Logic Programs for Repairing Inconsistent Dimensions in Data Warehouses

Loreto Bravo[1], **Mónica Caniupán**[2], and **Carlos Hurtado**[3]

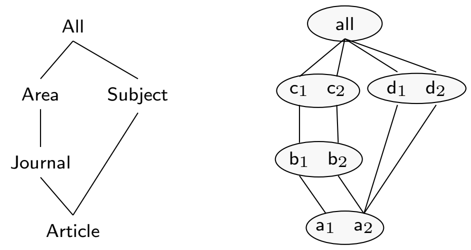
[1] Universidad de Concepción, Chile
[2] Universidad de Bio-Bio, Chile
[3] Universidad Adolfo Ibáñez

Abstract. A Data Warehouse (DW) is a data repository that integrates data from multiple sources and organizes the data according to a set of data structures called dimensions. Each dimension provides a perspective upon which the data can be viewed. In order to support an efficient processing of queries, a dimension is usually required to satisfy different classes of integrity constraints. In this paper, we study the problem of repairing a dimension when it fails to satisfy a set of two classes of integrity constraints: *strictness constraints* and *covering constraints*. We introduce the notion of minimal repair of a dimension in this context. A minimal repair is defined as a new dimension that is consistent with respect to the integrity constraints, which is obtained by applying a minimal amount of updates to the original dimension. We study the complexity of computing minimal repairs. Finally, we show how to characterize and compute minimal repairs of a dimension using Datalog programs with stable model semantics.

1 Introduction

Data Warehouses (DWs) are data repositories that integrate data from different sources, and keep historical data for analysis and decision support [10]. When generating reports, it is of central importance for these systems to compute queries that are summaries of data in a simple and efficient way. In order to do this, DWs organize data according to the multidimensional model. In the multidimensional model, dimensions reflect the perspectives upon which facts are viewed. Facts correspond to events which are usually associated to numeric values known as *measures*, and are referenced using the dimension elements. Dimensions are modeled as hierarchies whose nodes are called elements, where each element belongs to a category. The categories are also organized into a hierarchy called hierarchy schema.

Example 1. Consider a company that manages an online repository of research articles. The company maintains a DW to generate summaries used to analyze the download behavior of its users. The DW contains the dimensions Time and Publication. The Time dimension is structured using a hierarchy schema with a bottom category Date, which goes to the category Month, which in turn goes to the category Year. On the other hand, the Publication dimension is structured using the hierarchy schema shown in Figure 1(a), where the Article category goes to Journal, which in turn goes to Area. Also the Article category is connected



(a) Publication dimension: hierarchy schema (b) Publication dimension: elements and rollup relations

Downloads		
Article	Date	N
a_1	01-01-2007	3
a_2	01-01-2007	2
a_1	02-01-2007	5
a_2	02-01-2007	5

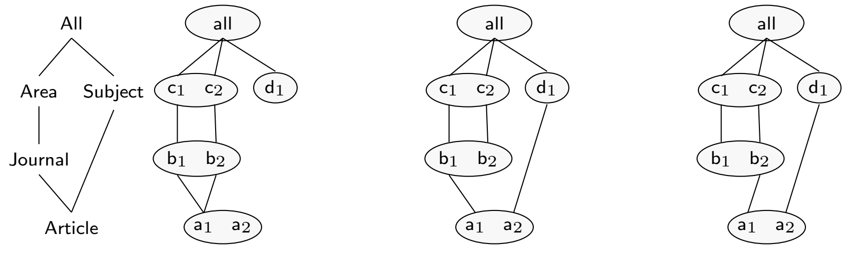
(c) Fact table

Fig. 1. Dimension for Example 1

to Subject category. The top category is All. Figure 1(b) shows the elements of the Publication dimension, along with the relations between elements of different categories, called rollup relations. For example, a_1 and a_2 are elements of category Article and b_1, b_2 are elements of category Journal. For each edge in the hierarchy schema, there is a rollup relation. For example, the rollup relation from the category Article to the category Journal contains the pairs or elements: (a_1, b_1) and (a_2, b_2). The fact table of the DW is shown in Figure 1(c). Each fact stored in the table represents the numbers of times an article was downloaded in a given date. As an example, article a_1 was downloaded three times on January 1, 2007. The hierarchical structure of dimensions allows users to access facts at different levels of granularity. As an example, using the aforementioned DW it is easy to compute summaries such as: number of times that each article was downloaded per month, or number of downloads broken down by area and year. □

In order to compute summaries efficiently, DWs use pre-computed summaries at low level categories to derive summaries at higher level categories. Two main classes of integrity constraints, *strictness* and *covering constraints*, are used to check whether such computations, called summarizations, are correct [13, 19, 25]. In a consistent summarization, each fact is aggregated once and not more than once. Strictness constraints are used to require rollup relations to be strict (many-to-one) relations. If a rollup relation is required to be strict, then there cannot exist an element connected to two different elements in the rollup relation. As an example, in the Publication dimension (Figure 1(a) and (b)) the rollup relation from Article to Journal is strict. In contrast, the rollup relation from Article to Subject is not strict, because a_2 is connected to two different elements in the category Subject. Covering constraints are used to require a rollup relation to connect all the elements that belongs to the category from which the rollup relation departs. As an example, in the dimension of Figure 1(b), the rollup relation from Article to Subject it is not covering since the element a_1 is not connected to any element in Subject.

Example 2. The summary $S_{\mathsf{Area}}(\mathsf{Area}, N) = \{(c_1, 8),\ (c_2, 7)\}$ (Figure 1) can be correctly derived from the summary $S_{\mathsf{Article}}(\mathsf{Article}, N) = \{(a_1, 8), (a_2, 7)\}$ by summing up the number of downloads for each group of articles that go to the same area in the dimension (element a_1 goes to c_1, and a_2 goes to c_2). The correctness of this derivation follows from the fact that the rollup relation from



(a) Dimension $\mathcal{D}$ (b) Repair $\mathcal{D}_1$ (c) Repair $\mathcal{D}_2$

Fig. 2. A Publication dimension inconsistent with respect to Article$\rightarrow$Area and Article$\Rightarrow$All

Article to Area is strict and covering. Similarly, S_{Area} can be correctly derived from $S_{\mathsf{Journal}}(\mathsf{Journal}, N) = \{(\mathsf{b}_1, 8), (\mathsf{b}_2, 7)\}$. However, the summary $S_{\mathsf{All}}(\mathsf{All}, N) = \{(\mathsf{all}, 15)\}$ cannot be correctly derived from $S_{\mathsf{Subject}}(\mathsf{Subject}, N) = \{(\mathsf{d}_1, 0), (\mathsf{d}_2, 7)\}$ since such derivation would yield $S_{\mathsf{All}}(\mathsf{All}, N) = \{(\mathsf{all}, 7)\}$. This derivation is incorrect because the rollup relation from Article to Subject is not covering. □

It has been shown that the dimensions that arise in real-world applications do not always have strict and covering rollup relations. The flexibility to represent rollup relations has been incorporated as a central feature of several dimension models [13, 24]. In order to keep the ability to check consistency of summarizations, it is important for DW systems to know the set of strictness and covering constraints that hold. This can be achieved by allowing DW designers to formulate the constraints that must hold when the dimensions are modeled. In this paper, we study the form of inconsistency that arises when a dimension does not satisfy a set of strictness and covering constraints. Similar to the vast majority of the work on inconsistency handling in databases (cf. [4]), we address the scenario where the constraints prevail over the data and the dimension must be updated in order for it to satisfy the constraints.

Example 3. Suppose that the DW administrator considers that the following constraints should be satisfied by the dimension $\mathcal{D}$ in Figure 2: φ_1 : Article $\Rightarrow$ All, φ_2 : Journal $\Rightarrow$ All, φ_3 : Area $\Rightarrow$ All, φ_4 : Subject $\Rightarrow$ All, which state that the rollup relations from categories Article, Journal, Area and Subject to All are covering, and the constraint φ_5 : Article $\rightarrow$ Area establishing that the rollup relation from Article to Area is strict. However, dimension $\mathcal{D}$ fail to satisfy constraint φ_5, since element a_1 goes to two different elements: c_1 and c_2 in the Area category. Also, $\mathcal{D}$ violates φ_1 because element a_2 does not reach all. The DW administrator considers that the constraints properly capture the semantics of the data, and therefore their violation indicates the presence of errors. Therefore, the dimension should be repaired (corrected) in order to resolve the inconsistency. Figure 2(b)-(c) are two possible repairs of $\mathcal{D}$. Repair $\mathcal{D}_1$ is obtained by deleting edge $(\mathsf{a}_1, \mathsf{b}_2)$ to remove the violation of φ_5 and inserting $(\mathsf{a}_2, \mathsf{d}_1)$ to remove the violation of φ_1. Repair $\mathcal{D}_2$ is obtained by deleting edge $(\mathsf{a}_1, \mathsf{b}_1)$ and inserting $(\mathsf{a}_2, \mathsf{d}_1)$. □

We introduce the notion of a minimal repair of a dimension that does not satisfy a set of constraints. A minimal repair is defined as a new dimension that satisfies the constraints and is obtained by applying a minimal number of changes to the original dimension. We study the complexity of the problem of computing minimal repairs, and show that in general the problem is NP-hard. We also explore the use of Datalog programs with stable model semantics [11] to characterize minimal repairs. Logic programs act as a compact and executable logical specification of dimension repairs. The rest of the paper is organized as follows: Section 2 presents DW dimensions, and strictness and covering constraints. Next, Section 3 presents the notion of repair and describes complexity issues. Section 4 presents the logic programs to compute repairs. Finally, Sections 5 and 6 present related work and the conclusions of the paper.

2 Preliminaries

In this section we formalize dimensions using standard concepts from multidimensional models obtained from [9, 15, 17, 24], with some minor modifications to facilitate presentation.

A *hierarchy schema* $\mathcal{H}$ consists of a pair $(\mathcal{C}_{\mathcal{H}}, \nearrow_{\mathcal{H}})$, where $(\mathcal{C}_{\mathcal{H}}, \nearrow_{\mathcal{H}})$ is an acyclic directed graph. Vertices in the set $\mathcal{C}_{\mathcal{H}}$ are categories and the edges $\nearrow_{\mathcal{H}}$ represent the child/parent relations between categories. The transitive and reflexive closure of $\nearrow_{\mathcal{H}}$ is denoted by $\nearrow^*_{\mathcal{H}}$. The set of categories $\mathcal{C}_{\mathcal{H}}$ contains a distinguished *top* category denoted $\mathsf{All}_{\mathcal{H}}$, which is reachable from every other category in $\mathcal{C}_{\mathcal{H}}$ and has no outgoing edges, that is, there is no category $\mathsf{c}_i \in \mathcal{C}_{\mathcal{H}}$ such that $(\mathsf{All}_{\mathcal{H}}, \mathsf{c}_i) \in \nearrow_{\mathcal{H}}$ and for every $\mathsf{c}_j \in \mathcal{C}$, $(\mathsf{c}_j, \mathsf{All}_{\mathcal{H}}) \in \nearrow^*_{\mathcal{H}}$. Sometimes, we will write $c_a \nearrow_{\mathcal{H}} c_b$ instead of $(c_a, c_b) \in \nearrow_{\mathcal{H}}$.

Example 4. The hierarchy schema $\mathcal{H} = (\mathcal{C}_{\mathcal{H}}, \nearrow_{\mathcal{H}})$, depicted in Figure 1(a), is as follows: $\mathcal{C}_{\mathcal{H}} = \{$Article, Journal, Subject, Area, All$\}$; $\mathsf{All}_{\mathcal{H}} =$ All; and $\nearrow_{\mathcal{H}}=\{$ (Article, Journal), (Journal, Area), (Area, All), (Article, Subject), (Subject, All)$\}$. □

A *dimension* $\mathcal{D}$ is a tuple $(\mathcal{H}_{\mathcal{D}}, \mathcal{E}_{\mathcal{D}}, \mathsf{Cat}_{\mathcal{D}}, <_{\mathcal{D}})$, where $\mathcal{H}_{\mathcal{D}} = (\mathcal{C}_{\mathcal{H}_{\mathcal{D}}}, \nearrow_{\mathcal{H}_{\mathcal{D}}})$ is a hierarchy schema; $\mathcal{E}_{\mathcal{D}}$ is a set of constants, called elements; $\mathsf{Cat}_{\mathcal{D}} : \mathcal{E}_{\mathcal{D}} \rightarrow \mathcal{C}_{\mathcal{H}_{\mathcal{D}}}$ is a function that defines to which category each element in $\mathcal{E}_{\mathcal{D}}$ belongs to; and the relation $<_{\mathcal{D}} \subseteq \mathcal{E}_{\mathcal{D}} \times \mathcal{E}_{\mathcal{D}}$ represents the child/parent relations between elements of different categories. We denote by $<^*_{\mathcal{D}}$ the reflexive and transitive closure of $<_{\mathcal{D}}$. The following conditions hold: (i) $\mathsf{all}_{\mathcal{D}}$ is the only element in category $\mathsf{All}_{\mathcal{H}_{\mathcal{D}}}$ (ii) for all pair of elements $a, b \in \mathcal{E}_{\mathcal{D}}$ if $a <_{\mathcal{D}} b$ then $\mathsf{Cat}_{\mathcal{D}}(a) \nearrow_{\mathcal{H}_{\mathcal{D}}} \mathsf{Cat}_{\mathcal{D}}(b)$. Condition (ii) ensures that the child/parent relation ($<_{\mathcal{D}}$) only connects elements of categories that are connected in the schema.

Example 5. Let $\mathcal{D} = (\mathcal{H}_{\mathcal{D}}, \mathcal{E}_{\mathcal{D}}, \mathsf{Cat}_{\mathcal{D}}, <_{\mathcal{D}})$ be the dimension given in Figure 1(b). Then $\mathcal{D}$ is as follows $\mathcal{E}_{\mathcal{D}} = \{\mathsf{all}, \mathsf{a}_1, \mathsf{a}_2, \mathsf{b}_1, \mathsf{b}_2, \mathsf{c}_1, \mathsf{c}_2, \mathsf{d}_1, \mathsf{d}_2\}$; $\mathsf{all}_{\mathcal{D}} = \mathsf{all}$; $\mathsf{Cat}_{\mathcal{D}} = \{\mathsf{all} \mapsto \mathsf{All}, \mathsf{a}_1 \mapsto \mathsf{Article}, \mathsf{a}_2 \mapsto \mathsf{Article}, \mathsf{b}_1 \mapsto \mathsf{Journal}, \mathsf{b}_2 \mapsto \mathsf{Journal}, \mathsf{c}_1 \mapsto \mathsf{Area}, \mathsf{c}_2 \mapsto \mathsf{Area}, \mathsf{d}_1 \mapsto \mathsf{Subject}, \mathsf{d}_2 \mapsto \mathsf{Subject}, \}$; and $<_{\mathcal{D}}= \{(\mathsf{a}_1,\mathsf{b}_1), (\mathsf{a}_2,\mathsf{b}_2), (\mathsf{b}_1,\mathsf{c}_1), (\mathsf{b}_2,\mathsf{c}_2), (\mathsf{c}_1,\mathsf{all}), (\mathsf{c}_2,\mathsf{all}), (\mathsf{a}_2,\mathsf{d}_1), (\mathsf{a}_2,\mathsf{d}_2), (\mathsf{d}_1,\mathsf{all}), (\mathsf{d}_2,\mathsf{all})\}$. □

The set of rollup relations of a dimension $\mathcal{D}$ is defined as follows. For each pair of categories $\mathsf{c}_i, \mathsf{c}_j \in \mathcal{C}_{\mathcal{H}_{\mathcal{D}}}$ such that $\mathsf{c}_i \nearrow^*_{\mathcal{H}_{\mathcal{D}}} \mathsf{c}_j$, there is a rollup relation denoted by $\mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ that has the following set of pairs $\{(a, b) | \mathsf{Cat}_{\mathcal{D}}(a) = \mathsf{c}_i, \mathsf{Cat}_{\mathcal{D}}(b) = \mathsf{c}_j \text{ and } a <^*_{\mathcal{D}} b\}$.

Example 6. Dimension in Figure 1 has, for example, $\mathcal{R}_{\mathcal{D}}(\mathsf{Article}, \mathsf{Journal}) = \{(\mathsf{a_1},\mathsf{b_1}), (\mathsf{a_2},\mathsf{b_2})\}$ and $\mathcal{R}_{\mathcal{D}}(\mathsf{Article}, \mathsf{Subject}) = \{(\mathsf{a_2},\mathsf{d_1}), (\mathsf{a_2},\mathsf{d_2})\}$. $\square$

We say that the rollup relation $\mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ is *strict* if for all elements x, y, z in $\mathcal{E}$, if $(x, y) \in \mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ and $(x, z) \in \mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ then $y = z$. The rollup relation $\mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ is *covering* if for all elements $e \in \mathcal{E}$ such that $\mathsf{Cat}(e) = \mathsf{c}_i$, there exists an element $e' \in \mathcal{E}$ such that $(e, e') \in \mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$. A dimension is strict if all its rollup relations are strict. Otherwise, the dimension is said to be non-strict. Similarly, we use the notions of covering and non-covering dimension. Covering dimensions are also called homogeneous and non-covering dimensions are called heterogeneous.

The vast majority of research and industrial applications of DWs consider dimensions that are strict and covering [18]. For strict and covering dimensions [13] summarization operations that aggregate facts between two categories that are connected in the hierarchy schema are always correct. However, it has been shown that in some real situations dimensions fail to satisfy these conditions [14, 24]. In these cases, in order to keep the ability to verify summarizability, it is useful to allow the DW administrator to specify integrity constraints to identify rollup relations that are strict or covering [14].

Let $\mathcal{H} = (\mathcal{C}_{\mathcal{H}}, \nearrow_{\mathcal{H}})$ be a hierarchy schema and let $\mathcal{D} = (\mathcal{H}_{\mathcal{D}}, \mathcal{E}_{\mathcal{D}}, \mathsf{Cat}_{\mathcal{D}}, <_{\mathcal{D}})$ be a dimension such that $\mathcal{H}_{\mathcal{D}} = \mathcal{H}$.
(i) A *strictness constraint* over $\mathcal{H}$ is an expression of the form $\mathsf{c}_i \rightarrow \mathsf{c}_j$ where $\mathsf{c}_i, \mathsf{c}_j \in \mathcal{C}_{\mathcal{H}}$ and $\mathsf{c}_i \nearrow^*_{\mathcal{H}} \mathsf{c}_j$. The dimension $\mathcal{D}$ *satisfies* the strictness constraint $\mathsf{c}_i \rightarrow \mathsf{c}_j$ if and only if the rollup relation $\mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ is strict.
(ii) A *covering constraint* over $\mathcal{H}$ is an expression of the form $\mathsf{c}_i \Rightarrow \mathsf{c}_j$ where $\mathsf{c}_i, \mathsf{c}_j \in \mathcal{C}_{\mathcal{H}}$ and $\mathsf{c}_i \nearrow^*_{\mathcal{H}} \mathsf{c}_j$. The dimension $\mathcal{D}$ *satisfies* the covering constraint $\mathsf{c}_i \Rightarrow \mathsf{c}_j$ if and only if the rollup relation $\mathcal{R}_{\mathcal{D}}(\mathsf{c}_i, \mathsf{c}_j)$ is covering.
We denote by $\Sigma_s(\mathcal{H})$ and $\Sigma_c(\mathcal{H})$ the set of all possible strictness constraints and covering constraints, respectively, over hierarchy schema $\mathcal{H}$. We say that a dimension $\mathcal{D}$ satisfies a set of constraints Σ if $\mathcal{D}$ satisfies every constraint in Σ. Otherwise, we say that the dimension $\mathcal{D}$ is inconsistent with respect to Σ.

Example 7. Let $\mathcal{D}$ be the publication dimension depicted in Figure 1(b). This dimension satisfies the following set Σ of strictness constraints: $\{\mathsf{Article} \rightarrow \mathsf{Journal}, \mathsf{Article} \rightarrow \mathsf{Area}, \mathsf{Journal} \rightarrow \mathsf{Area}\}$. However, $\mathcal{D}$ does not satisfy $\mathsf{Article} \rightarrow \mathsf{Subject}$, since an element of $\mathsf{Article}$ may reach more than one element in $\mathsf{Subject}$. In contrast, dimension given in Figure 2 is inconsistent with respect to Σ since it violates the strictness constraint $\mathsf{Article} \rightarrow \mathsf{Area}$. Indeed, the rollup relation $\mathcal{R}_{\mathcal{D}}(\mathsf{Article}, \mathsf{Area}) = \{(\mathsf{a_1}, \mathsf{c_1}), (\mathsf{a_1}, \mathsf{c_2})\}$ is non-strict. $\square$

The problem of determining if a dimension $\mathcal{D}$ satisfies a set of constraints can be solved in polynomial time by computing $<^*_{\mathcal{D}}$.

3 Dimension Repairs

If a dimension $\mathcal{D}$ does not satisfy a set of constraints Σ, we propose to compute the minimal repairs of $\mathcal{D}$, that is, dimensions over the hierarchy schema of $\mathcal{D}$ that satisfy Σ and that stay as close as possible to the inconsistent dimension.

Example 8. Let $\mathcal{D}$ be the dimension given in Figure 2, and let $\Sigma = \{\mathsf{Article} \Rightarrow \mathsf{All}, \mathsf{Journal} \Rightarrow \mathsf{All}, \mathsf{Area} \Rightarrow \mathsf{All}, \mathsf{Subject} \Rightarrow \mathsf{All}, \mathsf{Article} \rightarrow \mathsf{Area}\}$. Clearly, $\mathcal{D}$ is inconsistent with respect to Σ. Figure 2(b)-(c) shows repairs of the dimension $\mathcal{D}$ with respect to Σ. Repair $\mathcal{D}_1$ is obtained by deleting edge (a_1,b_2) and inserting $(\mathsf{a}_2,\mathsf{d}_1)$, and repair $\mathcal{D}_2$ is obtained by deleting (a_1,b_1) and inserting $(\mathsf{a}_2,\mathsf{d}_1)$. There are other possible repairs, for instance, a repair $\mathcal{D}_3$ can be obtained by deleting edge $(\mathsf{b}_1,\mathsf{c}_1)$ and inserting edges $(\mathsf{b}_1,\mathsf{c}_2)$ and $(\mathsf{a}_2,\mathsf{d}_1)$. Even though repair $\mathcal{D}_3$ get rid of the inconsistencies, repairs $\mathcal{D}_1$, and $\mathcal{D}_2$ remain closer to dimension $\mathcal{D}$. $\Box$

Given two dimensions $\mathcal{D} = (\mathcal{H}_\mathcal{D}, \mathcal{E}_\mathcal{D}, \mathsf{Cat}_\mathcal{D}, <_\mathcal{D})$ and $\mathcal{D}' = (\mathcal{H}_{\mathcal{D}'}, \mathcal{E}_{\mathcal{D}'}, \mathsf{Cat}_{\mathcal{D}'}, <_{\mathcal{D}'})$, the distance between them, $\mathit{dist}(\mathcal{D}, \mathcal{D}')$, is defined as $|(<_{\mathcal{D}'} \setminus <_\mathcal{D}) \cup (<_\mathcal{D} \setminus <_{\mathcal{D}'})|$. The distance $\mathit{dist}(\mathcal{D}, \mathcal{D}')$ is the size of the symmetric difference between the child/parent relations of the two dimensions. Next, we define the notions of repair and minimal repair.

Definition 1. [Repair, Minimal Repair] Let $\mathcal{D} = (\mathcal{H}_\mathcal{D}, \mathcal{E}_\mathcal{D}, \mathsf{Cat}_\mathcal{D}, <_\mathcal{D})$ be a dimension and Σ be a set of integrity constraints over $\mathcal{H}_\mathcal{D}$.
(i) A *repair* of $\mathcal{D}$ with respect to Σ is a dimension $\mathcal{D}' = (\mathcal{H}_{\mathcal{D}'}, \mathcal{E}_{\mathcal{D}'}, \mathsf{Cat}_{\mathcal{D}'}, <_{\mathcal{D}'})$ such that $\mathcal{H}_{\mathcal{D}'} = \mathcal{H}_\mathcal{D}$, $\mathcal{E}_{\mathcal{D}'} = \mathcal{E}_\mathcal{D}$, $\mathsf{Cat}_{\mathcal{D}'} = \mathsf{Cat}_\mathcal{D}$, and $\mathcal{D}'$ satisfies Σ.
(ii) A *minimal* repair of $\mathcal{D}$ with respect to Σ is a repair $\mathcal{D}'$, such that $\mathit{dist}(\mathcal{D}, \mathcal{D}')$ is minimal among all the repairs of $\mathcal{D}$ with respect to Σ. $\Box$

In the notion of repair we propose, the set of elements in each category of the original dimension is preserved over all the repairs, that is deletions or additions of new elements are not allowed.

Example 9. The distance between $\mathcal{D}$ and the repairs in Figure 2(b)-(c) are $\mathit{dist}(\mathcal{D}, \mathcal{D}_1) = \mathit{dist}(\mathcal{D}, \mathcal{D}_2) = 2$. Indeed, $\mathcal{D}_1$, and $\mathcal{D}_2$ are the minimal repairs of $\mathcal{D}$ with respect to Σ. $\Box$

It can be shown that there always exists a repair of a dimension provided that the dimension has at least one element in each category. However, there might be an exponential number of them. Thus, it becomes relevant to study the problem of finding one.

Theorem 1. Let $\mathcal{D}$ be a dimension, and k an integer. The problem of deciding if there exists a repair $\mathcal{D}'$ of $\mathcal{D}$ such that $\mathit{dist}(\mathcal{D}, \mathcal{D}') \leq k$ is NP-complete. $\Box$

Hardness is proven by reduction from the set covering problem. It follows then, that the problem of computing a minimal repair is NP-hard and that deciding if a dimension is a minimal repair is co-NP-complete.

4 Computing Repairs

Computing repairs of dimensions inconsistent with respect to a set of covering and strictness constraints can be expensive in terms of computational resources. However, we can use Datalog programs with negation under stable model semantics [11] to represent and compute the minimal repairs of a dimension. Because of space limitations, we will present here the programs with examples.

Even though Datalog programs with negation under stable models semantics can be used to express NP problems, current implementations, such as DLV[4] [20] and Smodels[5] [26], have shown to be quite efficient in practice [20]. In what follows, we will present a Datalog program with negation using predicates in Table 1 that can be used to obtain the minimal repairs of a dimension.

Atom	**Interpretation**
$C(a)$	element a belongs to category C in dimension $\mathcal{D}$
$R(a, b, C_1, C_2)$	in $\mathcal{D}$, the parent of element a is b, $\mathsf{Cat}_{\mathcal{D}}(a) = \mathcal{C}_1$ and $\mathsf{Cat}_{\mathcal{D}}(b) = C_2$
$R'(a, b, C_1, C_2)$	in $\mathcal{D}'$, the parent of element a is b, $\mathsf{Cat}_{\mathcal{D}}(a) = C_1$ and $\mathsf{Cat}_{\mathcal{D}}(b) = C_2$
$RT'(a, b, C_1, C_2)$	transitive closure of R'
$Ins(a, b, C_1, C_2)$	the child/parent relation (a, b) with $\mathsf{Cat}_{\mathcal{D}}(a) = C_1$ and $\mathsf{Cat}_{\mathcal{D}}(b) = C_2$, was inserted to get $\mathcal{D}'$
$Del(a, b, C_1, C_2)$	the child/parent relation (a, b) with $\mathsf{Cat}_{\mathcal{D}}(a) = C_1$ and $\mathsf{Cat}_{\mathcal{D}}(b) = C_2$, was removed to get $\mathcal{D}'$

Table 1. *Predicates of the repair program $\Pi(\mathcal{D}, \Sigma)$ that computes a repair $\mathcal{D}'$ of $\mathcal{D}$ with respect to Σ*

In order to simplify the presentation, we will first discuss the case where $\Sigma = \Sigma_s(\mathcal{H}) \cup \Sigma_c(\mathcal{H})$, that is we have to fix a dimension that, according to the constraints, should be strict and covering. Let us observe that a covering and strict dimension is such that every element will rollup to exactly one element in each parent category. The repair program relies on this observation and first computes a set of candidate dimensions that contains the same elements as the inconsistent dimension and only one rollup between every element and every parent category (note that all these candidate dimensions satisfy the covering constraints). Then, it discards the models that do not satisfy the strictness constraints.

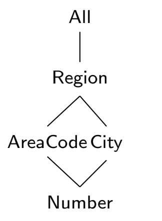
Example 10. Consider the Phone dimensions $\mathcal{D}_{Phone}$ in Figure 3. Even though the dimension is expected to be covering and strict, the dimension is not since the strictness constraint Number $\rightarrow$ Region and the covering constraint Number $\Rightarrow$ AreaCode are violated. This is, $\mathcal{D}_{Phone}$ is inconsistent with respect to the set of constraints $\Sigma_c(\mathcal{H}) \cup \Sigma_s(\mathcal{H})$. The repair program to compute the minimal repairs contains the following facts and rules:

Number(N_1). Number(N_2). Number(N_3). AreaCode(45). AreaCode(41).
City(TCH). City(TEM). City(CCP). All(all). Region(VIII).
Region(IX). R(N_1,41,Number,AreaCode). R(N_3,45,Number,AreaCode).
R(N_1,TCH,Number,City). R(N_2,TEM,Number,City). R(N_3,CCP,Number,City).
R(45,IX,AreaCode,Region). R(41,VIII,AreaCode,Region). R(TCH,VIII,City,Region).
R(TEM,IX,City,Region). R(CCP,VIII,City,Region). R(IX,all,Region,All).
R(VIII,all,Region,All).

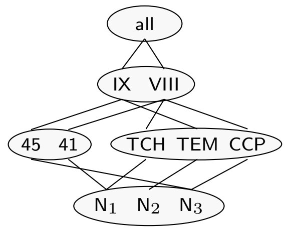
$$R'(X, Y, \mathsf{number}, \mathsf{areaCode}) \leftarrow \mathsf{Number}(X), \mathsf{AreaCode}(Y), \mathtt{choice}((X, \mathsf{areaCode})(Y)). \quad (1)$$
$$R'(X, Y, \mathsf{number}, \mathsf{city}) \leftarrow \mathsf{Number}(X), \mathsf{City}(Y), \mathtt{choice}((X, \mathsf{city})(Y)). \quad (2)$$
$$R'(X, Y, \mathsf{areaCode}, \mathsf{region}) \leftarrow \mathsf{AreaCode}(X), \mathsf{Region}(Y), \mathtt{choice}((X, \mathsf{region})(Y)). \quad (3)$$
$$R'(X, Y, \mathsf{city}, \mathsf{region}) \leftarrow \mathsf{City}(X), \mathsf{Region}(Y), \mathtt{choice}((X, \mathsf{region})(Y)). \quad (4)$$
$$R'(X, Y, \mathsf{areaCode}, \mathsf{all}) \leftarrow \mathsf{AreaCode}(X), \mathsf{All}(Y), \mathtt{choice}((X, \mathsf{all})(Y)). \quad (5)$$
$$R'(X, Y, \mathsf{city}, \mathsf{all}) \leftarrow \mathsf{City}(X), \mathsf{All}(Y), \mathtt{choice}((X, \mathsf{all})(Y)). \quad (6)$$

[4] DLV System: http://www.dbai.tuwien.ac.at/proj/dlv/

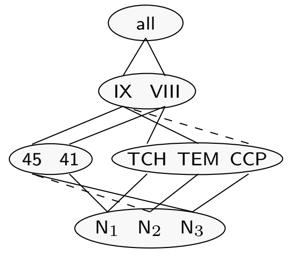
[5] Smodels System: http://www.tcs.hut.fi/Software/smodels/



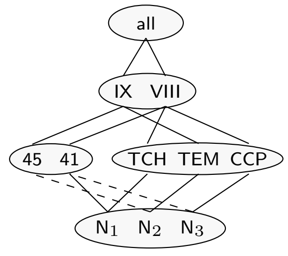
(a) Dimension $\mathcal{D}_{Phone}$: hierarchy schema



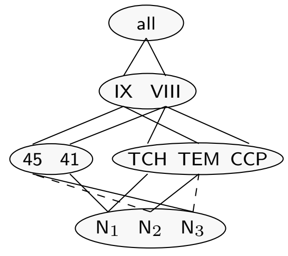
(b) Dimension $\mathcal{D}_{Phone}$: elements and rollup relations



(c) Repair $\mathcal{D}^1_{Phone}$



(d) Repair $\mathcal{D}^2_{Phone}$



(e) Repair $\mathcal{D}^3_{Phone}$

Fig. 3. Phone Dimension and repairs of it with respect to $\Sigma_c(\mathcal{H}) \cup \Sigma_s(\mathcal{H})$

$$RT'(X, Y, N_1, N_2) \leftarrow R'(X, Y, N_1, N_2). \tag{7}$$
$$RT'(X, Z, N_1, N_3) \leftarrow RT'(X, Y, N_1, N_2), R'(Y, Z, N_2, N_3). \tag{8}$$
$$\leftarrow RT'(X, Y, N_1, N_2), RT'(X, Z, N_1, N_2), Y \neq Z. \tag{9}$$
$$\mathit{Ins}(X, Y, N_1, N_2) \leftarrow R'(X, Y, N_1, N_2), \mathit{not}\ R(X, Y, N_1, N_2). \tag{10}$$
$$\mathit{Del}(X, Y, N_1, N_2) \leftarrow R(X, Y, N_1, N_2), \mathit{not}\ R'(X, Y, N_1, N_2). \tag{11}$$
$$\Leftarrow \mathit{Ins}(X, Y, N_1, N_2)[1:1]. \tag{12}$$
$$\Leftarrow \mathit{Del}(X, Y, N_1, N_2)[1:1]. \tag{13}$$

Facts of the repair program correspond to the elements and rollup relations in $\mathcal{D}_{Phone}$. The rest of the rules in the repair program are used to compute the repairs. In general terms, the program: generates all possible dimensions $\mathcal{D}' = (\mathcal{H}_{\mathcal{D}'}, \mathcal{E}_{\mathcal{D}'}, \mathsf{Cat}_{\mathcal{D}'}, <_{\mathcal{D}'})$ such that $\mathcal{H}_{\mathcal{D}'} = \mathcal{H}_{\mathcal{D}_{Phone}}$, $\mathcal{E}_{\mathcal{D}'} = \mathcal{E}_{\mathcal{D}_{Phone}}$, $\mathsf{Cat}_{\mathcal{D}'} = \mathsf{Cat}_{\mathcal{D}_{Phone}}$ and such that every element has a unique parent in every parent category (using rules (1) to (6)), then discards the ones that are not strict (with rules (7) to (9)) and finally chooses the ones that are at a minimal distance (using rules (10) to (13)).

More specifically, rules (1) to (6) populate predicate R' with a new covering child-parent relation for the elements in $\mathcal{E}_{\mathcal{D}_{Phone}}$. This new relation assigns to each element a a new parent in every category c_j such that $\mathsf{Cat}(a) \nearrow_{\mathcal{H}_{\mathcal{D}_{Phone}}} \mathsf{c}_j$. This parent is obtained by using the `choice` operator [12] that generates one model for each c such that $\mathsf{Cat}(c) = \mathsf{c}_j$. In fact, $\texttt{choice}((X, c_j), (Y))$ will assign a unique value to Y in each stable model for each combination (X, c_j).

Rules (7) and (9) compute the transitive closure of the child-parent relation of the repair. Rule (9) is a program constraint (a rule without head) which enforces that it is not possible to have that an element x rolls-up to two different parents in the same category. Namely, it discards all the models in which the rollup relations, captured in predicate RT', are not strict. Rules (10) and (11) keep track of the insertion and deletions, respectively, that are needed to obtain the repair. The weak constraints [8], denoted by $\Leftarrow$, in rules (12) and (13) are used to minimize the number of insertion and deletions needed to restore consistency. In general, a weak constraint is of the form $\Leftarrow \varphi \quad [\mathit{weight} : \mathit{level}]$, where φ is a conjunction of atoms, and weight and level are integers. If φ is true in the model the weak constraint is violated and the weight is added up to the cost of the level. The stable models which minimize the sum of the weights of the violated weak constraints are called the *best models* of the program.

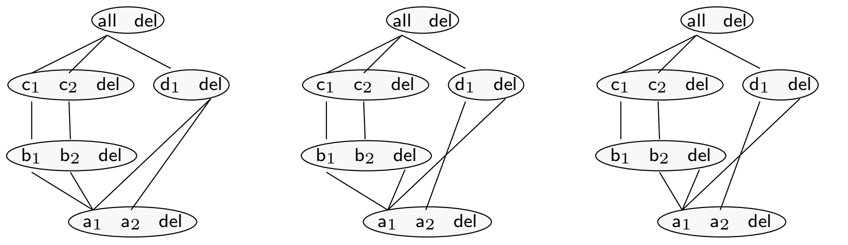
This program has three best models of total weight three[6]: $\mathcal{M}_1 = \{\mathit{Ins}(\mathsf{N2}, \mathsf{45}, \mathsf{Number}, \mathsf{AreaCode}), \mathit{Ins}(\mathsf{CCP}, \mathsf{IX}, \mathsf{City}, \mathsf{Region}), \mathit{Del}(\mathsf{CCP}, \mathsf{VIII}, \mathsf{City}, \mathsf{Region})\}$, $\mathcal{M}_2 = \{\mathit{Ins}(\mathsf{N3}, \mathsf{41}, \mathsf{Number}, \mathsf{AreaCode}), \mathit{Ins}(\mathsf{N2}, \mathsf{45}, \mathsf{Number}, \mathsf{AreaCode}), \mathit{Del}(\mathsf{N3}, \mathsf{45}, \mathsf{Number}, \mathsf{AreaCode})\}$ and $\mathcal{M}_3 = \{\mathit{Ins}(\mathsf{N3}, \mathsf{TEM}, \mathsf{Number}, \mathsf{City}), \mathit{Ins}(\mathsf{N2}, \mathsf{45}, \mathsf{Number}, \mathsf{AreaCode}), \mathit{Del}(\mathsf{N3}, \mathsf{CCP}, \mathsf{Number}, \mathsf{City})\}$. The total weight of each best model corresponds to the distance between each of the repairs and dimension $\mathcal{D}$. Each of these models corresponds indeed to the repairs of $\mathcal{D}_{Phone}$ with respect to $\Sigma_c(\mathcal{H}) \cup \Sigma_s(\mathcal{H})$ shown in Figure 3(c)-(e). □

The correctness of the program relies on the fact that in strict and covering dimensions every element contains exactly one parent in each parent category. In the general case, where the dimension is not necessarily required to be strict and covering, this does not hold anymore. However, in a minimal repair of a dimension $\mathcal{D}$, the number of parents in a category for every element can be shown to be between zero and the maximum between one (1) and the number of parents for that same element in $\mathcal{D}$. Thus, in the general case, in order to compute the minimal repairs, we have to be able to reduce the number of parents an element can have. This is achieved by constructing a new dimension $\mathcal{D}^{del}$ from $\mathcal{D}$ for which the repairs of $\mathcal{D}$ can be computed only through reclassification of child-parent relations of $\mathcal{D}^{del}$.

Example 11. Figure 4(a) shows the del-dimension $\mathcal{D}^{del}$ obtained from dimension $\mathcal{D}$ in Figure 2(a) which is inconsistent with respect to $\Sigma = \{\mathsf{Article} \Rightarrow \mathsf{All}, \mathsf{Journal} \Rightarrow \mathsf{All}, \mathsf{Area} \Rightarrow \mathsf{All}, \mathsf{Subject} \Rightarrow \mathsf{All}, \mathsf{Article} \rightarrow \mathsf{Area}\}$. From it we can compute a set of candidate repairs by reclassifying all the child/parent relations. From these candidate repairs, the only two that satisfy the constraints (ignoring the child/parent relations that involve any del-elements) are shown in Figure 4(b)-(c). If the del-elements are removed, these dimensions correspond to the repairs of $\mathcal{D}$ with respect to Σ shown in Figure 2(b)-(c). The repair program for $\mathcal{D}$ with respect to Σ contains the following facts and rules:

```
Article(a₁).     Article(a₂).     Article(del).     Journal(b₁).     Journal(b₂).
Journal(del).    Area(c₁).        Area(c₂).         Area(del).       Subject(d₁).
Subject(del).    All(all).        All(del).         R(a₁,b₁,Article,Journal).
```

[6] To simplify the presentation we show only the *Ins* and *Del* atoms which show the modifications needed to restore consistency.



(a) Dimension $\mathcal{D}^{del}$ (b) $\mathcal{D}_1$ with del elements (c) $\mathcal{D}_2$ with del elements

Fig. 4. del-Dimension for Example 11

R(a_1,b_2,Article,Journal). R(b_1,c_1,Journal,Area). R(b_2,c_2,Journal,Area).
R(c_1,all,Area,All). R(c_2,all,Area,All). R(d_1,all,Subject,All).
R(a_1,del,Article,Subject). R(a_2,del,Article,Subject).

$$R'(X, Y', Z, \mathsf{Journal}) \leftarrow R(X, Y, Z, \mathsf{Journal}), \mathsf{Journal}(Y'), \mathtt{choice}((X, Y, \mathsf{Journal})(Y')). \quad (14)$$
$$R'(X, Y', Z, \mathsf{Area}) \leftarrow R(X, Y, Z, \mathsf{Area}), \mathsf{Area}(Y'), \mathtt{choice}((X, Y, \mathsf{Area})(Y')). \quad (15)$$
$$R'(X, Y', Z, \mathsf{Subject}) \leftarrow R(X, Y, Z, \mathsf{Subject}), \mathsf{Subject}(Y'), \mathtt{choice}((X, Y, \mathsf{Subject})(Y')). \quad (16)$$
$$R'(X, Y', Z, \mathsf{All}) \leftarrow R(X, Y, Z, \mathsf{All}), \mathsf{All}(Y'), \mathtt{choice}((X, Y, \mathsf{All})(Y')). \quad (17)$$
$$RT'(X, Y, N_1, N_2) \leftarrow R'(X, Y, N_1, N_2), X \neq \mathsf{del}, Y \neq \mathsf{del}. \quad (18)$$
$$RT'(X, Z, N_1, N_3) \leftarrow RT'(X, Y, N_1, N_2), R'(Y, Z, N_2, N_3), X \neq \mathsf{del}, Y \neq \mathsf{del}, Z \neq \mathsf{del}. \quad (19)$$
$$\leftarrow RT'(X, Y, \mathsf{Article}, \mathsf{Area}), RT'(X, Z, \mathsf{Article}, \mathsf{Area}), Y \neq Z. \quad (20)$$
$$\leftarrow \mathsf{Article}(X), \mathit{not}\ \mathsf{aux}_{\mathsf{All}}(X), X \neq \mathsf{del}. \qquad \leftarrow \mathsf{Journal}(X), \mathit{not}\ \mathsf{aux}_{\mathsf{All}}(X), X \neq \mathsf{del}. \quad (21)$$
$$\leftarrow \mathsf{Area}(X), \mathit{not}\ \mathsf{aux}_{\mathsf{All}}(X), X \neq \mathsf{del}. \qquad \leftarrow \mathsf{Subject}(X), \mathit{not}\ \mathsf{aux}_{\mathsf{All}}(X), X \neq \mathsf{del}. \quad (22)$$
$$\mathsf{aux}_{\mathsf{All}}(X) \leftarrow RT'(X, Y, \mathsf{Area1}, \mathsf{All}). \quad (23)$$
$$\mathit{Ins}(X, Y, N_1, N_2) \leftarrow R'(X, Y, N_1, N_2), \mathit{not}\ R(X, Y, N_1, N_2), Y \neq \mathsf{del}. \quad (24)$$
$$\mathit{Del}(X, Y, N_1, N_2) \leftarrow R(X, Y, N_1, N_2), \mathit{not}\ R_(X, Y, N_1, N_2), Y \neq \mathsf{del}. \quad (25)$$
$$\Leftarrow \mathit{Ins}(X, Y, N_1, N_2).[1:1] \qquad \Leftarrow \mathit{Del}(X, Y, N_1, N_2).[1:1]. \quad (26)$$

The facts of the repair program correspond to the elements and rollup relations in $\mathcal{D}^{del}$. Rules (14) to (17) compute a new candidate child-parent relation for the elements in $\mathcal{E}_{\mathcal{D}^{del}}$. Then, the consistency of each candidate repair with respect to strictness (rules (18) to (20)) and covering (rules (21) to (23)) constraints is checked ignoring every child/parent relation with a del-element. Next, rules (24) and (26) chooses the ones that are at a minimal distance. Finally, the repairs are obtained from the consistent candidate repairs by removing every del-element. The best models of this program correspond to the repairs of $\mathcal{D}$ as shown in Figure 2(b)-(c). □

5 Related Work

Letz et al. [21] motivate the importance of enforcing strictness constraints in dimensions using constraints represented in dimensions hierarchies. In particular, the constraints are used to keep dimensions strict upon successive update operations. Pedersen et al. [23] present a method to model non-strict dimensions as strict dimensions when the OLAP implementation requires the dimensions to be strict. In our setting, in contrast, the objective is to clean errors that turn a dimension inconsistent with respect to a set of constraints.

The problem of repairing relational databases with respect to a set of integrity constraints (e.g. functional dependencies and inclusion dependencies) has been extensively studied (see [4] for a survey). Even though there are several ways to represent a data warehouse using relations (e.g. star schema or snowflake schema [10, 18]) it is not possible to use the relational repair techniques to compute DW's repairs. This is because, it is either not possible to represent the required strictness and covering constraints with the relational constraints, or the minimal relational repairs obtained by tuple deletions, insertions or updates do not coincide with the minimal repairs of DWs. Indeed, they might result in the removal of more roll-up relations than needed or in discarding repairs that are minimal in the DWs sense, but not in the relational case.

Logic programs have been used to specify the repairs of relational databases with respect to relational integrity constraints (e.g. functional dependencies, foreign key constraints) [2, 3, 7]. As far as we know, this paper is the first work on the application of logic programs to restore consistency of DWs.

6 Conclusions

In this paper we propose logic programs to compute repairs of dimensions, which are obtained by performing insertions and deletions of edges between dimension elements. We do not consider the insertion or elimination of dimension elements, options that might be helpful to restore consistency of dimensions. We leave this analysis for future work.

Since an enterprise data warehouses can contain terabytes of data [10] ensuring that the dimensions satisfy the required integrity constraints can be vital for an efficient computation of reports and summaries. Therefore, it is important to develop tools to allow the designer of a DW to check integrity constraints and to help in the process of repairing inconsistencies with respect to them. We believe that such tools would be particularly useful in the creation and cleaning stage of DWs and when the dimensions are updated and adapted due to changes in data sources or modifications to the business rules [6, 15, 16, 22].

We have explored a novel line of research which is the application of logic programming to support data cleaning tasks in data warehousing. An interested idea for future research is to compute consistent answers to queries over DWs, following the framework proposed in [1, 4], where repairs are used as auxiliary concepts to characterize consistent answers to queries. A preliminary analysis in this direction is presented in [5].

Acknowledgements: Loreto Bravo is funded by CONICYT grant PSD-57. Mónica Caniupán is funded by FONDECYT grant #11070186.

References

1. M. Arenas, L. Bertossi, and J. Chomicki. Consistent Query Answers in Inconsistent Databases. In *PODS'99*, pages 68–79, 1999.

2. M. Arenas, L. Bertossi, and J. Chomicki. Answer Sets for Consistent Query Answering in Inconsistent Databases. *TPLP*, 3(4-5):393–424, 2003.
3. P. Barceló and L. Bertossi. Logic Programs for Querying Inconsistent Databases. In *PADL 03*, Springer LNCS 2562, pages 208–222, 2003.
4. L. Bertossi. Consistent Query Answering in Databases. *ACM Sigmod Record*, 35(2):68–76, 2006.
5. L. Bertossi, L. Bravo, and M. Caniupan. Consistent Query Answering in Data Warehouses. In *AMW'09*, volume 450, 2009.
6. M. Body, M. Miquel, Y. Bédard, and A. Tchounikine. Handling Evolutions in Multidimensional Structures. In *ICDE'03*, pages 581–591, 2003.
7. L. Bravo and L. Bertossi. Semantically Correct Query Answers in the Presence of Null Values. In *IIDB 06*, Springer LNCS 4254, pages 336–357, 2006.
8. F. Buccafurri, N. Leone, and P. Rullo. Enhancing Disjunctive Datalog by Constraints. *TKDE*, 12(5):845–860, 2000.
9. L. Cabibbo and R. Torlone. Querying Multidimensional Databases. In *DBLP'98*, pages 319–335. Springer-Verlag, 1998.
10. S. Chaudhuri and U. Dayal. An Overview of Data Warehousing and OLAP Technology. *SIGMOD Record*, 26(1):65–74, 1997.
11. M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. In *ICLP'88*, pages 1070–1080, 1988.
12. F. Giannotti, S. Greco, D. Saccà, and C. Zaniolo. Programming with Nondeterminism in Deductive Databases. *AMAI*, 19(1-2):97–125, 1997.
13. C. Hurtado, C. Gutierrez, and A. Mendelzon. Capturing Summarizability with Integrity Constraints in OLAP. *TODS*, 30(3):854–886, 2005.
14. C. Hurtado and A. Mendelzon. Reasoning about Summarizability in Heterogeneous Multidimensional Schemas. In *ICDT'01*, pages 375–389, 2001.
15. C. Hurtado, A. Mendelzon, and A. Vaisman. Maintaining Data Cubes under Dimension Updates. In *ICDE'99*, pages 346–355, 1999.
16. C. Hurtado, A. Mendelzon, and A. Vaisman. Updating OLAP Dimensions. In *DOLAP'99*, pages 60–66, 1999.
17. H. V. Jagadish, L. Lakshmanan, and D. Srivastava. What can Hierarchies do for Data Warehouses? In *VLDB'99*, pages 530–541, 1999.
18. R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Inc., 2002.
19. H. Lenz and A. Shoshani. Summarizability in OLAP and Statistical Data Bases. In *SSDBM'97*, pages 132–143, 1997.
20. N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, C. Koch, C.Mateis, S. Perri, and F. Scarcello. The DLV System for Knowledge Representation and Reasoning. *TOCL*, 7(3):499–562, 2006.
21. C. Letz, E. Henn, and G. Vossen. Consistency in Data Warehouse Dimensions. In *IDEAS'02*, pages 224–232, 2002.
22. A. Mendelzon and A. Vaisman. Temporal Queries in OLAP. In *VLDB'00*, pages 242–253, 2000.
23. T. Pedersen, C. Jensen, and C. Dyreson. Extending Practical Pre-Aggregation in On-Line Analytical Processing. In *VLDB'99*, pages 663–674, 1999.
24. T. Pedersen, C. Jensen, and C. Dyreson. A Foundation for Capturing and Querying Complex Multidimensional Data. *Inf. Syst.*, 26(5):383–423, 2001.
25. M. Rafanelli and A.Shoshani. STORM: a Statistical Object Representation Model. In *SSDBM'90*, pages 14–29, 1990.
26. T. Syrjänen and I. Niemelä. *LPNMR'01*, chapter The Smodels System, pages 434–438. Springer LNCS 2173. 2001.