ROMA TRE
UNIVERSITÀ DEGLI STUDI

Dipartimento di Informatica e Automazione
Via della Vasca Navale, 79
00146 Roma, Italy

A Model and a Language for Context-Aware Databases

Davide Martinenghi[1], Riccardo Torlone[2]

RT-DIA-152-2009 **July 2009**

(1) Dip. di Elettronica e Informazione,
Politecnico di Milano, Italy
martinen@elet.polimi.it

(2) Dip. di Informatica e Automazione,
Università Roma Tre, Italy
torlone@dia.uniroma3.it

Davide Martinenghi acknowledges support from the "Search Computing" (SeCo) project, funded by the ERC under the 2008 Call for "IDEAS Advanced Grants".

ABSTRACT

Context awareness is an enabling technology of ubiquitous computing aimed at utilizing the location, the time, and other properties that characterize the context of use to select the information that is most appropriate to final users. Although it is widely considered a fundamental ability of modern applications, current database technology does not provide a support to context awareness yet.

In this paper, we propose a logical model and an abstract query language as a foundation for context-aware database management systems. The model is a natural extension of the relational model in which contexts are first class citizens and can be described at different levels of granularity. This guarantees a smooth implementation of the approach with current database technology. The query language is a conservative extension of relational algebra where special operators allow the specification of queries over contexts. As it happens in practice, contexts in queries and contexts associated with data can be at different granularities: this is made possible by a partial order relationship defined over contexts. We also study equivalence and rewriting properties of the query language that can be used for the optimization of context-aware queries.

1 Introduction

Nowadays, context awareness is widely recognized to be a fundamental means to guide users through the wealth of information available in digital form [8]. Basically, a context aware application provides some ability to filter, adapt, and personalize the accessible content according to a set of features characterizing the user context. These features may include the location, time, device as well as any available aspect that allows the identification of the most appropriate information for the final user.

In spite of a steady trend towards more user-centric, personalized, and adaptive applications, the support of current database technology to context awareness is very limited. The only examples are some ad-hoc functionality provided by specialized systems, such as temporal and spatial DBMS, that explicitly take into account some specific contextual property. Conversely, we believe that context-aware applications can benefit greatly from applying the relational database technology enhanced with a comprehensive support for the management of context. In particular, it will become possible to formulate as queries much more sophisticated search requests, such as those required by geographic and location-based applications [7].

In the light of these considerations, we propose in this paper a logical data model and an abstract query language for context-aware databases with the goal of providing a solid foundation to the problem of adding context awareness to database systems.

The data model is a natural extension of the relational model in which contexts are first class citizens: this guarantees a smooth implementation of the approach with current database technology. As it happens in practice, contexts can be expressed at different levels of detail: this is made possible by a partial order relationship defined over contexts both at the schema and at the instance level. Basically, a context c associated with a relational tuple t specifies that t is valid in c and in any other context c' that includes c according to the partial order defined on contexts: it follows that t can be selected in a query that refers to c'.

The query language we propose, called Contextual Relational Algebra (CRA), is a conservative extension of relational algebra that allows the specification of queries over contexts. CRA allows a simple, natural embedding of contexts into ordinary queries through some special operators for contexts that extend the capabilities of standard projection, selection and join operators. These operators make it possible to formulate complex context-aware queries, such as those required by location-based search (find an opera concert in or close to Rome next summer), multifaceted product search (find a cheap bluray player with an adequate user rating), social search (find the objects that the friends of Jane like), and device adaptation (resize the picture to fit my screen). We also present general algebraic rules governing the operators for contexts and their interactions with standard relational algebra operators. The rules provide a formal foundation for query equivalence and for the algebraic optimization of context-aware queries.

In sum, the contributions of this paper are the following:

- a simple but solid framework for embedding contexts into relational databases: the framework does not depend on a specific domain and makes the comparison of heterogeneous contexts possible and straightforward;
- a simple but powerful algebraic language for expressing context-aware queries: the query language makes it possible to formulate complex context-aware searches in

different application domains;

- the investigation of the relationships between the query language operators and the identification of a number of equivalence rules: the rules provide a formal foundation for the algebraic optimization of context-aware queries.

The rest of the paper is organized as follows. In Section 2 we introduce some preliminary notion and present the data model. CRA, the query language for this model, is illustrated in Section 3. In Section 4 we show how the operators of CRA interact, and provide a number of equivalence rules that can be used for query optimization. In Section 5 we compare our approach with related works and finally, in Section 6, we draw some conclusions and sketch future works.

2 A Data Model with Contexts

In this section, we present an extension of the relational model in which contexts are first class citizens. We start with some preliminary notions on partial orders, which are basic ingredients of our model.

2.1 Partial orders and lattices

A (weak) *partial order* $\leq$ on a domain V is a subset of $V \times V$ whose elements are denoted by $v_1 \leq v_2$ that is: reflexive ($v \leq v$ for all $v \in V$), antisymmetric (if $v_1 \leq v_2$ and $v_2 \leq v_1$ then $v_1 = v_2$), and transitive (if $v_1 \leq v_2$ and $v_2 \leq v_3$ then $v_1 \leq v_3$) [3]. If $v_1 \leq v_2$ we say that v_1 *is included in* v_2. A set of values V with a partial order $\leq$ is called a *poset*.

A lower bound (upper bound) of two elements v_1 and v_2 in a poset $(V, \leq)$ is an element $b \in V$ such that $b \leq v_1$ and $b \leq v_2$ ($v_1 \leq b$ and $v_2 \leq b$). A *maximal lower bound* (*minimal upper bound*) is a lower bound (upper bound) b of two elements v_1 and v_2 in a poset $(V, \leq)$ such that there is no lower bound (upper bound) b' of v_1 and v_2 such that $b' \leq b$ ($b \leq b'$).

The *greatest lower bound* or *glb* (*least upper bound* or *lub*) is a lower bound (upper bound) b of two elements v_1 and v_2 in a poset $(V, \leq)$ such that $b' \leq b$ ($b \leq b'$) for any other lower bound (upper bound) b' of v_1 and v_2. It easily follows that if a lub (glb) exists, then it is unique. The glb and the lub are also called *meet* and *join*, respectively.

A *lattice* is a poset in which any two elements have both a glb and a lub [2]. The glb and lub can also be defined over a set of elements. By induction, it follows that every non-empty finite subset of a lattice has a glb and a lub.

2.2 Contextual dimensions

We represent a context by means of a set of *dimensions*, each of which represents an autonomous aspect that should influence the delivery of data. Examples of dimensions are the time and the location in which the source is accessed, the access device and the network used, and so on. Each dimension is described by means of a set of *levels* representing the dimension at different degrees of granularity. For instance, the time dimension can be organized in levels like day, week, month and year.

Definition 1 (Contextual dimension) *A* (contextual) dimension d *is composed of:*

- *a finite set* $L = \{l_1, \ldots, l_k\}$ *of* levels*, each of which is associated with a set of values called the* members *of the level and denoted by* $M(l)$*;*
- *a partial order* $\leq_L$ *on* L *having a bottom element, denoted by* $\bot_L$*, and a top element, denoted by* $\top_L$*, such that:*
  - $M(\bot_L)$ *contains a set of* ground *members whereas all the other levels contain members that represent groups of ground members;*
  - $M(\top_L)$ *contains only a special member* $m_\top$ *that represents all the ground members;*
- *a family* CM *of* containment mappings $\text{CMAP}_{l_1}^{l_2} : M(l_1) \rightarrow M(l_2)$ *for each pair of levels* $l_1 \leq_L l_2$ *satisfying the following* consistency conditions*:*
  - *for each level* l*, the function* CMAP_l^l *is the identity on the members of* l*;*
  - *for each pair of levels* l_1 *and* l_2 *such that* $l_1 \leq_L l \leq_L l_2$ *and* $l_1 \leq_L l' \leq_L l_2$ *for some* $l \neq l'$*, we have:* $\text{CMAP}_l^{l_2}(\text{CMAP}_{l_1}^l(m)) = \text{CMAP}_{l'}^{l_2}(\text{CMAP}_{l_1}^{l'}(m))$ *for each member* m *of* l_1*.*

Example 1 The contextual dimension *time* has a bottom level whose (ground) members are timestamps and a top level whose only member, `anytime`, represents all possible timestamps. Other levels can be day, week, month, quarter, season and year, where day $\leq_L$ month $\leq_L$ quarter $\leq_L$ year and day $\leq_L$ season. A possible member of the Day level is 23/07/2009, which is mapped by the containment mappings to the member 07/2009 of the level month and to the member `Summer` of the level season. ■

The following result can be easily shown.

Proposition 1 *The poset* $(L, \leq_L)$ *is a lattice and therefore every pair of levels* l_1 *and* l_2 *in* L *has both a glb and a lub.*

Actually, a partial order $\leq_M$ can also be defined on the members M of a dimension D: it is induced by the containment mappings as follows.

Definition 2 (Poset on members) *Let* D *be a dimension and* m_1 *and* m_2 *be members of levels* l_1 *and* l_2 *of* D*, respectively. We have that* $m_1 \leq_M m_2$ *if: (i)* $l_1 \leq_L l_2$ *and (ii)* $\text{CMAP}_{l_1}^{l_2}(m_1) = m_2$*.*

Example 2 Consider the dimension of Example 1. Given the members m_1 = 29/06/2009 and m_2 = 23/08/2009 of the level day, m_3 = 06/2009 and m_4 = 08/2009 of the level month, m_5 = 2Q 2009 and m_6 = 3Q 2009 of the level quarter, m_7 = 2009 of the level year, and m_8 = `Summer` of the level season, we have: $m_1 \leq_M m_3 \leq_M m_5 \leq_M m_7$, $m_2 \leq_M m_4 \leq_M m_6 \leq_M m_7$, and $m_1 \leq_M m_8$ and $m_2 \leq_M m_8$. ■

Example 2 shows an interesting property: differently from the poset on the dimension levels, the poset on the members of a dimension is not a lattice in general. Consider for instance the members m_1 and m_2 of this example: they have no lower bounds, since their intersection is empty (more precisely, the intersection of the ground members that they represent is empty), and have two incomparable minimal upper bounds: 2009 at the

year level and `Summer` at the season level. Indeed, it is possible to show that the poset $(M, \leq_M)$ can be converted into a lattice by adding to M all the elements of the *powerset* of the ground members (including the empty set, which would become the bottom level). This however would imply an explosion of the number of members and an unnatural representation of a contextual dimension.

2.3 Contexts

We are ready to introduce our notion of context.

Definition 3 (Context) *Let D be a set of contextual dimensions. We denote by $C = (A_1 : l_1, \ldots, A_k : l_k)$ a* context schema *where each A_i is a distinct* attribute *name and each l_i is a level of some dimension in D. A* context *c over C is a function that maps each attribute A_i to a member of l_i.*

Given a context c over a schema C and an attribute A_i occurring in C on level l_i, we will denote by $c[A_i : l_i]$ the member of level l_i associated with c on A_i. Also, for a subset C' of C, we will denote by $c[C']$ the restriction of c to C'. Moreover, for the sake of simplicity, often in the following we will not make any distinction between the name of an attribute of a context and the name of the corresponding level when no ambiguities can arise.

Example 3 As an example, a context schema over the dimensions *time*, *location* and *weather conditions* can be the following: (Time: day, Location: city, Weather: brief). A possible context over this schema is: $c =$ `(12/05/2009; Rome; Sunny)`. Then we have: c[Location:city] = `Rome`. ■

A partial order relation on both context schemas and contexts can be also defined in a natural way.

Definition 4 (Poset on context schemas) *Let C_1 and C_2 be context schemas over D_1 and D_2 respectively. We have that $C_1 \leq_C C_2$ if: (i) $D_2 \subseteq D_1$, and (ii) for each $A_i : l_i \in C_2$ there is an element $A_i : l_j \in C_1$ such that $l_j \leq_L l_i$.*

Definition 5 (Poset on contexts) *Let c_1 and c_2 be contexts over C_1 and C_2 respectively. We have that $c_1 \leq_c c_2$ if: (i) $C_1 \leq_C C_2$, and (ii) for each $A_i : l_i \in C_2$ there is an element $A_i : l_j \in C_1$ such that $c_1[A_i : l_j] \leq_M c_2[A_i : l_i]$.*

Note that, in these definitions, we assume that levels of the same dimension occur in different contexts with the same attribute name: this strongly simplifies the notation that follows without loss of expressibility. Basically, it suffices to use as attribute name the role played by the dimension in the application domain.

Example 4 Consider the context $c_1 =$ `(10/05/2009; 12/05/2009; Rome; Sunny)` over the context schema C_1 = (StartTime: day; EndTime: day; Location: city; Weather: brief); the context $c_2 =$ `(11/08/2009; Rimini)` over the context schema C_2 = (StartTime: day; Location: city); the context $c_3 =$ `(2Q 2009; Lazio)` over the context schema C_3 = (StartTime: quarter; Location: region); and finally the context $c_4 =$ `(Summer; Italy)` over the context schema C_4 = (StartTime: season; Location: country). Then, it is easy to see that: (i) $C_1 \leq_C C_2 \leq_C C_3$, and $C_2 \leq_C C_4$, and (ii) $c_1 \leq_c c_3$ and $c_2 \leq_c c_4$. ■

The same considerations done for the poset on dimension levels also apply for the poset on context schemata.

Proposition 2 *Let $\mathcal{C}$ be the set of all possible context schemas over a set of dimensions D. Then, the poset $(\mathcal{C}, \leq_C)$ is a lattice.*

Conversely, the poset on contexts is not a lattice in general since, it is easy to show that, given two contexts, they can have more than one minimal upper bound (but necessarily at least one) as well as more than one maximal lower bound (possibly none).

2.4 C-relations

As usual, we denote by $X = (A_1 : V_1, \ldots, A_k : V_k)$ a *relation schema*, where each A_i is a distinct *attribute* and each V_i is a set of values called the *domain* of A_i. A *tuple* t over a relation schema X is a function that associates with each A_i occurring in X a value taken from V_i. A *relation* r over a relation schema X is a finite set of tuples over X.

A contextualized relation, or simply a *c-relation*, is a database relation whose tuples include contexts. Basically, a context c associated with a tuple t means that t represents a valid piece of information in c.

Definition 6 (C-relation) *Let D be a set of contextual dimensions. We denote by $R(X^s \| X^c)$ a* c-relation schema*, where R is the* name *of the schema, X^s is a relation schema and X^c is a context schema. A* c-relation *over $R(X^s \| X^c)$ is a set of tuples $t = (s \| c)$ where s is a tuple over X^s and c is a context over X^c. The members of X^c are called* contextual attributes.

We recall that, for the sake of simplicity, we will often make no distinction between the name of a contextual attribute and the corresponding level.

We now give some examples of c-relations that will be used for query answering purposes in the next section.

Example 5 An example of c-relation over the schema:

$$R_1(\text{Opera} : \text{string}, \text{Director} : \text{string} \| \text{Time: day}, \text{Location: theater})$$

is the following.

$r_1 =$

Opera	Director	Time:day	Location:theater	
La Traviata	Abbado	11/05/2009	La Scala	$t_{1,1}$
La Bohème	Chailly	24/04/2008	Opéra	$t_{1,2}$
Turandot	Maazel	24/07/2009	Arena	$t_{1,3}$
Rigoletto	Muti	24/04/2008	La Scala	$t_{1,4}$

An example of c-relation over $R_2(\text{Price} : \text{real} \| \text{Location: theater}, \text{Time: quarter})$ is:

$r_2 =$

Price	Location:theater	Time:quarter	
150	La Scala	1Q 2009	$t_{2,1}$

The following c-relation is over $R_3(\text{Company} : \text{string} \| \text{Location: airport})$.

$r_3 =$

Company	Location:airport	
Alitalia	Villafranca	$t_{3,1}$
Air France	Roissy	$t_{3,2}$

Finally, a c-relation over $R_4(\text{Discount} : \text{percentage} \| \text{Time: month})$ is:

$r_4 =$

Discount	Time:month	
10%	03/2009	$t_{4,1}$
20%	06/2009	$t_{4,2}$

Note that a tabular representation has been chosen here for a c-relation but other solutions are possible. For instance, we could represent the non-contextual part of the tuples in an array in which the context provides the coordinates. The following is a representation of r_1.

	La Scala	Opéra	Arena
11/05/2009	La Traviata, Abbado	-	-
24/04/2008	Rigoletto, Muti	La Bohème, Chailly	-
24/07/2009	-	-	Turandot, Maazel

■

As shown in the previous example, our model is a logical model that can therefore be implemented in several ways.

3 Querying contextual data

In this section we present CRA (Contextual Relational Algebra) an extension of the relational algebra over c-relations. This language provides insights on the way in which contextual data can be manipulated and, for its procedural nature, can be profitably used to specify query optimization. The goal is provide a solid foundation to querying over contexts.

Similarly to what happens with the standard relational algebra, the operators of CRA are closed, that is, they apply to c-relations and produce a c-relation as result. In this way, the various operators can be composed to form the *c-expressions* of the language.

CRA is a conservative extension of basic relational algebra (RA) and so it includes its standard operators: selection (σ), projection (π), and natural join ($\bowtie$). It also includes some variants of these operators that are obtained by combining them with the following two new operators.

Definition 7 (Upward extension) *Let r be a c-relation over $R(X^s \| X^c)$, A be a contextual attribute in X^c defined over a level l, and l' be a level such that $l \leq_c l'$. The* upward extension *of r on l', denoted by $\hat{\varepsilon}^{A:l'}_{A:l}(r)$, is the c-relation over $R'(X^s \| X^c \cup \{A : l'\})$ defined as follows:*

$$\hat{\varepsilon}^{A:l'}_{A:l}(r) = \{(s \| c) \mid \exists (\bar{s} \| \bar{c}) \in r : s = \bar{s}, c[X^c] = \bar{c}, c[A : l'] = \textsc{cmap}^{l'}_{l}(\bar{c}[A : l])\}$$

Definition 8 (Downward extension) *Let r be a c-relation over $R(X^s \| X^c)$, A be a contextual attribute in X^c defined over a level l, and l' be a level such that $l' \leq_c l$. The* downward extension *of r on l', denoted by $\check{\varepsilon}^{A:l}_{A:l'}(r)$, is the c-relation over $R'(X^s \| X^c \cup \{A : l'\})$ defined as follows:*

$$\check{\varepsilon}^{A:l}_{A:l'}(r) = \{(s \| c) \mid \exists (\bar{s} \| \bar{c}) \in r : s = \bar{s}, c[X^c] = \bar{c}, \bar{c}[A : l] = \textsc{cmap}^{l}_{l'}(c[A : l'])\}$$

For simplicity, in the following we will often simply write $\hat{\varepsilon}_l^{l'}$ or $\check{\varepsilon}_l^{l'}$, when there is no ambiguity on the attribute name associated with the corresponding levels.

Example 6 Consider the c-relations r_1 and r_2 from Example 5. The result of $\hat{\varepsilon}^{\mathsf{City}}_{\mathsf{Theater}}(r_1)$ is the following c-relation.

$r_5 =$

Opera	Director	Time:day	Location:theater	Location:city	
La Traviata	Abbado	11/05/2009	La Scala	Milan	$t_{5,1}$
La Bohème	Chailly	24/04/2008	Opéra	Paris	$t_{5,2}$
Turandot	Maazel	24/07/2009	Arena	Verona	$t_{5,3}$
Rigoletto	Muti	24/04/2008	La Scala	Milan	$t_{5,4}$

The result of $\check{\varepsilon}^{\mathsf{Quarter}}_{\mathsf{Month}}(r_2)$ is the following c-relation.

$r_6 =$

Price	Location:theater	Time:quarter	Time:month	
150	La Scala	1Q 2009	01/2009	$t_{6,1}$
150	La Scala	1Q 2009	02/2009	$t_{6,2}$
150	La Scala	1Q 2009	03/2009	$t_{6,3}$

■

The main rationale behind the introduction of the upward extension is the need to relax a query with respect the level of detail of the queried information. For example, one might want to find events taking place in a given country, even though the events might be stored with a finer granularity (e.g., city). Similarly, the downward extension allows the relaxation of the answer with respect to the level of detail of the query. For instance, a query about products available in a given day may return the products available in that day's month. Both kinds of extensions meet needs that arise naturally in several application domains.

To this end, we introduce two new operators for the selection over the contextual part of a c-relation; they can reference a context that is more general or more specific than that occurring in its tuples.

Definition 9 (Upward selection) *Let r be a c-relation over $R(X^s \| X^c)$, A be a contextual attribute in X^c defined over l, m be a member of l' with $l \leq_c l'$, and $\theta \in \{=, <, >, \leq, \geq, \neq\}$: the* upward selection *of r with respect to $A\,\theta\,m$ on level l, denoted by $\hat{\sigma}_{A:l\,\theta\,m}(r)$, is the c-relation over $R(X^s \| X^c)$ defined as follows:*

$$\hat{\sigma}_{A:l\,\theta\,m}(r) = \{(s\|c) \in r \mid \textsc{cmap}_l^{l'}(c[A:l])\,\theta\,m\}$$

Definition 10 (Downward selection) *Let r be a c-relation over $R(X^s \| X^c)$, A be a contextual attribute in X^c defined over l, m be a member of l' with $l' \leq_c l$, and $\theta \in \{=, <, >, \leq, \geq, \neq\}$: the* downward selection *of r with respect to $A\,\theta\,m$ on level l, denoted by $\check{\sigma}_{A:l\,\theta\,m}(r)$, is the c-relation over $R(X^s \| X^c)$ defined as follows:*

$$\check{\sigma}_{A:l\,\theta\,m}(r) = \{(s\|c) \in r \mid \textsc{cmap}_{l'}^{l}(m)\,\theta\,c[A:l]\}$$

It can be easily seen that these operators can be obtained by composing the upward or downward extension, the (standard) selection, and the projection operators, as shown in (1) and (2) below.

$$\hat{\sigma}_{A:l\,\theta\,m}(r) = \pi_{X^s \cup X^c}(\sigma_{A:l'\,\theta\,m}(\hat{\varepsilon}^{A:l'}_{A:l}(r))) \quad (1)$$
$$\check{\sigma}_{A:l\,\theta\,m}(r) = \pi_{X^s \cup X^c}(\sigma_{A:l'\,\theta\,m}(\check{\varepsilon}^{A:l}_{A:l'}(r))) \quad (2)$$

In the following, we will often simply write $\hat{\sigma}_{A\,\theta\,m}$ and $\check{\sigma}_{A\,\theta\,m}$, without explicitly indicating the name of the level, when this is unambiguously determined by the corresponding attribute.

Example 7 Consider again the c-relations r_1 and r_2 from Example 5. We have that: $\hat{\sigma}_{\mathsf{City}=\mathtt{Milan}}(r_1) = \{t_{1,1}, t_{1,4}\}$ and $\check{\sigma}_{\mathsf{Day}=\mathtt{13/03/2009}}(r_2) = \{t_{2,1}\}$. ■

Finally, we introduce two new join operators. Their main purpose is to combine information stored at different levels of granularity.

Definition 11 (Upward join) *Let r_1 and r_2 be two c-relations over $R_1(X_1^s \| X_1^c)$ and $R_2(X_2^s \| X_2^c)$ respectively, and let X^c be an upper bound of a subset $\bar{X}_1^c$ of X_1^c and a subset $\bar{X}_2^c$ of X_2^c. The* upward join *of r_1 and r_2 with respect to X^c on $\bar{X}_1^c$ and $\bar{X}_2^c$, denoted by $r_1 \hat{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2$, is the c-relation over $R_{12}(X_1^s \cup X_2^s \| X_1^c \cup X_2^c)$ defined as follows:*

$$r_1 \hat{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2 = \{ \begin{array}{l} (s\|c) \mid \exists (s_1\|c_1) \in r_1, \exists (s_2\|c_2) \in r_2, \exists c' \, over \, X^c : c_1[\bar{X}_1^c] \leq_c c', \\ c_2[\bar{X}_2^c] \leq_c c', s[X_1^s] = s_1, s[X_2^s] = s_2, c[X_1^c] = c_1, c[X_2^c] = c_2\} \end{array}$$

Definition 12 (Downward join) *Let r_1 and r_2 be two c-relations over $R_1(X_1^s \| X_1^c)$ and $R_2(X_2^s \| X_2^c)$ respectively, and let X^c be a lower bound of a subset $\bar{X}_1^c$ of X_1^c and a subset $\bar{X}_2^c$ of X_2^c. The* downward join *of r_1 and r_2 with respect to X^c on $\bar{X}_1^c$ and $\bar{X}_2^c$, denoted by $r_1 \check{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2$, is the c-relation over $R_{12}(X_1^s \cup X_2^s \| X_1^c \cup X_2^c)$ defined as follows:*

$$r_1 \check{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2 = \{ \begin{array}{l} (s\|c) \mid \exists (s_1\|c_1) \in r_1, \exists (s_2\|c_2) \in r_2, \exists c' \, over \, X^c : c' \leq_c c_1[\bar{X}_1^c], \\ c' \leq_c c_2[\bar{X}_2^c], s[X_1^s] = s_1, s[X_2^s] = s_2, c[X_1^c] = c_1, c[X_2^c] = c_2, \} \end{array}$$

In the following, we will omit the indication of $\bar{X}_1^c$ and $\bar{X}_2^c$ when evident from the context.

Example 8 Consider the c-relations r_1 and r_3 from Example 5. The result of $r_1 \hat{\bowtie}_{\mathsf{City}} r_3$ is the following c-relation:

$r_7 =$

Opera	Director	Company	Time:day	Location:theater	Location:airport	
La Bohème	Chailly	Air France	24/04/2008	Opéra	Roissy	$t_{7,1}$
Turandot	Maazel	Alitalia	24/07/2009	Arena	Villafranca	$t_{7,2}$

Consider now the c-relations r_2 and r_4 from Example 5. The result of $r_2 \check{\bowtie}_{\mathsf{Theater,Day}} r_4$ is the following c-relation:

$r_8 =$

Price	Discount	Location:theater	Time:quarter	Time:month	
150	10%	La Scala	1Q 2009	03/2009	$t_{8,1}$

■

Also in this case, both the upward join and the downward join can be obtained by combining the upward extension or the downward extension, and the (standard) join. Equation (3) below shows this for the upward join, where $X^c = \{A^1 : l^1, \ldots, A^n : l^n\}$, $\bar{X}_i^c \supseteq \{A^1 : l_i^1, \ldots, A^n : l_i^n\}$ for $i = 1, 2$, and P is a predicate requiring pairwise equality in both sides of the join for all fields added by the extensions.

$$r_1 \hat{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2 = \hat{\varepsilon}^{A^1:l^1}_{A^1:l_1^1} \cdots \hat{\varepsilon}^{A^n:l^n}_{A^n:l_1^n}(r_1) \bowtie_P \hat{\varepsilon}^{A^1:l^1}_{A^1:l_2^1} \cdots \hat{\varepsilon}^{A^n:l^n}_{A^n:l_2^n}(r_2) \tag{3}$$

Equation (4) below shows this for the downward join, where $X^c \supseteq \{A^1 : l^1, \ldots, A^n : l^n\}$, $\bar{X}_i^c \supseteq \{A^1 : l_i^1, \ldots, A^n : l_i^n\}$ for $i = 1, 2$, and P is as above.

$$r_1 \check{\bowtie}_{X^c:\bar{X}_1^c,\bar{X}_2^c} r_2 = \check{\varepsilon}^{A^1:l_1^1}_{A^1:l^1} \cdots \check{\varepsilon}^{A^n:l_1^n}_{A^n:l^n}(r_1) \bowtie_P \check{\varepsilon}^{A^1:l_2^1}_{A^1:l^1} \cdots \check{\varepsilon}^{A^n:l_2^n}_{A^n:l^n}(r_2) \tag{4}$$

As in the standard relational algebra, it is possible to build complex expressions combining several CRA operators thanks to the fact that CRA is closed, i.e., the result of every application of an operator is a c-relation. Formally, one can define and build the expressions of CRA, called c-expressions, by assuming that c-relations themselves are c-expressions, and by substituting the c-relations appearing in Definitions 7-12 with a c-expression.

4 Equivalences in Contextual Relational Algebra

One of the main applications of Relational Algebra is the use of algebraic properties for query optimization. In particular, equivalences allow transforming a relational expression into an equivalent expression in which the average size of the relations yielded by subexpressions is smaller. Typical rewritings exploit properties such as commutativity and idempotency, and may be used, e.g., to break up an application of an operator into several, smaller applications, or to move operators to more convenient places in the expression (e.g., pushing selection and projection through join).

In analogy with the standard case, we are now going to describe a collection of new equivalences that can be used for query optimization in Contextual Relational Algebra.

In the remainder of this section, we shall use, together with possible subscripts and primes, the letter r to denote a contextual relation, l for a level, A for a set of attributes, and P for a (selection or join) predicate.

4.1 Upward and downward extension

Border cases

$$\hat{\varepsilon}^l_l(r) = \check{\varepsilon}^l_l(r) = r \tag{5}$$

Equivalence (5) shows that if the upper and lower level of an extension coincide, then the extension is idle, both for the upward and for the downward case. The proof of (5) follows immediately from Definitions 7 and 8, as long as l is assumed to be the level of an attribute in r.

Idempotency

$$\hat{\varepsilon}_l^{l'}(\hat{\varepsilon}_l^{l'}(r)) = \hat{\varepsilon}_l^{l'}(r) \tag{6}$$
$$\check{\varepsilon}_{l'}^{l}(\check{\varepsilon}_{l'}^{l}(r)) = \check{\varepsilon}_{l'}^{l}(r) \tag{7}$$

Equivalences (6) and (7) state that repeated applications of the same extension are idle, both for the upward and for the downward case. In both, it is assumed that l is the level of an attribute of r and that $l \leq_L l'$. Here, too, the proof follows immediately from Definitions 7 and 8.

Duality

$$\hat{\varepsilon}_{l'}^{l}(\check{\varepsilon}_{l'}^{l}(r)) = \check{\varepsilon}_{l'}^{l}(r) \tag{8}$$

The above equivalence (8) shows that an upward extension is always idle after a downward extension on the same levels. To prove (8), it suffices to assume that there already is a contextual attribute of level l in the schema of r, and to consider that the mapping from members of a lower level to members of an upper level is many-to-one, so no new tuple can be generated by the upward extension. Note, however, that the downward extension after an upward extension on the same levels is generally not redundant, since the mapping from members of an upper level to members of a lower level is one-to-many.

Commutativity

$$\hat{\varepsilon}_{l_2}^{l_2'}(\hat{\varepsilon}_{l_1}^{l_1'}(r)) = \hat{\varepsilon}_{l_1}^{l_1'}(\hat{\varepsilon}_{l_2}^{l_2'}(r)) \tag{9}$$
$$\check{\varepsilon}_{l_2'}^{l_2}(\check{\varepsilon}_{l_1'}^{l_1}(r)) = \check{\varepsilon}_{l_1'}^{l_1}(\check{\varepsilon}_{l_2'}^{l_2}(r)) \tag{10}$$

The above equivalences (9) and (10) state that two contextual extensions of the same kind can be swapped as long as both starting levels (here, l_1 and l_2) are levels of contextual attributes of r. Both follows straightforwardly from Definitions 7 and 8.

Interplay with standard projection

$$\pi_{A_p}\hat{\varepsilon}_{A:l}^{A:l''}(r) = \pi_{A_p}\hat{\varepsilon}_{A:l'}^{A:l''}(\hat{\varepsilon}_{A:l}^{A:l'}(r)) \tag{11}$$
$$\pi_{A_p}\check{\varepsilon}_{A:l''}^{A:l}(r) = \pi_{A_p}\check{\varepsilon}_{A:l''}^{A:l'}(\check{\varepsilon}_{A:l'}^{A:l}(r)) \tag{12}$$

Assuming that l is the level of an attribute A in a relation r over $R(X^s \| X^c)$, and that $l \leq_L l' \leq_L l''$, the above Equivalence (11) holds on the condition that A_p be a subset of $X^s \cup X^c$ not including $A : l'$. Note that the outer π_{A_p} is necessary, because, in case $l \neq l' \neq l''$, the left-hand sides of the equivalences would be c-relations that do not include the attribute-level pair $A : l'$, whereas the right-hand sides would; therefore, projecting away $A : l'$ is essential.

For similar reasons, it is also possible to swap contextual extension and standard projection provided that the projection does not retain the attribute that has been added by the extension:

$$\pi_{A_p}(\hat{\varepsilon}_{A:l}^{A:l'}(r)) = \hat{\varepsilon}_{A:l}^{A:l'}(\pi_{A_p}(r)) \tag{13}$$
$$\pi_{A_p}(\check{\varepsilon}_{A:l'}^{A:l}(r)) = \check{\varepsilon}_{A:l'}^{A:l}(\pi_{A_p}(r)) \tag{14}$$

Equivalences (13) and (14) show this both for the upward and the downward case, where l is the level of an attribute A in a relation r over $R(X^s \| X^c)$, $l \leq_L l'$, and A_p is a subset of $X^s \cup X^c$ not including $A : l'$.

Interplay with standard selection

$$\sigma_P(\hat{\varepsilon}^{A:l'}_{A:l}(r)) = \hat{\varepsilon}^{A:l'}_{A:l}(\sigma_P(r)) \tag{15}$$
$$\sigma_P(\check{\varepsilon}^{A:l}_{A:l'}(r)) = \check{\varepsilon}^{A:l}_{A:l'}(\sigma_P(r)) \tag{16}$$

Equivalences (15) and (16) show that swapping is also possible between contextual extension and standard selection, provided that the attribute-level pair that has been added by the extension is immaterial to the selection predicate. It is assumed that l is the level of an attribute A in r, $l \leq_L l'$, and P is a selection predicate that does not refer to $A : l'$.

Interplay with standard join

$$\hat{\varepsilon}^{A:l'}_{A:l}(r_1 \bowtie_P r_2) = (\hat{\varepsilon}^{A:l'}_{A:l}(r_1)) \bowtie_P r_2 \tag{17}$$
$$\check{\varepsilon}^{A:l}_{A:l'}(r_1 \bowtie_P r_2) = (\check{\varepsilon}^{A:l}_{A:l'}(r_1)) \bowtie_P r_2 \tag{18}$$

Equivalences (17) and (18) show that contextual extension can be "pushed" through standard join. It is assumed that $A : l$ is in the context schema of r_1, $l \leq_L l'$, and P is a join predicate not referring to $A : l'$. (Note that, if $A : l$ was in the context schema of both r_1 and of r_2, the extension should be "pushed" through both sides of the join.)

4.2 Upward and downward selection

Idempotency

$$\hat{\sigma}_{A:l\,\theta\,m}(\hat{\sigma}_{A:l\,\theta\,m}(r)) = \hat{\sigma}_{A:l\,\theta\,m}(r) \tag{19}$$
$$\check{\sigma}_{A:l\,\theta\,m}(\check{\sigma}_{A:l\,\theta\,m}(r)) = \check{\sigma}_{A:l\,\theta\,m}(r) \tag{20}$$

Equivalences (19) and (20) state that repeated applications of the same contextual selection are idle, both for the upward and for the downward case. In both, it is assumed that r has an attribute A of a level l such that $l \leq_L l'$ for (19) and $l' \leq_L l$ for (20), where l' is the level of m. To prove (19), consider that, by (1), the left-hand side of the equivalence can be written as:

$$\pi_{X^s \cup X^c}(\sigma_{A:l'\,\theta\,m}(\hat{\varepsilon}^{A:l'}_{A:l}(\pi_{X^s \cup X^c}(\sigma_{A:l'\,\theta\,m}(\hat{\varepsilon}^{A:l'}_{A:l}(r))))))$$

where X^s is the set of standard attributes of r and X^c is the context schema of r. The innermost projection can be moved outside the upward selection by using equivalence (13) if $l \neq l'$ or equivalence (5) if $l = l'$. By using standard properties of the relational operators, the innermost projection can also be moved outside the outermost selection, and eliminated by idempotency:

$$\pi_{X^s \cup X^c}(\sigma_{A:l'\,\theta\,m}(\hat{\varepsilon}^{A:l'}_{A:l}(\sigma_{A:l'\,\theta\,m}(\hat{\varepsilon}^{A:l'}_{A:l}(r)))))$$

Now, equivalence (15) allows swapping selection and upward extension provided that the selection predicate does not refer to the attribute-level pair introduced by the upward

extension. This condition is only required to make sure that, after the swap, the selection refers to an existing attribute-level pair. Therefore, equivalence (15) can be used here to move the innermost selection outside the outermost upward extension, although $A : l' \, \theta \, m$ is a predicate that clearly refers to $A : l'$ (l' being the level of m), since $A : l'$ is already introduced by the innermost extension. By idempotency of both standard selection and upward extension (as of equivalence (6)), we obtain

$$\pi_{X^s \cup X^c}(\sigma_{A:l' \, \theta \, m}(\hat{\varepsilon}^{A:l'}_{A:l}(r)))$$

which, by (1), corresponds to the right-hand side of (19).

An analogous argument can be used to prove (20).

Commutativity

$$\hat{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(\hat{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(r)) = \hat{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(\hat{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(r)) \tag{21}$$

$$\check{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(\check{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(r)) = \check{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(\check{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(r)) \tag{22}$$

$$\check{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(\hat{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(r)) = \hat{\sigma}_{l_1:A_1 \, \theta_1 \, m_1}(\check{\sigma}_{l_2:A_2 \, \theta_2 \, m_2}(r)) \tag{23}$$

The above equivalences state that contextual selection is commutative, both for the upward and the downward case. Moreover, an upward selection can be swapped with a downward selection (and vice versa), as shown in equivalence (23). The proof of these follows straightforwardly from commutativity of standard selection and interplay of contextual extension and standard selection.

4.3 Upward and downward join

Pushing upward and downward selection through upward and downward join

$$\hat{\sigma}_{A:l \, \theta \, m}(r_1 \hat{\bowtie}_{C_b:C_1,C_2} r_2) = (\hat{\sigma}_{A:l \, \theta \, m} r_1) \hat{\bowtie}_{C_b:C_1,C_2} r_2 \tag{24}$$

$$\hat{\sigma}_{A:l \, \theta \, m}(r_1 \check{\bowtie}_{C_b:C_1,C_2} r_2) = (\hat{\sigma}_{A:l \, \theta \, m} r_1) \check{\bowtie}_{C_b:C_1,C_2} r_2 \tag{25}$$

$$\check{\sigma}_{A:l \, \theta \, m}(r_1 \hat{\bowtie}_{C_b:C_1,C_2} r_2) = (\check{\sigma}_{A:l \, \theta \, m} r_1) \hat{\bowtie}_{C_b:C_1,C_2} r_2 \tag{26}$$

$$\check{\sigma}_{A:l \, \theta \, m}(r_1 \check{\bowtie}_{C_b:C_1,C_2} r_2) = (\check{\sigma}_{A:l \, \theta \, m} r_1) \check{\bowtie}_{C_b:C_1,C_2} r_2 \tag{27}$$

The above equivalences (24)-(27) indicate that a contextual selection can be "pushed" through a contextual join on the side that involves the attribute-level pair used in the selection. They assume that $A : l$ is in the context schema of r_1 but not of r_2, and that C_b is a (lower or upper) bound of a subset C_1 of the context schema of r_1 and a subset C_2 of the context schema of r_2. To prove the equivalences, it suffices to use (1)-(4) and the properties of standard operators.

Pushing standard projection through upward and downward join

$$\pi_L(r_1 \hat{\bowtie}_{C_b:C_1,C_2} r_2) = \pi_L((\pi_{L_1} r_1) \hat{\bowtie}_{C_b:C_1,C_2} (\pi_{L_2} r_2)) \tag{28}$$

$$\pi_L(r_1 \check{\bowtie}_{C_b:C_1,C_2} r_2) = \pi_L((\pi_{L_1} r_1) \check{\bowtie}_{C_b:C_1,C_2} (\pi_{L_2} r_2)) \tag{29}$$

In the above equivalences (28) and (29), it is shown how standard projection can be "pushed" through an upward or downward join to both sides of the join by properly

breaking up the projection attributes into smaller sets. It is assumed that r_i is a c-relation over $R_i(X_i^s \| X_i^c)$ for $i = 1, 2$, C_b is a (lower or upper) bound of a subset C_1 of X_1^c and a subset C_2 of X_1^c, L is a subset of the attributes in $X_1^s \cup X_1^c \cup X_2^s \cup X_2^c$, and $L_i = L \setminus (X_i^s \cup X_i^c) \cup C_i$ for $i = 1, 2$. Again, the equivalences follow immediately by applying (3) and (4) together with the standard "push" of projection through join and through contextual extension (as of equivalences (13) and (14)).

Based on the above discussion, we conclude this section with the claim of correctness of the presented equivalences.

Theorem 1 *Equivalences (5)-(29) are sound, i.e., they hold for any possible c-relation.*

Theorem 1 together with the fact that CRA is closed entails that equivalences (5)-(29) can also be used to test equivalence of complex c-expressions.

5 Related work

Context awareness has been studied in very diverse application domains including, among others, interface adaptation, information retrieval, service discovery, and artificial intelligence (see [1] for a survey on context-aware systems).

In this paper, we have focused on the problem of selecting the most appropriate data according to a context of reference and in particular on context-aware query answering [5]. Actually, several context data models have been proposed and a comprehensive survey of the most interesting data-oriented approaches has been presented in [4]. We have not considered the rich features provided by these models but rather we have concentrated on the basic ingredients that need to be added to the relational model with the aim of adding context-aware capabilities to current database technology. In this respect, the most relevant characteristic is the ability of the model to represent contexts at different levels of detail [4]. The model we have proposed is indeed a variation of a multidimensional model, which provides this ability for data warehousing applications [6]. Recently, some proposals of multidimensional context models similar to our approach have been presented [13, 14], but, unlike ours, they cannot be considered strict extensions of the relational model. A multidimensional data model for dealing with contexts has also been proposed in [16, 17], but the goal of the authors is rather different and focuses on contextual preferences, a problem that we have not addressed in this paper.

To our knowledge, CRA is the first proposal of a query language for context-aware relational databases that extends the classical relational algebra. The algebra presented in [13] is in fact rather informal and does not interact with standard operators. A query language with a similar objective has been proposed in a different framework [15]. However, the systematic analysis of the interaction between the various operators for optimization purposes has never been studied before.

The problem addressed in this paper is related to location-based search [7] and taxonomic search [11], which can be considered special cases of context-aware search. Since CRA queries can refer to contexts different from those associated with data, our approach is also related to the problem of query relaxation [12] and malleable schemas [18], in which the focus is on relaxing query constraints in order to deal with vague and heterogeneous schemas. Finally, a systematic study of querying into a taxonomy based on hierarchical properties of data, as it happens in our framework, has been addressed in [9, 10].

6 Conclusion

In this paper, we have presented a logical model and an algebraic language as a foundation for querying context-aware databases. In order to facilitate the implementation of the approach with current technology, they rely on a natural extension of the relational model. The hierarchical organization of contexts allows the specification of queries that refer to contexts at different levels of details, possibly different from those associated with data. We have also studied the interaction between the various operators of the query language as a formal foundation for the optimization of context-aware queries.

We believe that several interesting directions of research can be pursued within the framework presented in this paper. We are particularly interested into a deep investigation of general properties of the query language. In particular, we plan to develop methods for the automatic identification of the level in which two heterogeneous c-relations can be joined and to extend the approach to address user preferences [17] and the more general problem of query relaxation [18].

On the practical side, we are currently implementing a prototype for the management of context-aware databases based on the presented approach. With this prototype, we plan to develop quantitative analysis oriented to the optimization of context-aware queries. The equivalence results presented in this paper provide an important contribution in this direction.

References

[1] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *Int. Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263-277, 2007.

[2] G. Birkhoff. *Lattice Theory. Colloquium Publications, Volume XXV*, American Mathematical Society, third edition, 1967.

[3] G. Birkhoff and S. MacLane. Algebra. Third edition. *AMS Chelsea Publishing*, 1999.

[4] C. Bolchini, C. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca. A data-oriented survey of context models. *SIGMOD Record*, 36(4): 19–26, 2007.

[5] C. Bolchini, C. Curino, G. Orsi, E. Quintarelli, R. Rossato, F. A. Schreiber, and L. Tanca. And what can context do for data? *Commun. of ACM*, 2009.

[6] L. Cabibbo and R. Torlone. A logical approach to multidimensional databases. In *Proc. of EDBT*, pages 183–197, 1998.

[7] Y. Chen, To. Suel, and A. Markowetz. Efficient query processing in geographic web search engines. In *Proc. of SIGMOD*, pag. 277–288, 2006.

[8] J. Coutaz, J. L. Crowley, S. Dobso, and D. Garlan. Context is key. *Commun. of ACM*, 48(3): 49–53, 2009.

[9] R. Fagin, R. V. Guha, R. Kumar, J. Novak, D. Sivakumar, and A. Tomkins. Multi-structural databases. In *Proc. of PODS*, pag. 184–195, 2005.

[10] R. Fagin, P. G. Kolaitis, R. V. Guha, R. Kumar, J. Novak, D. Sivakumar, and A. Tomkins. Efficient Implementation of Large-Scale Multi-Structural Databases. In *Proc. of SIGMOD*, pag. 958–969, 2005.

[11] M. Fontoura, V. Josifovski, R. Kumar, C. Olston, A. Tomkins, and S. Vassilvitskii. Relaxation in text search using taxonomies. *Proc. of the VLDB Endowment*, 1(1): 672–683, 2008.

[12] N. Koudas, C. Li, A. K. H. Tung, and R. Vernica. Relaxing join and selection queries. In *Proc. of VLDB*, pag. 199–210, 2006.

[13] Y. Roussos, Y. Stavrakas, and V. Pavlaki. Towards a Context-Aware Relational Model. In *Workshop on Context Representation and Reasoning*, 2005.

[14] Y. Stavrakas and M. Gergatsoulis. Multidimensional Semistructured Data: Representing Context-Dependent Information on the Web. In *Proc. of CAiSE*, pag. 183–199, 2002.

[15] Y. Stavrakas, K. Pristouris, A. Efandis, and T. K. Sellis. Implementing a Query Language for Context-Dependent Semistructured Data. In *Proc. of ADBIS*, pag. 173–188, 2004.

[16] K. Stefanidis, E. Pitoura, and P. Vassiliadis. Modeling and Storing Context-Aware Preferences. In *Proc. of ADBIS*, pag. 124–140, 2006.

[17] K. Stefanidis, E. Pitoura, and P. Vassiliadis. Adding Context to Preferences. In *Proc. of ICDE*, pag. 846–855, 2007.

[18] X. Zhou, J. Gaugaz, W. Balke, and W. Nejdl. Query relaxation using malleable schemas. In *Proc. of SIGMOD*, pag. 545–556, 2007.