimension instance d over H such that $d \models \Sigma$. The set of dimensions instances over D will be denoted by $I(D)$.*

Definition 7 (Schema Equivalence and Isomorphism).

Let $D = (H, \Sigma)$ and $D' = (H', \Sigma')$ be to dimension schemas.

1. D and D' are equivalent*, denoted $D \equiv D'$, iff $H = H'$ and Σ is equivalent to Σ'.*

2. D and D' are isomorphic*, denoted $D \cong D'$, iff there exists a graph isomorphism $f : H \rightarrow H'$ such that $(f(H), f(\Sigma)) \equiv (H', \Sigma')$, where $f(\Sigma)$ stands for Σ modulo renaming by f.*

Notice that the notion of equivalence of Definition 7 implies isomorphism.

2.4 Classes of Dimension Schemas

The model we have presented subsumes the dimension models presented in the literature. The following definition formalizes two classes of dimension schemas that arise in OLAP.

Definition 8 (Classes of Dimension Schemas). *Let $D = (H, \Sigma)$ be a dimension schema.*

1. D is canonical *iff H has no shortcuts and Σ is equivalent to $\{\langle c, c'\rangle \mid c \nearrow c'\}$.*

2. D is balanced *iff D is canonical and H has a source.*

A dimension instance d is *homogeneous* if for every pair of categories $c_1 \nearrow c_2$ it holds that the rollup mapping $\Gamma_{c_1}^{c_2} d$ is a total function. Note that the constraint $\langle c, c' \rangle$ where $c \nearrow c'$ forces the rollup mapping from c to c' to be total. Therefore, canonical schemas convey all the homogeneous instances over its hierarchy schema. In this sense, in canonical schemas, Σ captures exactly homogeneity. Also notice that we have defined a canonical schema to be shortcut-free, because otherwise Σ would force the categories from which the shortcut start to be empty in every dimension conveyed by the schema. Balanced schemas correspond to the basic class of schemas introduced in early works on OLAP. They are the logical representation of dimension schemas in early snowflake schemas [CD97]. Canonical schemas were introduced by Jagadish et al. [JLS99] to overcome some of the weaknesses of balanced schemas. Canonical schemas allow unbalancedness, that is, they can have dimension instances with two members in the bottom categories having ancestors in different sets of categories. This has been shown to be an important feature to provide flexibility in OLAP modeling.

Example 3. If we delete the constraint (f) to the dimension schema `productC` described in Example 2, the schema turns into a canonical schema.

Given two classes of schemas S_1, S_2, we define $S_1 \sqsubseteq S_2$ iff for each schema in S_1, there is an equivalent schema in S_2. Then it holds Balanced Schemas $\sqsubseteq$ Canonical Schemas $\sqsubseteq$ Dimension Schemas, and the inclusions are proper.

3 Hierarchical Equivalence

In this section we present the notion of hierarchical equivalence in which dimension schemas are related via mappings that preserve the hierarchy domain of the dimensions.

Observe that the notion of schema equivalence of Definition 7 does not allow us to compare schemas having different hierarchy schemas. The following definition generalizes Definition 7 for schemas over arbitrary hierarchy schemas.

Definition 9 (Hierarchical Equivalence). *Two dimension schemas D and D' are* hierarchically equivalent *(h-equivalent) if and only if there is a bijective function $f : I(D) \rightarrow I(D')$ such that for all $d \in I(D)$, $dom(d) = dom(f(d))$. In this case we write $D \equiv_h D'$.*

Observe that the relation $\equiv_h$ is an equivalence relation. Also, it is worth noting that the instance mapping f required for h-equivalence is *internal* [Hul86], i.e., it does neither create nor destroy members or constants in the instances. Moreover, the mapping is *generic* [Hul86], that is, given a pair of dimension instances d and d' with $d' = f(d)$, if we apply the same permutation π of members to d and to d', if $\pi(d)$ is in the domain of f then $\pi(d') = f(\pi(d))$. Thus, hierarchical equivalence is a more restricted notion than internal and generic equivalence.

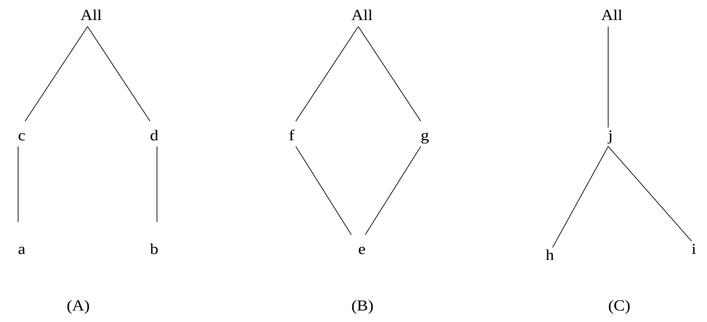

Fig. 4. Three hierarchy schemas.

Example 4. Consider the dimension schemas $D_1 = (A, \Sigma_1)$, $D_2 = (B, \Sigma_2)$ and $D_3 = (C, \Sigma_3)$, where A, B, C are the hierarchy schemas in Figure 4, $\Sigma_1 = \Sigma_3 = \emptyset$ and $\Sigma_2 = \{\neg\langle e, f\rangle \vee \neg\langle e, g\rangle\}$. Then $D_1 \equiv_h D_2$ via mapping the members of c to f, the members of d to g, and the members of a and b to e. However, it is not the case that $D_1 \equiv_h D_3$. Indeed, given a member m, there is a unique dimension instance in $I(D_3)$ whose child/parent relation is $\{m < \texttt{all}\}$, but there are two dimension instances in $\mathcal{I}(D_2)$ whose child/parent relation is $\{m < \texttt{all}\}$.

It is not difficult to check that if $D \cong D'$ then $D \equiv_h D'$. We end this section by showing that it is straightforward to show that the converse also holds for balanced schemas.

A dimension instance d is *exact* if d is bijective. It is easily verified that all canonical dimension schemas have an exact dimension instance.

Theorem 1 (h-Equivalence of Balanced Schemas). *Two balanced dimension schemas $D = (H, \Sigma)$ and $D' = (H', \Sigma')$ are h-equivalent if and only if H and H' are (graph) isomorphic.*

Proof. (Sketch.) One direction is obvious.

Assume that $D \equiv_h D'$ via f. Consider an exact dimension d of (H, Σ). Then as graphs $H \cong \text{dom}(d) \cong \text{dom}(f(d))$. Now, because D' is balanced there is a (graph) monomorphism $\mu : \text{dom}(f(d)) \to H'$ with $\mu(\texttt{all}) = \texttt{All}$ (if $\mu(v) = \mu(w)$ for $v \neq w$, the source of $\text{dom}(f(d))$ would have two ancestors in the same category, violating condition 3 of Definition 3.) Hence there is a monomorphism $H \to H'$. By the same argument on the reverse direction, there is a monomorphism $H' \to H$. Hence because H, H' are finite graphs, $H \cong H'$. $\square$

4 Hierarchical Equivalence of Canonical Schemas

This section extends the results of Theorem 1 to canonical dimensions. The importance of this result is twofold: (1) The notion of h-equivalence has a simple and intuitive characterization as graph isomorphism in canonical schemas (this

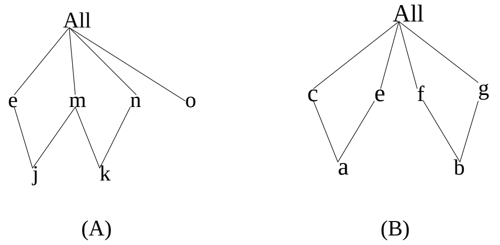

Fig. 5. Two hierarchy schemas.

result is stated in Theorem 2 in this section). (2) From Theorem 2, it follows that canonical schemas are strictly more expressive than balanced schemas (because given a canonical and not balanced schema there is no balanced schema isomorphic to it.) So we have now a formal argument that justifies the introduction of canonical schemas for OLAP modeling.

First, observe that the argument in the proof of Theorem 1 does not necessarily work for canonical schemas (there could be no injective μ).

Example 5. Let D and D' be the dimension schemas of figures 5 (A) and 5 (B), respectively. We define μ as in the proof of Theorem 1. Here, however, μ is not necessarily bijective. In particular, it could be the case that $h(d)$, where d is the exact dimension of D, is a dimension whose non-empty categories are a, c, e, and `All`. And therefore, μ could not be injective.

The following is the main result of the section. We need the following notation: a dimension d is complete with respect to a subgraph H' of its hierarchy schema if $\mathrm{ran}(d) = H'$.

Theorem 2 (h-Equivalence of Canonical Schemas). *Let $D = (H, \Sigma)$ and $D' = (H', \Sigma')$ be two canonical schemas. Then, $D \equiv_h D'$ if and only if H is (graph) isomorphic to H'.*

Proof. (Sketch.) Let us sketch the non-trivial direction of the proof. Let $H = (C, \nearrow)$ and $H' = (C', \nearrow')$ and $f : I(D) \to I(D')$ be the bijection given by $\equiv_h$.

(*) Let $d_1 : (M, <) \to H$ be an exact dimension of D (hence $H \cong (M, <)$). Let $f(d_1) : (M, <) \to H'$ be the image of d_1 under f (by hypothesis $f(d_1)$ has the same domain as d_1). Let d'_1 be an exact dimension of $f(d_1)(M)$. Let d_2 be an exact dimension of $f^{-1}(d'_1)$. Continue this process until $H_1 = \mathrm{Im}(d_i) \cong \mathrm{Im}(d'_i) = H'_1$. Denote by μ_1 this isomorphism. Note that H_1 is well defined because the process terminates by a graph theoretic argument.

For each dimension instance $d : M_1 \to H$ with $d(M_1) = H_1$ do: Redefine f by performing the following operations: $y := f(d)$; $f(d) := (\mu_1 \circ d)$; $ff^{-1}(\mu_1 \circ d) := y$. Recall from Section 2 that an instance d takes its domain from a possibly infinite set $\mathbf{M}$. Here we assume that the set $\mathbf{M}$ is finite, hence the loop ends. The extension to the infinite case is straightforward. It is easily verified that at the

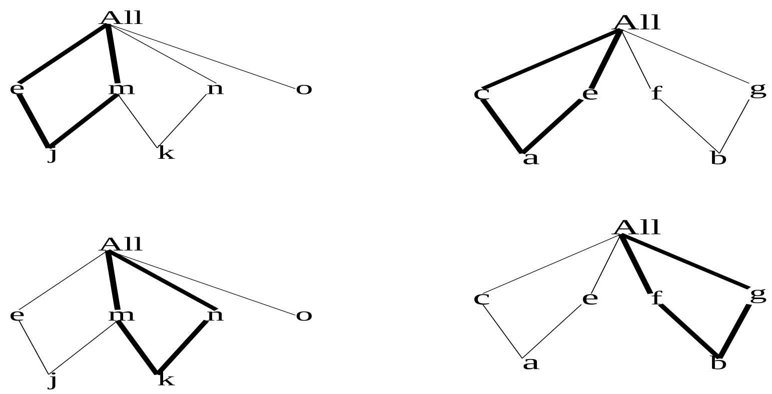

Fig. 6. A series of matchings illustrating proof of Theorem 2

end of this process we will have that for all complete d of H_1, it holds that $f(d) = (\mu_1 \circ d)$. Call f_1 this new f.

Now we repeat the whole process starting from (*) with f_1. This process generates a H_2, H_2' and a new f_2.

Observe that $H_2 \neq H_1$, because otherwise there must be a complete dimension d of H_1 which is not mapped to the complete dimension $(\mu_1 \circ d)$ via f_1.

By repeating this process we generate a series $(H_1, H_1', f_1), (H_2, H_2', f_2), \ldots$ This series has the property $H_i \neq H_j$ for $i \neq j$ by an argument similar to the case $i = 1$. Finally just note that this series must be infinite, but there are only finitely many subgraphs of each hierarchy schema, a contradiction. $\square$

The following examples illustrates the main idea of the previous proof.

Example 6. Let D and D' be the dimension schemas of Figures 5(A) and 5(B). Clearly they are not graph isomorphic. Assume that $D \equiv_h D'$ via an instance mapping f. Figure 6 depicts, on the top, the triple (H_1, H_1', f_1), and in the bottom (H_2, H_2', f_2), in a possible sequence generated in the proof for D and D'. The map f_1 sends the complete dimension of the subschema underlined to the one underlined in H_1'. Similarly for f_2. This property forces the schema H_2 (resp. H_2') to be different from H_1 (resp. H_1'). This series is infinite, but it can be checked now that there is no next triple (H_3, H_3', f_3), yielding a contradiction. Hence $D \not\equiv_h D'$.

5 Hierarchical Equivalence of Dimension Schemas

In this section we present a characterization of hierarchical equivalence for dimension schemas, which yields an algorithm for testing hierarchical equivalence. The characterization will be based on another notion of equivalence, which is defined in terms of mappings between finite sets of minimal dimensions conveyed by the schemas called frozen dimensions.

5.1 Frozen Equivalence

We introduce a notion of equivalence, *frozen equivalence*, defined in terms of injective mappings between special kinds of dimension instances, called frozen. Intuitively, a *frozen dimension* is a minimal dimension conveyed by a dimension schema. Frozen dimensions were introduced in previous work [HM02] to test implication of dimension constraints. In order to test whether a dimension schema satisfies a dimension constraint α, we only need to check α in each frozen dimension of the schema, which yields a finite set of tests (exponential in the size of the schema). In this section we use the notion of frozen dimension for giving an algorithmic version of h-equivalence.

Let $D = (H, \Sigma)$ be a dimension schema, we denote by $\mathtt{Const}_D(c)$ the set of constants k that occur in atoms of the form $\langle c_i, .., c = k\rangle$ in Σ.

Definition 10 (Frozen Dimension). *Given a dimension schema D and $c \in C$, a* frozen dimension *with root c is a dimension instance $d : (M, <) \to (C, <)$ of D such that:*

1. *d is injective (i.e., each category has at most one member);*
2. *$d^{-1}(c)$ is a source of $(M, <)$;*

There could be infinitely many frozen dimensions, but there are only finitely many up to isomorphism, where isomorphism is defined as follows: d is isomorphic to d' iff there exists a graph mapping $f : (M, <) \to (M', <')$ such that $d = d' \circ f$, and if $k \in \mathtt{Const}_D(c_j)$ and $d(k) = c_j = d'(k)$, then $f(k) = k$.

From now one, we will consider frozen dimensions up to isomorphism. Given a dimension schema D and a category c of it, we denote by $\mathtt{Frozen}(D, c)$ the set of frozen dimensions of D (up to isomorphism) with root c, and by $\mathtt{Frozen}(D)$ the union of all $\mathtt{Frozen}(D, c)$ for all categories c of D.

Example 7. Figure 7 (top) shows three subgraphs of the hierarchy schema of the schema `productA` (given in Example 2). Each subgraph is induced by non-empty edges of a frozen dimension of `productA` with root *Product*. Intuitively, the frozen dimensions show the different structures that are mixed in the schema `productA`. Recall that frozen dimensions are dimension instances, but due to lack of space we do not show them directly.

The following notion of equivalence compares the information capacity of two schemas D and D' based on the frozen dimension they convey. To compare D and D' we establish a correspondence Ω between their sets of categories and then check that it induces a bijective relation between their frozen dimensions.

Definition 11 (Frozen Equivalence). *Let $D = (H, \Sigma)$ and $D' = (H', \Sigma')$ be dimension schemas, where $H = (C, \nearrow)$ and $H' = (C', \nearrow')$:*

Two frozen dimensions $d \in \mathtt{Frozen}(D)$ and $d' \in \mathtt{Frozen}(D')$ are isomorphic iff there exists a graph isomorphism $f : (M, <) \to (M', <')$ such that if $c \in C$ and $k \in \mathtt{Const}_D(c)$, then $f(k) = k$. Notice that f induces a isomorphism $\bar{f} : Im(d) \to Im(d')$.

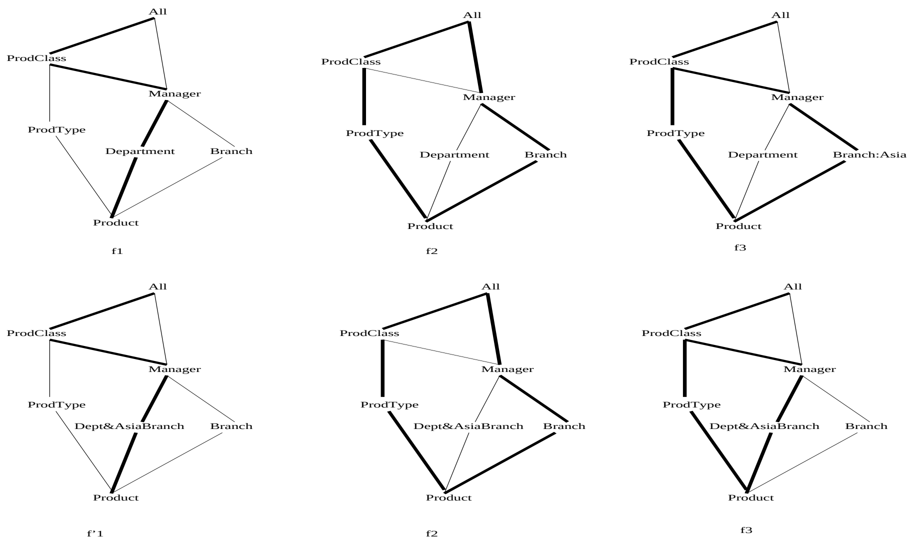

Fig. 7. Frozen Relation between the frozen dimension of `productA` and `productB`.

Let $\Omega \subseteq C \times C'$ be a category correspondence. An Ω-frozen relation $\mathcal{F}_\Omega \subseteq$ $\texttt{Frozen}(D) \times \texttt{Frozen}(D')$ is defined as the set of pairs (d, d') related by an isomorphism $f : d \to d'$ such that $\bar{f} \subseteq \Omega$.

Two schemas D and D' are frozen equivalent (in the sequel f-equivalent) if there exists a bijective Ω-frozen relation from $\texttt{Frozen}(D)$ to $\texttt{Frozen}(D')$.

Example 8. Consider the dimension schemas `productA` and `productB`, and a category correspondence Ω between them having the pairs of categories (*Department*, *Department&AsisBranch*), (*Branch*, *Department&AsisBranch*) and a pair (c, c) for each category c in both schemas. The Ω-frozen relation between the sets $\texttt{Frozen}(\texttt{productA})$ and $\texttt{Frozen}(\texttt{productB})$ is showed in Figure 7. (the figure only shows the frozen dimensions with root Product.)

Proposition 1 (f-Equivalence implies h-Equivalence). *Let D and D' be two dimension schemas. If D and D' are f-equivalent, then D and D' are h-equivalent.*

The proof of this Proposition builds a bijective mapping $f : I(D) \to I(D')$ using the bijective frozen relation. Intuitively, if there is a 1-1 frozen relation (induced by some category correspondence between the schemas) then we can also define a 1-1 instance mapping between the instances of schemas. In Section 5.2 we will state the converse and sketch its proof.

5.2 h-Equivalence and f-Equivalence

In this section we show that h-equivalence implies f-equivalence. This result along with Proposition 1 shows that f-equivalence characterizes h-equivalence.

Firstly, we will introduce *frozen schemas*, dimension schemas that are normal forms, in the sense that every dimension schema is h-equivalent to a frozen schema.

Definition 12 (Frozen Schema). *A* frozen schema *is a dimension schema D such that each category c in D has a unique frozen dimension d and $Im(d)$ is exactly the upgraph of c.*

The following are some basic properties of frozen schemas: their dimension instances are homogeneous; they do not have shortcuts; and they subsume canonical schemas.

Example 9. The dimension schema `productC` given in Example 2 is a frozen schema, because its constraints cause each category c to have a single frozen dimension with root c. The category *AsiaBranch* is the only category whose frozen dimension has a constant (*Asia*). Schemas `productA` and `productB` are not frozen schemas since they convey several frozen dimensions with the category *Product* as root.

Next, we show that testing h-equivalence of frozen schemas defined over the same set of constants reduces to testing whether the schemas are isomorphic. This result generalizes Theorem 2 because canonical schemas are frozen schemas.

Note that two isomorphic frozen dimension must have the same set of constants. We will say that two frozen schemas are normalized w.r.t. a set of constants if for all $c \in C$ and $c' \in C'$ it holds $\texttt{Const}_D(c) = \texttt{Const}_{D'}(c')$, i.e., their equality atoms mention the same constants for each category.

Theorem 3 (h-Equivalence of Frozen Schemas). *Let D and D' be two normalized frozen schemas. Then D and D' are h-equivalent iff they are isomorphic (i.e., $D \equiv_h D'$ iff $D \cong D'$).*

The proof is a generalization of the proof of Theorem 2. This theorem also shows that dimension schemas are more expressive than canonical schemas because some frozen schemas are not isomorphic to any canonical schema. That is, there are dimension schemas for which there are no h-equivalent canonical schemas.

Finally, we prove the main result of this section.

Theorem 4 (h-Equivalence of dimension Schemas). *Let D and D' be two normalized frozen dimension schemas. Then D and D' are f-equivalent iff they are h-equivalent.*

Proof. (Sketch.) One direction is Proposition 1.

So assume that D and D' are h-equivalent. First define a schema transformation that takes D and produces a frozen schema D_f h-equivalent to D. The transformation works as follows: (1) Compute the frozen dimensions of D using the DIMSAT algorithm presented in previous work [HM02]; (2) Reverse the graph $(C, \nearrow')$ and do a topological sort of the resulting graph. (3) Follow the topological sort, and for each category c with more than one frozen dimension, split c into $c_1, \ldots, c_n$ (preserving adjacent edges). Add constraints to the schema in order to have a single frozen dimension in each category $c_1, \ldots, c_n$. This process yields a new dimension schema with a single frozen dimension in each category; (4) For each category c_j of the schema delete adjacent edges that do not match the frozen dimension.

Each split in step (3) induces induces the following category correspondence between the hierarchy schemas before and after the split: (c, c_j) for all $1 \leq j \leq n$, and for the remaining categories c' that appear in both hierarchy schemas we have (c, c'). It is not difficult to verify that this category correspondence induces a bijective frozen relation between the old and the new schema. By composing these frozen relations we get a bijective frozen relation between D and D_f.

In the same manner, we built a frozen schema D'_f and a bijective frozen relation between D' and D'_f. Hence, we have bijective frozen relations $D \rightarrow D_f$ and $D' \rightarrow D'_f$. Also, from Theorem 3 we know that $D_f \cong D'_f$ via some μ. From μ we can derive a bijective frozen relation between D_f and D'_f. Composing these relations we derive the statement of the theorem. $\Box$

Example 10. The bijective frozen relation of Figure 7 shows that the schemas `productA` and `productB` of Example 2 are h-equivalent. By a simmilar procedure it can be easily verified that schema `productC` is h-equivalent to schemas `productA` and `productB`.

5.3 Transforming Dimension Schemas into Homogeneous Schemas

From the proof of Theorem 4 it follows that any dimension schema can be transformed into a hierarchically equivalent homogeneous schema. In Figure 8 we sketch an algorithm to perform such a transformation.

The algorithm outputs dimension schemas having the constraints that state the homogeneity condition. The schemas may also have additional constraints with equality atoms. Path atoms other than the ones that state the homogeneity condition are irrelevant for the resulting schemas because a path atom is either $\perp$ or $\top$ in every instance of a homogeneous schema. In Line 2 the algorithm computes the set of frozen dimensions of D. In Line 5, for each category c of D and subset S of categories directly above c, the algorithm adds to H a new category `CatName`(n) (the function `CatName`(n) returns a name for a new category when a integer n is given) connected to the categories that are connected to c; then, the set of frozen dimensions F is updated in order to keep them consistent with the fact that the new category `CatName`(n) represents the members that have parents only in categories in S. In Line 10 the algorithm does a traversal of the set F deleting empty edges in the hierarchy schema, and adding to Σ' equality atoms associated to the constants that appear in F. In this step the algorithm also adds the constraints of the form $\langle c, c' \rangle$ (homogeneity constraints) for each edge (c, c') in the resulting hierarchy schema.

In previous work [HM02] we provided an algorithm to compute the set of frozen dimensions of a schema in exponential time on the size of the schema. Thus, we can prove:

Input: A dimension schema $D = (H, \Sigma)$
Output: A homogeneous dimension schema D', such that D is h-equivalent to D'
(1) Let C_{ini} be the set of categories of D, $\Sigma_H := \emptyset$; $T := \emptyset$; $n := 0$
(2) Compute the set of frozen dimensions F of D
(3) For every category $c \in C_{ini}$ do
(4) For every non-empty set S of categories connected from c do
(5) Split c into $\texttt{CatName}(n)$ and c in H
(6) For each frozen dimension f of F, if $f \models \bigwedge_{c_i \in S} \langle c, c_i \rangle \wedge \bigwedge_{\{c_j : c \nearrow c_j\} \setminus S} \langle c, c_j \rangle$ then rename c with $\texttt{CatName}(n)$ in f;
(7) $n := n + 1$
(8) EndFor
(9) EndFor
(10) Scan the set F deleting the empty edges from H and adding equality and homogeneity constraints to Σ'
(11) Return (H, Σ')
End

Fig. 8. Algorithm that transforms a dimension schema into an h-equivalent homogeneous schema.

Proposition 2. *Assume the set of frozen dimensions of a schema are computed, then the algorithm of Figure 8 runs in time $O(NF2^N)$, where N is the size of the hierarchy schema, and F is the number of frozen dimensions.*

5.4 Algorithmic Aspects of Testing Hierarchical Equivalence

From the proof of Theorem 4, we can derive an algorithm for testing h-equivalence and prove that this problem is decidable. The naive application of the procedure in the proof yields a double exponential time algorithm. In fact, we can test whether $D \equiv_h D'$ in the following two steps: (1) apply the transformation in step (4) in the proof to transform D into D_f and D' into D'_f; and (2) test whether H_f is graph isomorphic to H'_f, where H_f (resp. H'_f) is the hierarchy schema of D_f (resp. D'_f).

The number of categories in the frozen schemas is in $O(n2^nK)$, where n is the number of categories and K is the number of constants mentioned in the schema. This bound is the order of the number of splits used in each transformation. Essentially, we may have as many categories in the resulting schema as frozen dimensions in the original schema. Since the size of D_f (resp. D'_f) is exponential in the size of the initial schema D (resp. D') we get the stated bound due to the test in 2.[1]

The following result shows that the problem is hard.

[1] DAG isomorphism is graph isomorphism complete. Recall that the "exact" complexity of deciding whether two graphs are isomorphic is still not known. The problem has neither been proved to be NP complete nor in P.

Theorem 5 (Testing h-Equiv.). *Testing whether two dimension schemas D and D' are h-equivalent is co-NP hard.*

The proof is a reduction of VALIDITY (given a proposition P, is P satisfied by all truth assignments?) to this problem.

We end this section by sketching an exponential time algorithm for testing h-equivalence: (1) compute the frozen dimensions of D and D'; and (2) for every binary relation between categories, test whether it induces a bijective frozen mapping.

Step 1 can be done in exponential time on the size of the schema. (See [HM02] for detailed bounds.) The number of binary relations between categories we need to test in Step 2 is $O(2^{n^2})$. For each such relation, we have to compute the induced frozen relation R, i.e. we need to test for each pair of frozen dimensions $d \in \texttt{Frozen}(D)$ and $d' \in \texttt{Frozen}(D')$ whether $(d, d') \in R$. This test can be done in 2^{n^2} operations of $O(n)$ steps each, since we need to check at most 2^{n^2} possible isomorphisms between d and d'. Also, we have to perform one test for each pair of frozen dimensions d and d'. Since the number of frozen dimensions of a given schema is exponential in the size of the schema, Step 2 can be accomplished in time exponential on the size of the schemas.

6 Conclusion and Further Work

In this paper we have presented a series of results that give conceptual insights into the problem of modeling OLAP dimension schemas. In particular, our framework: allowed us to compare different classes of dimension schemas introduced in a variety of OLAP models; and provides a formal basis to further research on schema restructuring in OLAP warehouses.

Dimension schemas enriched with dimension constraints give users flexibility to choose among several options the best suited for the application at hand. Further work needed to turn this flexibility into practical OLAP applications includes the definition of normal forms, restructuring operators, and implementation issues.

Acknowledgments We thank Alberto Mendelzon for fruitful suggestions on early versions of this work. This research was supported by Millenium Nucleus, Center for Web Research (P01-029-F), Mideplan, Chile.

References

[Alb00] J. Albert. Theoretical foundations of schema restructuring in heterogeneous multidatabase systems. In *Proceedings of the ACM Conference on Information and Knowledge Management*, Washington, DC, USA, 2000.

[AV97] S. Abiteboul and V. Vianu. Regular path queries with path constraints. In *Proceedings of the 16th ACM Symposium on Principles of Database Systems*, Tucson, Arizona, USA, 1997.

[BFS98] P. Buneman, W. Fan, and Weinstein S. Path constraints on semistructured and structured data. In *Proceedings of the 17th ACM Symposium on Principles of Database Systems*, Seattle, Washington, USA, 1998.

[CD97] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. In *ACM SIGMOD Record 26(1)*, March 1997.

[CT97] L. Cabibbo and R. Torlone. Querying multidimensional databases. In *Proceedings of the 6th International Workshop on Database Programming Languages*, East Park, Colorado, USA, 1997.

[Gol81] B. A. Goldstein. Constraints on null values in relational databases. In *Proceedings of the 7th International Conference on Very Large Data Bases*, Cannes, France, 1981.

[HG03] C. Hurtado and C. Gutierrez. Equivalence of OLAP dimension schemas (extended version). In *Technical Report, Departamento de Ciencias de la Computacin, Universidad de Chile, TR/DCC-2003-7*, 2003.

[HGM03] C. Hurtado, C. Gutiérrez, and A. Mendelzon. Capturing summarizability with integrity constraints in OLAP. In *Technical Report, Departamento de Ciencias de la Computacin, Universidad de Chile, TR/DCC-2003-6*, 2003.

[HLV00] B. Huseman, J. Lechtenborger, and G. Vossen. Conceptual data warehouse design. In *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW)*, Stockholm, Sweden, 2000.

[HM02] C. Hurtado and A. Mendelzon. OLAP dimension constraints. In *Proc. PODS 2002*, Madison, USA, 2002.

[HMV99] C. Hurtado, A. Mendelzon, and A. Vaisman. Maintaining data cubes under dimension updates. In *Proceedings of the 15th IEEE International Conference on Data Engineering (ICDE)*, Sydney, Australia, 1999.

[Hul86] R. Hull. Relative information capacity of simple relational database schemata. In *SIAM Journal of Computing 15(3):865-886*, 1986.

[Hur02] C. Hurtado. Structurally heterogeneous OLAP dimensions. In *Ph.D. Thesis, Department of Computer Science, University of Toronto*, 2002.

[JLS99] H. V. Jagadish, L. V. S. Lakshmanan, and D. Srivastava. What can hierarchies do for data warehouses? In *Proc. of the 25th International Conference on Very Large Data Bases*, Edinburgh, Scotland, UK, 1999.

[LAW98] W. Lehner, H. Albrecht, and H. Wedekind. Multidimensional normal forms. In *Proceedings of the 10th Statistical and Scientific Database Management Conference*, Capri, Italy., 1998.

[MIR94] R. Miller, Y. Ioannidis, and R. Ramakrishnan. Schema equivalence in heterogeneous systems: Bridging theory and practice. In *Information Systems, vol. 19, no.1*, 1994.

[PJE99] T. B. Pedersen, C. S. Jensen, and Dyreson C. E. Extending practical preaggregation in on-line analytical processing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, Edinburgh, Scotland, 1999.

[VL00] Millist W. Vincent. and Mark Levene. Restructuring partitioned normal form relations without information loss. *SIAM Journal on Computing*, 29(5):1550–1567, 2000.