Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Theoretical Computer Science 311 (2004) 285–323

ELSEVIER

Theoretical Computer Science

www.elsevier.com/locate/tcs

ER modelling from first relational principles

Ernst-Erich Doberkat[a,*], Eugenio G. Omodeo[b]

[a] *Chair for Software-Technology (Informatik 10), University of Dortmund, D-44221 Dortmund, Germany*
[b] *Dipartimento di Informatica, Università degli Studi di L'Aquila, I-67100 L'Aquila, Italy*

Received 13 November 2001; received in revised form 26 August 2003; accepted 10 September 2003
Communicated by M. Wirsing

Abstract

Entity-Relationship (ER) modelling is a popular technique for data modelling. Despite its popularity and widespread use, it lacks a firm semantic foundation. We propose a translation of an ER-model into relation algebra, suggesting that this kind of algebra does provide suitable mechanisms for establishing a formal semantics of ER modelling. The work reported on here deals first with the techniques necessary for the translation, thus constructing a static view of an ER-model in an abstract setting of what might be called *logic without variables*. We then undertake a detailed analysis of the insertion and deletion operations for an ER-model represented in terms of the relation calculus.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Entity-Relationship models (static and dynamic view); Relation algebra; Relational foundations of data modelling

1. Introduction: goals of the present study

The structure of complex data may be specified through an Entity-Relationship (ER) model. This technique for modelling is rather popular, in part because it permits the visualization of the relationship between data, making the structure of their interrelationship easily grasped. The technique is based on Codd's seminal paper [5], it is available in many variants, and it is one of the ancestors of the popular design method *UML*, see e.g. [17].

Despite its popularity, this approach to data modelling lacks a firm formal foundation: it appeals rather to the intuition of a modeler, neglecting the necessity of formally

* Corresponding author. Tel.: +49-231-755-2780; fax: +49-231-755-2061.
E-mail addresses: doberkat@acm.org (E.-E. Doberkat), omodeo@di.univaq.it (E.G. Omodeo).

0304-3975/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2003.09.001

arguing about an ER-model. There are some proposals for a formal semantics of various versions of the model (Section 2 provides a brief overview) which are mainly based on algebraic formalisms like algebraic specifications. A semantics based purely on a calculus coming from logic has not been investigated yet to the authors' knowledge.

1.1. Motivation

The goal of the present study is a systematic investigation into the semantic foundations of ER modelling using relational algebra. Using relational algebra (called *map algebra* in [3]) as a formalization of set theory without variables [21] is intuitively appealing: The naive semantics given to an ER-model is essentially set based—entities are modelled through sets, relations through sets of n-ary tuples, and attributes as functions. It is this naive approach which is formalized here. It will investigate which tools for a relation calculus are necessary for an adequate representation of ER models. Conversely, it provides in this way an assessment for the power of relation algebra based methods for modelling concepts that are in practical use. This means that we have to find a sufficiently rich formal model which has the essential properties of ER models in practical use. It means also that we ought to find mathematical assumptions with which an adequate modelling is feasible (e.g., we will find it convenient from this point of view to represent entities and relations equally as binary relations—which of course does not imply that we suggest treating entities and relations on an equal footing when it comes to modelling data specific for some application).

These questions imply our pursuing two goals:

1. The representation of ER modelling from first relational principles, comparable to constructing a computing system from bits and bytes (one does not do this in practice, of course, either, but the investigations into these concepts yields a firm and sound basis for any kind of practical construction).
2. The actual proof that these relational methods are sufficiently powerful for representing adequately the conceptual foundations for a practically oriented method. The proof for this is given by performing the actual construction, and by providing the actually needed tools and devices.

Both goals complement each other, their realization requires delving deeply into formal details, as will be experienced shortly. They require the translation of an ER model into the target language of a relational calculus.

Our reference on relation algebra is [10] (the most fundamental reference of all is [21]), and we briefly report on it in Section 4. Let us recall here that this calculus, grown out of Boole's "algebra of thought" just a few decades after the latter had been conceived (cf. [19,20]), is an enriched form of it. Today it looks like a simplified version of Codd's relational algebra which also happens to be a denominator common to a number of taxonomic languages proposed for knowledge representation by several authors. Relation algebra can hence be viewed both as an assembly language for knowledge representation and as a common ancestor of the many systems of algebraic logic available today. It has also been used for decomposing relations in a database according to functional dependencies in [14], but these methods have not yet be utilized for a systematic investigation of the dynamic behavior of a data base.

We separate the static structure (the topology) of the ER model from its dynamic counterpart, and we show first how to model the static view using relation algebra. This is obviously not enough, because the dynamic nature of an ER model cannot be described using the static structure alone. Let us have a brief look at abstract data types for just conveying the flavor of our arguments.

1.2. The ADT view

An abstract data type (ADT) encapsulates data and the operations (usually called *methods*) on it. This notion of an ADT is fundamental in object oriented software construction, classes may be considered as special cases of ADTs. This notion is fundamental because it supports data abstraction and permits keeping data and their operation in one physically well defined place. ADTs serve as templates, they are instantiated, and the instances of an ADT are the living capsules data and operations are kept in. The state of an instance is just the collection of specific values the data of this instance are having. The approach *Design by Contract*, so forcefully advocated by Bertrand Meyer [15], and realized in his language Eiffel, goes one step beyond, associating with each ADT specific properties called *invariants*. Operations on an (instance of an) ADT have to respect these invariants in the sense that each operation that starts on an instance which satisfies the invariant leaves the instance in a state which also satisfies it. Each method m of an ADT is associated with a precondition pre_m and a postcondition post_m indicating a contract: entering m such that pre_m is satisfied guarantees leaving m with post_m satisfied. In Hoare's notation of predicate transforms,

$$\{\mathsf{inv} \wedge \mathsf{pre}_m\}\ m\ \{\mathsf{inv} \wedge \mathsf{post}_m\}.$$

Actually, Design by Contract entails more, because inheritance comes into the game through rather involved co- and contravariant rules relating methods from subclasses to super classes, but this will not concern us here.

Call an ADT *proper* iff it has invariants and pre- as well as postconditions, and if the Design by Contract rules are imposed on its methods.

An ER model $\mathscr{M}$ may be considered as an ADT. The data to be stored in an instance are composed of the data stored in the entities, relations and attributes, and the invariant is provided by the conditions imposed on the model's validity (see Definition 5). We should look for three families of operations:

- initializing an instance of $\mathscr{M}$,
- inserting elements into entities and relations,
- deleting elements from entities and relations.

Note that we do not talk about operations but rather about families of them; this is so since an operation like inserting an element into a relation R usually entails other operations (like inserting elements into the domain, and into the codomain of R); there may be more subtle dependencies as well, as we will see.

The invariant to be maintained by these operations constitute the validity of the model; this means that the model before and after one of these families of operations has to conform to the model's declaration. The post conditions are in every case empty, because the operations are all geared towards maintaining the ADT's invariant. Then $\mathscr{M}$ is shown to form in fact a proper ADT.

1.3. Overview

What needs to be done then is to formulate the invariant and the precondition using the language we have chosen for our formalization. After we discuss the version of ER modelling we want to work with in Section 3, we introduce relation algebra in Section 4, there we will also provide some abbreviations that are helpful for the discussions to follow. Section 8 deals with a formulation of the preconditions for insertions. For reasons of managing the complexity, this is reduced first into the bare bones version of an ER model which does not entertain attributes, leading to the notion of a *weakly valid* ER model. It is shown under which conditions weak validity is maintained. Attributes are added to the discussions at that point, introducing the notion of a valid model, and strengthening the preconditions towards keeping validity invariant. A very similar procedure is observed when discussing deletions in Section 9, which quite surprisingly turns out to be easier to handle than insertions. This is mainly due to the fact that most of the interesting properties are downward closed: if a relational expression W observes it, then all relational expressions $V \subseteq W$ do, too. Section 11 proposes further investigations along the lines suggested here, discussing for example how model checking as a technique to ascertain properties of an ER model could be incorporated.

Preliminary versions of this work could be presented at the RelMis 2001 workshop at ETAPS 2001 Genova [16], and at the RelMics'6-TARSKI workshop at the University of Tilburg [9].

2. Related work

The present study stands in line with other approaches to provide a firmly based semantics for ER-models. They are mainly based on algebraic modelling techniques and capitalize on the semantic framework that come with them. The work being done on query algebras for data base models or on the semantics of UML class diagrams pursues different goals, hence is not mentioned here.

Hettler [12] gives a translation of these models into the specification language SPECTRUM, essentially modelling entities as records with attributes as entries, but not taking inheritance into account. The report [4] transforms an ER-diagram into an attributed graph signature, and the integrity constraints into first-order logic formulas. The ER-models considered do not take inheritance explicitly into account. The paper focusses on the dynamic aspects—viz., transactions–through homomorphisms, hence showing how transactions may be caught through an algebraic framework. The formal semantics of an extended ER-model is investigated in [11,13] from a database point of view, proposing the semantics of a database signature as the set of all interpretations; this work does not mention algebraic specifications explicitly. In [7] it is shown how to generate an algebraic specification from an ER-model, hereby carrying the model based semantics of such a specification over to ER-models. Doberkat [8] generalizes this approach somewhat by proposing the colimit of a diagram extracted from an ER-model as the categorial semantics of the model.

Relational algebra has been used for decomposing relations in a database according to functional dependencies in [14]; these methods, however, have not yet be utilized for a systematic investigation of the dynamic behavior of a data base.

3. ER models

We assume the reader to be familiar with ER modelling [5] as a popular and widespread technique for data modelling. Many variants have been discussed (Thalheim's encyclopedic book [22] provides an overview). We will fix now for the rest of the paper the model which will stand in the center of our attention.

3.1. The variant to be considered

We will restrict ourselves to a rather basic variant in which

- All relations are binary, and the only cardinality restriction that may be imposed on a relation is that it is left- or right-unique.
- Inheritance is restricted to single inheritance.
- Relations are assumed to be total. In fact, in the presence of inheritance non-total relations may be transformed into total ones by introducing additional entities for the domain, and for the range, respectively, as will be demonstrated in Section 6.1.
- Attributes are defined on entities only.

This version of ER modelling is a bit more restrictive than the one investigated in [7]. The restrictions can be removed or refined at the cost of a more complicated technical development. We feel, however, that the methods we develop here provide a way of modelling these more complicated situations.

ER modelling may be embedded into the UML when modelling classes before methods are incorporated, thus capturing the static aspects of a class hierarchy. In terms of the UML, we restrict ourselves here to representing these classes when only simple generalization is permitted, associations have to be binary and are without attributes, and multiplicity restrictions may only express left- or right-uniqueness. For practical use, these are of course serious restrictions, for our fundamental investigation, however, they are not, since they permit us to clarify the basic mechanisms. An added layer of technical development will certainly show how to technically cope with additional properties along the lines we are developing here.

3.2. The process model

We are given an instance $\mathscr{M}$ of an ER model which is *valid*, so all constraints formulated in the declaration of the model are satisfied. We want to investigate *change*, namely we want to investigate conditions under which insertions and deletions into $\mathscr{M}$ lead to a valid model again. In order to investigate this for insertions, we assume that we have complete information about the items to be inserted. Thus, if E is an entity, we know the items δ^+E to be inserted into E, yielding $E \cup \delta^+E$ as the new version of this entity. Similarly, we know for relations R the tuples δ^+R to be inserted, and we know for attributes α the changes in $\delta^+\alpha$. What we want to know is, under which conditions for $E, \delta^+E, R, \delta^+R$ and $\alpha, \delta^+\alpha$ the invariance of validity of the instance is maintained. The question arises *mutatis mutandis* for deletions.

Note that the assumption that the change sets δ^+ and δ^- are given does not address the problem of constructing them. When insertion is done interactively, and is not done with care, situations may arise when an infinite sequence of insertions may be necessary; this can be demonstrated through easily found examples. We bypass these complications by postulating that complete information is available from the outset.

4. Relational calculus

Very much like any logical formalism, relation calculus consists of a symbolic language, an intended semantics, a collection of logical axiom schemes (which, according to the intended semantics, are valid, i.e. true in any legal interpretation), and a collection of inference rules.

Making use of a formalism means describing a *universe* $\mathscr{U}$ *of discourse* by means of proper axioms stated in the symbolic language. More axioms means fewer interpretations, and therefore a larger collection of derivable consequences; as an extreme case, in absence of proper axioms, only valid sentences—which are a bit too vacuous to be really useful—are derivable.

Relation calculus was designed to ease reasoning about dyadic relations—*maps*, as we will call them—over an unspecified, yet fixed, universe $\mathscr{U}$. Its language is entirely equational, and ground (i.e., devoid of individual variables); we will therefore concentrate mainly on syntax and intended semantics, and summarize the logical axioms and inference rules in Definition 1.

Relation algebras formalize axiomatically the usual operations on binary relations (like composition or forming the converse), so that binary relations appear as one of several models that are possible for these algebras. We will provide a very brief introduction to these algebras, and we will fix some notations for the reader's convenience.

4.1. Relation algebra

A relation algebra is defined as a Boolean algebra with additional properties that are imposed because a composition relation is available. The version of relation algebras we want to use is defined below; for variants and further developments the reader is encouraged to consult [3, Chapter 2] or [2, Chapter 1].

Definition 1. $\langle \emptyset, \mathbb{1}, \cap, \overline{\cdot}, \boldsymbol{\iota}, ;, \cdot^{\smile} \rangle$ is called a *relation algebra* iff

1. $\langle \emptyset, \mathbb{1}, \cap, \overline{\cdot} \rangle$ is a Boolean algebra with smallest element $\emptyset$, largest element $\mathbb{1}$, intersection (meet) $\cap$, and complementation $\overline{\cdot}$; the associated order relation and union (join), and difference, are denoted by $\subseteq$, $\cup$, and $\setminus$, resp.

Table 1

Symbol	$\emptyset$	$\mathbb{1}$	ι	r_i	$\cap$	;	$\cdot^{\smile}$	$\overline{\cdot}$	$\cup$	$\setminus$
Degree	0	0	0	0	2	2	1	1	2	2
Priority					5	6	7	2	2	2

2. ; is a binary associative operation on the Boolean algebra with ι as the left- and right-neutral element,
3. $\cdot^{\smile}$ is a unary idempotent operation on the Boolean algebra,
4. the following properties hold:
   (a) $(P;Q)^{\smile} = Q^{\smile};P^{\smile}$,
   (b) $(P\cap Q)^{\smile} = P^{\smile}\cap Q^{\smile}$,
   (c) $P;(Q_1\cap Q_2)\subseteq P;Q_1\cap P;Q_2$ ($\cap$-subdistributivity),
   (d) $P;(Q_1\cup Q_2) = P;Q_1\cup P;Q_2$ ($\cup$-distributivity),
5. $P\subseteq Q$ implies $P;R\subseteq Q;R$,
6. $(P;Q)\cap R=\emptyset$ implies $(P^{\smile};R)\cap Q=\emptyset$ (Schröder's Rule).

Relation algebra consists of map equalities $P=Q$, where P and Q are map expressions:

Definition 2. Map expressions are terms of the signature according to the table below, where we have added union $\cup$ as an associative operation, and difference$\setminus$(which will be treated as left-associative operators) for convenience (Table 1):

Here r_i is one of the countably many *map letters* which we assume to be available.

Map letters are used to customizing relation algebra by attaching additional properties through additional axioms for the relation algebra, as we will see in the sequel.

An *interpretation* $\mathscr{I}$ over a universe $\mathscr{U}$ maps each map expression to a subset of the Cartesian square $\mathscr{U}^2 \equiv_{\mathrm{Def}} \mathscr{U}\times\mathscr{U}$ such that e.g.

$$\begin{array}{rl|rl} \emptyset^{\mathscr{I}} & =\emptyset & (P\cap Q)^{\mathscr{I}} & = P^{\mathscr{I}}\cap Q^{\mathscr{I}}, \\ \mathbb{1}^{\mathscr{I}} & =\mathscr{U}^2 & (P;Q)^{\mathscr{I}} & = P^{\mathscr{I}};Q^{\mathscr{I}}, \\ \iota^{\mathscr{I}} & =\Delta & (Q^{\smile})^{\mathscr{I}} & = (Q^{\mathscr{I}})^{\smile}. \end{array}$$

Here Δ is the diagonal $\{\langle a,a\rangle \mid a\in\mathscr{U}\}$ of $\mathscr{U}$, and the operations on the right-hand side are the familiar ones manipulating relations over sets. Hence e.g. $\cup$-distributivity translates into the set equality $R;(S_1\cup S_2)=R;S_1\cup R;S_2$ which is familiar for the relations $R\subseteq A\times B$ and $S_1,S_2\subseteq B\times C$ for sets A,B and C.

Adding new axioms through fixing properties of map letters has the effect of restricting interpretations: they have to satisfy the additional properties for the interpretation of the map letters, which in turn also have to be provided.

For conciseness, we use some abbreviations which are listed in Table 2.

For example, $\mathsf{Coll}(E)$ says that $E^{\mathscr{I}}$ is supposed to consist of pairs of the form $\langle a,a\rangle$ for some $a\in\mathscr{U}$, $\mathsf{Total}(E)$ indicates that $E^{\mathscr{I}}$ is (left-) total, hence that for each

Table 2

Notation	Expression	Note
$\mathsf{Coll}(R)$	$R \subseteq \iota$	$R^{\mathscr{I}}$ represents a collection
$\mathsf{Total}(R)$	$R;\mathbb{1} = \mathbb{1}$	$R^{\mathscr{I}}$ is a (left-) total relation
$\mathsf{dom}(R)$	$R;\mathbb{1} \cap \iota$	Domain of R
$\mathsf{img}(R)$	$\mathbb{1};R \cap \iota$	Range/image of R
$\mathsf{RUniq}(R)$	$\mathsf{Coll}(R^{\smile};R)$	$R^{\mathscr{I}}$ is a partial map
$\mathsf{LUniq}(R)$	$\mathsf{RUniq}(R^{\smile})$	$(R^{\smile})^{\mathscr{I}}$ is a partial map
$\mathsf{NonVoid}(R)$	$\mathsf{Total}(\mathbb{1};R)$	$R^{\mathscr{I}} \neq \emptyset$
$\mathsf{Snglt}(R)$	$\mathsf{NonVoid}(R)\ \&\ \mathsf{RUniq}(\mathbb{1};R)\ \&\ \mathsf{RUniq}(R^{\smile})$	$R^{\mathscr{I}}$ is a singleton
$\mathsf{DomSub}(R,S)$	$R;\mathbb{1} \subseteq S;\mathbb{1}$	Domain containment
$\mathsf{ImgSub}(R,S)$	$\mathbb{1};R \subseteq \mathbb{1};S$	Range containment
$\mathsf{Const}(R)$	$\mathsf{Snglt}(R)\ \&\ \mathsf{Coll}(R)$	$R^{\mathscr{I}}$ represents $\{\langle a,a\rangle\}$
$\times_{j=1}^{n+1} R_j$	$\left(\times_{j=1}^{n} R_j\right);R_{n+1}$	

$a \in \mathscr{U}$ there is some $b \in \mathscr{U}$ with $\langle a,b\rangle \in E^{\mathscr{I}}$. The reader is invited to formulate these expressions in terms of set-theoretic relations.

Some identities and inequalities will be particularly helpful in the sequel; we collect them here for easier reference, and point the reader to [21] and to [18].

Lemma 1. *Let P, Q and R be map expressions, then*
1. $\mathsf{RUniq}(R) \Leftrightarrow R;(P \cap Q) = R;P \cap R;Q$,
2. $(P \cup Q)^{\smile} = P^{\smile} \cup Q^{\smile}$,
3. $(P;Q)\setminus(R;Q) \subseteq (P\setminus R);Q$
4. $P, Q \subseteq \iota$, *then*
   (a) $P^{\smile};Q = \emptyset$, *provided* $P \cap Q = \emptyset$,
   (b) $(P\setminus Q);\mathbb{1} = (P;\mathbb{1})\setminus(Q;\mathbb{1})$,
5. $Q;R \subseteq S \Leftrightarrow Q^{\smile};\overline{S} \subseteq \overline{R} \Leftrightarrow \overline{S};R^{\smile} \subseteq \overline{Q}$ (*Schröder's Cycle Rules*),
6. $\overline{P^{\smile}} = (\overline{P})^{\smile}$.

Map letters. We assume that we have a countably infinite provision of map letters $\mathsf{r}_1, \mathsf{r}_2, \ldots$ at our disposal of which we reserve the first T for system purposes.

4.2. Projections and flat tuples

Tuples of length >2 forcibly enter into the study of database systems, if only because the operation of inserting an entity e into a database often causes the simultaneous assignment of a tuple of values to the attributes of e. To cope with this while avoiding the complications that would result from the treatment of relations of arity greater than 2, we want the universe $\mathscr{U}$ of discourse to include $\mathscr{A}^*$—viz., the set of all finite-length sequences whose components are entities drawn from $\mathscr{A}$ and are in some sense "atomic". We will assume for simplicity that $\mathscr{U} = \mathscr{A} \cup \mathscr{A}^*$ and $\mathscr{A} \cap \mathscr{A}^* = \emptyset$, and, to avoid triviality, that $\mathscr{A} \neq \emptyset$. We indicate by ε the null tuple in $\mathscr{A}^*$, and put $\mathscr{A}^+ =_{\mathrm{Den}} \mathscr{A}^* \setminus \{\varepsilon\}$.

Two operations on non-null tuples $t_0 t_1 \cdots t_m$ are essential, namely *head isolation*—which determines the first component t_0 of any given such tuple—and *tail extraction*—which determines the sub-tuple $t_1 \cdots t_m$ resulting from removal of the first component. Let us designate these operations by λ and ϱ, resp. Moreover, let us represent by υ and ε the collection $\mathscr{A}$ of atomic entities and the null tuple: more precisely, $\upsilon^{\Im} = \{\langle a,b\rangle \in \mathscr{A}^2 \mid a=b\}$ and $\varepsilon^{\Im} = \{\langle \varepsilon,\varepsilon\rangle\}$ in our intended interpretation $\Im$.

We can state that

- ε designates a singleton and diagonal map, by the condition $\mathsf{Const}(\varepsilon)$;
- υ represents a non-empty collection $\mathscr{A}$ of entities distinct from the entity, ε, represented by ε, by means of the conditions

$$\mathsf{Coll}(\upsilon)\ \&\ \mathsf{NonVoid}(\upsilon)\ \&\ \varepsilon \cap \upsilon\ =\ \emptyset,$$

- λ, ϱ designate functions $\lambda^{\Im}$ and $\varrho^{\Im}$ whose common domain $\mathscr{A}^+$ is the complement of $\mathscr{A} \cup \{\varepsilon\}$ in $\mathscr{U}$, by means of the conditions

$$\mathsf{RUniq}(\lambda)\ \&\ \mathsf{RUniq}(\varrho)\ \&\ \lambda\,;\mathbb{1} = \varrho\,;\mathbb{1} = \overline{(\upsilon \cup \varepsilon)\,;\mathbb{1}}$$

- $\varrho(p)$ belongs to $\mathscr{A}^*$ for all p, by means of the condition $\upsilon \cap \mathbb{1}\,;\varrho = \emptyset$;
- for all a in $\mathscr{A}$ and all q in $\mathscr{A}^*$ there is a tuple p with $\lambda(p) = a$ and $\varrho(p) = q$, by means of the condition

$$\upsilon\,;\overline{\mathbb{1}\,;\upsilon} = \lambda^{\smile}\,;\varrho$$

- the function $p \mapsto \langle \lambda(p), \varrho(p)\rangle$ is injective, by the condition

$$\mathsf{Coll}(\lambda\,;\lambda^{\smile} \cap \varrho\,;\varrho^{\smile}).$$

To sum all these conditions up, let us introduce the notation

$$\begin{aligned}\mathsf{HeadTail}(L,R,Y,E) \equiv_{\mathrm{Def}}\ & \mathsf{RUniq}(L)\ \ \&\ \ \mathsf{RUniq}(R)\ \ \&\ \ \mathsf{Const}(E)\\ & \&\ \ \mathsf{Coll}(Y)\ \ \&\ \ \mathsf{NonVoid}(Y)\ \ \&\ \ E \cap Y = \emptyset\\ & \&\ \ Y \cap \mathbb{1}\,;R = \emptyset\ \ \&\ \ L\,;\mathbb{1} = R\,;\mathbb{1} = \overline{(E \cup Y)\,;\mathbb{1}}\\ & \&\ \ Y\,;\overline{\mathbb{1}\,;Y} = L^{\smile}\,;R\ \ \&\ \ \mathsf{Coll}(L\,;L^{\smile} \cap R\,;R^{\smile}),\end{aligned}$$

so that we can concisely state everything by the single requirement $\mathsf{HeadTail}(\lambda,\varrho,\upsilon,\varepsilon)$.

After noticing that

$$\left(\times_{j=1}^{i-1}\, \varrho\right)\ ;\lambda$$

designates the operation of extracting the ith component of a tuple, let us also provide a handy characterization of h-tuples:

$$\begin{aligned} i^{\mathsf{th}}(L,R) &\equiv_{\mathrm{Def}}\ \times_{j=1}^{i-1} R\,;L,\\ h - \mathsf{tuples}(R) &\equiv_{\mathrm{Def}}\ \mathsf{img}(R) \cap \mathsf{dom}(\,h^{\mathsf{th}}(R,R)\,) \backslash \mathsf{dom}(\,(h+1)^{\mathsf{th}}(R,R)\,).\end{aligned}$$

(The notation i^{th} and h-tuples is a little sloppy but rather expressive, so that we trust the reader will appreciate it over a more formal one).

Thus, under assumption that $\mathsf{HeadTail}(\lambda, \varrho, \upsilon, \varepsilon)$ holds, the map expression $h\text{-}\mathsf{tuples}(\varrho)$ represents the collection of all tuples $t_1 \cdots t_h$ in $\mathscr{A}^*$ for any natural number h.

Notice that for the sake of completeness we should also include in our specification of $\lambda, \varrho, \upsilon, \varepsilon$ an induction principle reflecting our assumption that $\mathscr{U}$ is the smallest superset of $\mathscr{A}$ which is closed w.r.t. tuple formation. One way of stating induction is by means of an equality scheme ensuring that

$$P = \mathbb{1}$$

follows from

$$((\varepsilon \cup \upsilon);\mathbb{1} \cup (\lambda;P \cap \varrho;P)) \backslash P = \emptyset$$

for all P, namely:

$$\mathbb{1};((\varepsilon \cup \upsilon);\mathbb{1} \cup \lambda;P \cap \varrho;P \backslash P);\mathbb{1} \cup P;\mathbb{1} = \mathbb{1}$$

(cf. [10, Section 6], and also [1, p. 175, Law S8]).

5. Preparations

Now let an ER model $\mathscr{M}$ be given. All information concerning $\mathscr{M}$ can be found in a declaration which represents the static information about the model, and which permits stating the validity of an instantiation for $\mathscr{M}$. Separating concerns, we concentrate for the time being on entities and relations, attributes will be added later on, as the technical development evolves.

5.1. Graphical representation

ER models are usually presented through graphical representations where entities are shown as boxes, and relations as diamonds. Every diamond represents a *dyadic* relation—a map in our terminology—and that no diamond bears any attributes.

We need to regard one parameter of a relation as first, and the other one as second, which we do according to the numbering stipulation detailed in Fig. 1.

Uniqueness of names is assumed to its fullest extent; viz., we are forbidding synonymy not only between (entity-)boxes, between diamonds and between ovals, but also between a box and a diamond, etc.

Digression on a little anomaly that may be caused by IsA. An IsA-edge connecting box E to box F indicates that the set $\mathbf{E}$ of entities designated by E is included at all times in the one, $\mathbf{F}$, designated by F; the IsA-relation is discussed in greater detail in Section 5.4. Unless this inclusion could be satisfied, at least occasionally, as a strict inclusion, distinguishing between E and F would hardly make any sense, and the ER-model would be somehow redundant.

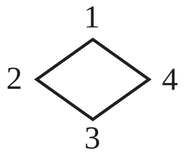
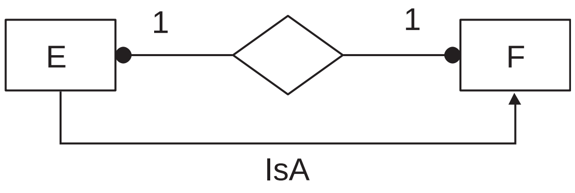


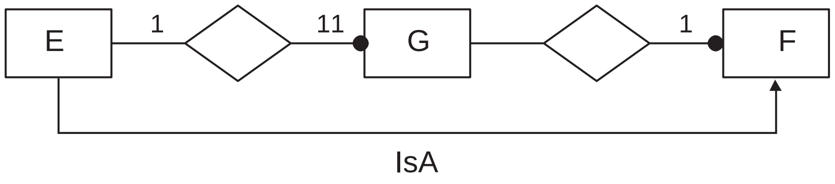
Fig. 1. Numbering counter clockwise.

Notice, however, that the following situation implies that E and F designate the same entity:



(Indeed, $\mathbf{F} = \mathbf{E}$ ensues from $|\mathbf{E}| = |\mathbf{F}|$ and $\mathbf{E} \subseteq \mathbf{F}$ when $\mathbf{F}$ is known to be finite).

Generalizations of this situation are easily found, e.g.

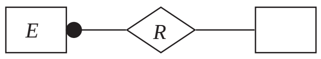


(where $\mathbf{F} = \mathbf{E}$ ensues from $|\mathbf{F}| \leqslant |\mathbf{G}| \leqslant |\mathbf{E}|$ and $\mathbf{E} \subseteq \mathbf{F}$ in view of the finiteness of $\mathbf{F}$—note that $\mathbf{G} = \mathbf{E}$ is not entailed, though!). Admitting slightly defective ER-models of the above kind would cause some marginal conceptual difficulties in the part of this research which regards dynamics which will be discussed in Sections 8 and 9. However, since we have the reasonable expectation that any similar anomaly can be revealed by a simple semantic check, so as to be then corrected, we assume that this check is available to us and we do not digress any further on this issue.

5.2. The basic model

If E is the domain of relation R with F as its co-domain, then we will assume that E and F are tight in the sense of the following definition:

Definition 3. We call E the *tight domain* of R in $\mathscr{M}$ if $\mathscr{M}$ requires that for each $e \in E$ there exists an element f in the codomain of R such that $\langle e, f \rangle \in R$. In diagrams this is indicated through a dot.



The definition of *tight codomain* is symmetric.

Introducing another shorthand notation, $E \bullet\!\!-\!\!- R -\!\!-\!\!\bullet F$ indicates that both E and F are tight.

In what follows, entities and relations will be considered an element of a fixed (but anonymous) relation algebra. An entity E is then represented through $\mathsf{Coll}(E)$, hence consists of pairs the first and the second component of which agree. This representation permits us to represent entities and relations in a uniform manner, thus allowing us to work in a uniform framework. This is of course a technical assumption which is not mirrored in the conceptual use of ER-modelling; we will have to pay a price for it in our exposition, since we now have always to distinguish "proper" relations from those that represent entities.

Tightness, as expressed through

$$E \bullet\!\!-\!\!- R -\!\!-\!\!\bullet F$$

translates to

$$\mathsf{DotDot}(E,R,F) \equiv_{\mathrm{Def}} \mathsf{Coll}(E) \;\&\; \mathsf{Coll}(F) \;\&\; \mathsf{DomSub}(E,R) \;\&\; \mathsf{ImgSub}(F,R).$$

Either relation $\bullet\!\!-\!\!-$ or $-\!\!-\!\!\bullet$ may be strengthened to $\overset{1}{\bullet\!\!-\!\!-}$ and $\overset{1}{-\!\!-\!\!\bullet}$, resp., indicating uniqueness. Thus $E \overset{1}{\bullet\!\!-\!\!-} R$ means in addition to $E \bullet\!\!-\!\!- R$ that

$$\langle x,y\rangle \in R \wedge \langle x',y\rangle \in R \Rightarrow x = x'$$

holds, which may be translated conveniently into $\mathsf{LUniq}(R)$. Similarly, $R \overset{1}{-\!\!-\!\!\bullet} F$, which means

$$\langle x,y\rangle \in R \wedge \langle x,y'\rangle \in R \Rightarrow y = y'$$

is translated into $\mathsf{RUniq}(R)$.

Note that either of these conditions depends only on the relation, not on the domain or the codomain.

The different way a relation relates to its domain and its codomain may be captured through the suitable combination of macros which are comprehensively listed in Table 3.

5.3. Adding place holders

It may sometimes happen that information is incomplete: an element x is inserted into entity E, and $E \bullet\!\!-\!\!- R -\!\!-\!\!\bullet F$ holds, but there is no y in F so that $\langle x,y\rangle$ is to be inserted into R. This then would violate the condition $E;\mathbb{1} \subseteq R;\mathbb{1}$. There may even occur some unpleasant situations when place holders are not admitted. Consider Fig. 2,

Table 3

Situation	Characterization
$E \overset{1}{\bullet\!-} R \overset{1}{-\!\bullet} F$	$\mathsf{DotDot}(E,R,F)\ \&\ \mathsf{LUniq}(R)\ \&\ \mathsf{RUniq}(R)$
$E \overset{1}{\bullet\!-} R -\!\bullet F$	$\mathsf{DotDot}(E,R,F)\ \&\ \mathsf{LUniq}(R)$
$E \bullet\!- R \overset{1}{-\!\bullet} F$	$\mathsf{DotDot}(E,R,F)\ \&\ \mathsf{RUniq}(R)$

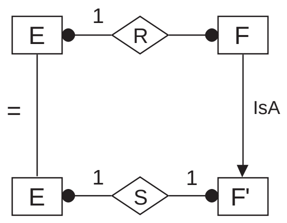


Fig. 2. Possible circularity.

where E and F are assumed to be different entities. Insert *one* into E, then $\langle one, two\rangle$ into R; then *two* must be new to F. Insert it into F, then it will be inserted into F' which requires the insertion of a pair $\langle three, two\rangle$ into S; *three* must be new to E. In this way a loop is created which will not terminate.

For enabling insertions also under somewhat problematic conditions, we postulate the existence of place holders which are collected in a relation P, so that in the situation considered $\langle x, *\rangle$ with $* \in P$ would be inserted into R. We assume that $\mathsf{Coll}(P)$ holds, and that the entities are free of place holders, thus $E \cap P = \emptyset$ is true for each entity E (note that this implies both $\mathbb{1};E \cap \mathbb{1};P = \emptyset$ and $E;\mathbb{1} \cap P;\mathbb{1} = \emptyset$ by Lemma 1). Let

$$\mathsf{Entity}(P,E) \equiv_{\mathrm{Def}} \mathsf{Coll}(E)\ \&\ E \cap P = \emptyset$$

denote that E is an entity.

Constraints on place holders. We need some constraints on the use of place holders that reflect their adequate use.

1. No placeholder occurs twice as the first or the second component of a pair in a relation R. Put

   $$\mathsf{NoTwice}(P,R) \equiv_{\mathrm{Def}} P \cap (R \cap R;\bar{\imath});\mathbb{1} = \emptyset,$$

   then

   $$\mathsf{NoTwice}(P,R)\ \&\ \mathsf{NoTwice}(P,R^{\smile})$$

   should hold,

2. No placeholder occurs in two different relations R, S as the first components of a pair, which is formulated as

$$\mathsf{NoBoth}(P, R, S) \equiv_{\text{Def}} P \cap R;\mathbb{1} \cap S;\mathbb{1} = \emptyset.$$

3. No placeholder occurs both as the first component in relation R and as the second component in relation S, hence

$$\mathsf{NoFirstSecond}(P, R, S) \equiv_{\text{Def}} \mathsf{NoBoth}(P, R, S^{\smile}).$$

4. No pair in a relation has place holders on both sides, thus

$$\mathsf{NoSamePair}(P, R) \equiv_{\text{Def}} R \cap P;\mathbb{1} \cap \mathbb{1};P = \emptyset.$$

5. The situation $\langle *, y\rangle$ and $\langle x, y\rangle$ with $x \neq *$ (and, for symmetry, in the second component) does not occur; this is captured through

$$\mathsf{NoDoubleFirst}(P, R) \equiv_{\text{Def}} R;R^{\smile} \cap P;\mathbb{1} \cap \mathbb{1};\overline{P} = \emptyset$$

and

$$\mathsf{NoDoubleSecond}(P, R) \equiv_{\text{Def}} R^{\smile};R \cap \mathbb{1};P \cap \overline{P};\mathbb{1} = \emptyset.$$

Summing up: If $\{R_1, \ldots, R_k\}$ are the identifiers for all the relations in play, the conjunction

$$\mathsf{PlaceHolder}(P, \{R_1, \ldots, R_k\})$$

should hold, where

$$\begin{aligned}\mathsf{PlaceHolder}(P, \{R_1, \ldots, R_k\}) \equiv_{\text{Def}} \; & \&_{i=1}^{k} \mathsf{NoTwice}(P, R_i) \mathbin{\&} \mathsf{NoTwice}(P, R_i^{\smile}) \\ \& \; & \&_{i=1}^{k} \&_{j=i+1}^{k} \mathsf{NoBoth}(P, R_i, R_j) \\ \& \; & \&_{i=1}^{k} \&_{j=i+1}^{k} \mathsf{NoFirstSecond}(P, R_i, R_j) \\ \& \; & \&_{i=1}^{k} \mathsf{NoSamePair}(P, R_i) \\ \& \; & \&_{i=1}^{k} \mathsf{NoDoubleFirst}(P, R_i) \\ \& \; & \&_{i=1}^{k} \mathsf{NoDoubleSecond}(P, R_i).\end{aligned}$$

We reserve the map letter π for place holders.

5.4. Inheritance

Immediate inheritance between entities is given through the IsA-relation. Hence

$$E \;\; \mathsf{IsA} \;\; F$$

translates into

$$\mathsf{Inherits}(P, E, F) \equiv_{\text{Def}} \mathsf{Entity}(P, E) \mathbin{\&} \mathsf{Entity}(P, F) \mathbin{\&} E \subseteq F.$$

This is in accordance with the familiar view of generalization as inclusion.

There are some restrictions to be observed concerning the IsA-relation, mainly acyclicity and single inheritance. This will be discussed now.

The IsA-graph is assumed to be acyclic; i.e.,

$$\mathsf{IsA}^{\smile} \cap \times_{i=1}^{n} \mathsf{IsA} = \emptyset$$

must hold for all $n \geqslant 0$. Moreover, we assume that at most one IsA-edge exits from each node:

$$\mathsf{IsA}^{\smile} \mathbin{;} \mathsf{IsA} \subseteq \boldsymbol{\iota}.$$

The latter assumption reflects two facts: on the one hand, we are taking into account *single inheritance* only; on the other hand, we are *forbidding shortcuts* in inheritance chains, by requiring that

$$\forall n \geqslant 0 : \mathsf{IsA} \cap \times_{i=1}^{n+2} \mathsf{IsA} = \emptyset.$$

Since an acyclic function defined on a finite set cannot be total, the IsA-graph will have nodes of out degree 0. Such "sink" nodes—which represent objects of maximal genericity—are usually called *roots* in the object-oriented approach. Clearly, there would be no loss of generality in assuming that there is exactly one root.

6. Renderings of ER-models in relation algebra

This section will show how to translate an ER model $\mathscr{M}$ into a set of map equalities. This happens in two steps: first $\mathscr{M}$ is normalized by making all relations left total as well as right total, the second step iterates over the model and forms the equalities, which in turn are formulated through the macros formulated above.

6.1. Normalization

Here is the preprocessing algorithm that converts any given ER-model (devoid of relation-attributes) into one that better fits our translation purposes:

Phase 0: Detect all pairs E, F with $E \not\equiv F$ such that, due to the IsA-anomaly, E and F must designate the same class of entities. For every such pair, coalesce E with F.

Phase 1 (*Factorization*): Make all relations both left- and right-total, by repeated use of the rewriting rules in Fig. 3 (where $\mathtt{s}, \mathtt{d} \in \{1, *\}$, and E', F' are fresh names).

After this normalization we may (and do) assume that both domain and codomain are tight. This technical assumption helps avoiding the discussion of cumbersome special cases. Fig. 3 is slightly redundant for reasons of symmetry.

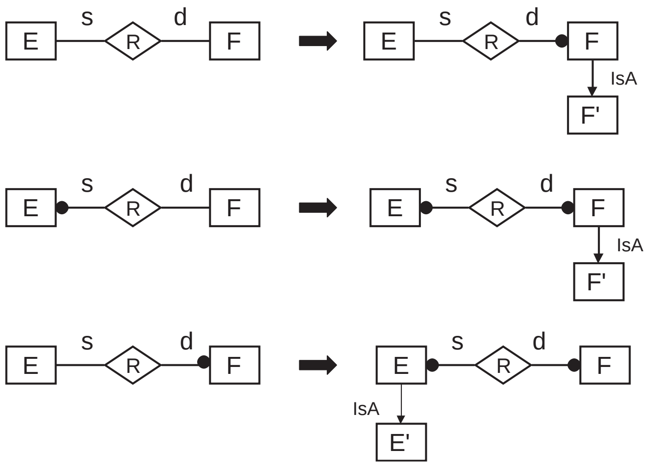


Fig. 3. Rewriting rules for normalization.

6.2. Translation

Here is one way of translating into a set of map equalities an ER-model $\mathscr{M}$ resulting from the normalization process presented above:

A. Introduce mutually distinct symbols $\pi, \lambda, \varrho, \varepsilon$ which also are distinct from υ and from any identifier in $\mathscr{M}$. Postulate that

$$\mathsf{HeadTail}(\lambda, \varrho, \upsilon, \varepsilon) \quad \& \quad \mathsf{PlaceHolder}(\pi, \mathsf{R})$$

(cf. Sections 4.2 and 5.3), where R is the set of all attribute-, and relation-identifiers in $\mathscr{M}$.

We are as above regarding the elements of $\mathsf{R} \cup \{\pi, \lambda, \varrho, \upsilon, \varepsilon\}$—as well as the entity identifiers in $\mathscr{M}$—as symbols drawn from the alphabet of map letters r_i. This identification with map letters induces a linear ordering between identifiers which we will tacitly assume.

B. For every IsA-edge between E and F in $\mathscr{M}$, require that $\mathsf{dom}(E) \subseteq \mathsf{dom}(F)$. For every entity identifier F which has no issuing IsA-edge, require that $\mathsf{dom}(F) \subseteq \upsilon$.

C. For every entity identifier F, let $A_1, \ldots, A_h$ (with $h > 0$) be all of the distinct attribute identifiers that refer to F in $\mathscr{M}$. Require that

$$\&_{i=1}^{h} \mathsf{Attr}(A_i, F, \upsilon) \quad \& \quad \mathsf{Key}(\{A_1, \ldots, A_h\}, \lambda, \varrho, \pi)$$

holds, where the following abbreviating notation is adopted (see also Section 4.2):

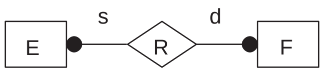
$$\mathsf{Attr}(B, E, Y) \equiv_{\mathrm{Def}} \mathsf{RUniq}(B) \;\&\; \mathsf{dom}(B) \subseteq E \;\&\; \mathsf{img}(B) \subseteq Y\,,$$

$$\mathsf{Key}(\{B_1, \ldots, B_\ell\}, L, R, P) \equiv_{\mathrm{Def}} \mathsf{RUniq}\left(\bigcap_{j=1}^{\ell} j^{\mathrm{th}}(\lambda, \varrho) \,;\, B_j^{\smile}\right)$$

$$\& \quad \&_{j=1}^{\ell} \mathsf{img}(B_j) \cap P \;=\; \emptyset\,.$$

Moreover, for each mandatory attribute A_i, require that $\mathsf{Attr}(A_i, F, \upsilon)$ & $F \subseteq \mathsf{dom}(A_i)$; and, for each key $A_{i_1}, \ldots, A_{i_k}$ (where $k > 0$ and the A_{i_j} are mandatory attributes drawn from among $A_1, \ldots, A_h$), require that $\mathsf{Key}(\{A_{i_1}, \ldots, A_{i_k}\}, \lambda, \varrho, \pi)$.

D. For every part



of $\mathscr{M}$, require that

$$\mathsf{dom}(E) = \mathsf{dom}(R) \backslash \pi \ \ \& \ \ \mathsf{dom}(F) = \mathsf{img}(R) \backslash \pi \, .$$

Moreover, if $\mathsf{d} \equiv 1$, then require that $\mathsf{RUniq}(R)$; and, if $\mathsf{s} \equiv 1$, then require that $\mathsf{RUniq}(R^{\smile})$.

7. Map letters

We assume that we have countably many map letters $\mathsf{r}_1, \mathsf{r}_2, \ldots$ at our disposal, of which we reserve the first T initially for system purposes. We have reserved already π for place holders, λ and ρ for head isolation, and for tail extraction, and υ resp. ε for the collection $\mathscr{A}$ of atomic entities and the null tuple, see Section 4.2. Some additional reservations will have to be made now.

7.1. Layout: blocks

The map letters with indices beyond T will be used for the ER model under consideration in the following way. $\mathsf{r}_{T+1}, \ldots, \mathsf{r}_{T+S}$ will be reserved for entities, the next block $\mathsf{r}_{T+S+1}, \ldots, \mathsf{r}_{T+S+B}$ of B map letters will be reserved for relations, and finally we will reserve the next block of A map letters for attributes. In case of an insertion or a deletion, we reserve the next block of S map letters for the δ^+ resp. δ^--values for entities, the next block of size B for those values for relations, and finally the next block A map letters for attributes. We continue the sequence with the results, according to the following scheme (with $\Sigma := S + B + A$): if entity E corresponds to map letter r_{T+i} with $\delta^+ E$ corresponding to $\mathsf{r}_{T+\Sigma+i}$, then $E \cup \delta^+ E$ will be deposited at $\mathsf{r}_{T+2\cdot\Sigma+i}$. In the same linear way—proceeding in a block wise fashion—we deposit the changed values for relations and attributes. The arrangement of map letters is indicated in Fig. 4, where corresponding map letters are represented as lying on a vertical line.

7.2. Keeping track

We keep a record of the respective relations between entities and relations through a map

$$\mathsf{Track} : \{T + S + 1, \ldots, T + S + B\} \to \{T + 1, \ldots, T + S\} \times \{T + 1, \ldots, T + S\}$$

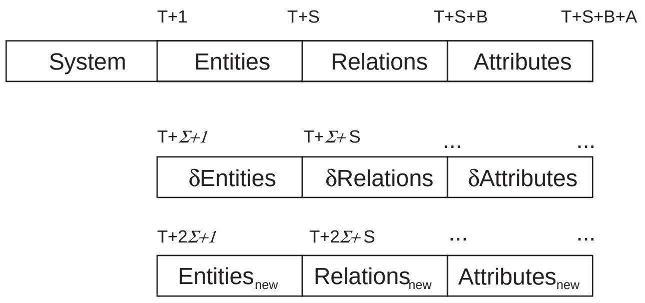


Fig. 4. Arrangement of Map Letters. The map letters $1\dots T$ intended for system purposes are set graphically apart from the recurring blocks for entities, relations and attributes. Corresponding map letters are drawn below each other for greater clarity.

upon setting

$$\mathsf{Track}(t) = \langle i,j\rangle \;\Leftrightarrow\; \mathsf{r}_i \bullet\!\!-\!\!- \mathsf{r}_t -\!\!-\!\!\bullet \mathsf{r}_j.$$

Define for the relational index $t \in \{T+S+1,\dots,T+S+B\}$ that

$$t \in \mathsf{LeftOne} \;\Leftrightarrow\; \exists i,j : \mathsf{r}_i \overset{1}{\bullet\!\!-\!\!-} \mathsf{r}_t -\!\!-\!\!\bullet \mathsf{r}_j$$

and

$$t \in \mathsf{RightOne} \;\Leftrightarrow\; \exists i,j : \mathsf{r}_i \bullet\!\!-\!\!- \mathsf{r}_t \overset{1}{-\!\!-\!\!\bullet} \mathsf{r}_j.$$

Through these sets we get access to left- and right-unique relations.

Again, $\mathsf{Track}, \mathsf{LeftOne}$ and $\mathsf{RightOne}$ can be shifted linearly along each Σ-block of indices.

The reflexive and transitive closure IsA^* of the inheritance relation is recorded through a reflexive and transitive relation Up on the set $\{T+1,\dots,T+S\}$; note that this relation may be shifted linearly to the sets $\{T+k\cdot\Sigma+1,\dots,T+k\cdot\Sigma+S\}$. The necessary properties of IsA^* are described in 5.3.

Attributes. If entity E is represented by map letter r_i with $i \in \{T+1,\dots,T+S\}$, then

$$\mathsf{Attributes}(i) \subseteq \{T+S+B+1,\dots,T+S+B+A\}$$

is the set of map letters that are associated with E's attributes. Clearly,

$$\{\mathsf{Attributes}(i) \mid T+1 \leqslant i \leqslant T+S\}$$

forms a partition of the set $\{T+S+B+1,\dots,T+S+B+A\}$. The set

$$\mathsf{Mandatory}(i) \subseteq \mathsf{Attributes}(i)$$

contains the indices of all mandatory attributes (those attributes which are defined on all of E), and the set

$$\mathsf{Key}(i) \subseteq \mathsf{Mandatory}(i)$$

contains all indices of the key attributes. We assume having only one set of key attributes per entity. It would be easy to work with a varying number of sets of keys for each entity, but this would only complicate the notation without adding any new ideas.

When we execute an insertion or a deletion, we change the contents assigned to the map letters by manipulating the extension of the corresponding data containers. Our block oriented scheme ensures that this process can be repeated without much ado by simply changing the base address where it all begins from T to $T + 2 \cdot \Sigma$.

8. Insertions: validity

This section formulates the validity of an ER model, first without taking attributes into account. This leads to the notion of *weak validity*. Conditions are formulated under which the weak validity of an ER model is preserved. Then we add attributes to our discussion, and the notion of validity is formulated. Again, conditions are given under which the attributes of the model arising from insertions satisfy the constraints, this time leading to the instance of a valid ER model.

8.1. Weak validity

An instance $\mathscr{M}$ of the ER model under consideration is *weakly valid* iff it satisfies all the constraints imposed on the entities and the relations laid down in the model's declaration. This can be described now formally:

Definition 4. The instance $\mathscr{M}$ is called *weakly valid* iff

$$\begin{aligned}\&_{T+S+1 \leqslant t \leqslant T+S+B} \mathsf{DotDot}(\mathsf{r}_{\pi_1(\mathsf{Track}(t))}, \mathsf{r}_t, \mathsf{r}_{\pi_2(\mathsf{Track}(t))}) \;&\&\; \&_{t \in \mathsf{LeftOne}} \mathsf{LUniq}(\mathsf{r}_t)\\ &\&\; \&_{t \in \mathsf{RightOne}} \mathsf{RUniq}(\mathsf{r}_t)\\ &\&\; \&_{T+1 \leqslant i \leqslant T+S}\ \mathsf{Entity}(\pi, \mathsf{r}_i)\\ &\&\; \mathsf{PlaceHolder}(\pi, \{\mathsf{r}_{T+S+1}, \ldots, \mathsf{r}_{T+S+B}\})\\ &\&\; \&_{\langle i,j \rangle \in \mathsf{Up}}\ \mathsf{r}_i \subseteq \mathsf{r}_j.\end{aligned}$$

Note that weak validity is formulated using a fixed base address T, which, however, has not been incorporated into the notation that is already cluttered enough.

The definition above gives a formal description of the invariant we have to maintain when inserting a new element. It makes sure that each relation is tight, collects left unique resp. right unique relations, guarantees that the entities are at their proper place, relates placeholders and relations properly and checks that the IsA-relation is set properly.

8.2. Maintaining weak validity

The insertions to be performed start from a weakly valid ER model and should of course maintain weak validity as an invariant; this issue has been hinted at in Section 1.2 already. We will need some preconditions. Before formulating them, however, we elaborate on the insertions proper. If E is an entity, and δ^+E contains the insertions into E, then $E \cup \delta^+E$ will be formed, and this will be the new version of this entity. It is a bit more complicated with a relation R, since we cannot simply form $R \cup \delta^+R$ without running the risk of violating $\mathsf{NoDoubleFirst}(P, R \cup \delta^+R)$ or $\mathsf{NoDoubleSecond}(P, R \cup \delta^+R)$. Hence we have to clean up R by removing candidates for violations; they are easily seen to belong to

$$(\mathbb{1};\pi \cap \delta^+R;\mathbb{1}) \cup (\pi;\mathbb{1} \cap \mathbb{1};\delta^+R).$$

Thus we work with

$$[R, \delta^+R] \equiv_{\mathrm{Def}} R \backslash ((\mathbb{1};\pi \cap \delta^+R;\mathbb{1}) \cup (\pi;\mathbb{1} \cap \mathbb{1};\delta^+R))$$

instead of R and form $[R, \delta^+R] \cup \delta^+R$ as the new version of relation R. Occasionally we will replace the map letter π by the free variable P; the expression then will be denoted by $[R, \delta^+R]_P$.

For describing under which conditions weak validity is maintained, we need to set the stage by providing some technical preparations. We first show under which conditions the property of domain resp. codomain containment after insertions remains intact. They will be put to use when invariance of weak validity (Proposition 1) and of validity (Proposition 2) under insertions are formulated.

The following lemma states under which conditions domain containment is maintained under insertions. It makes sure that among others domain containment should be true before the insertion, and that the inserted parts are related through domain containment, that the relation the domain of which is contained in the other one represents an entity, and that the same is true for the portion to be inserted; finally there is a technical condition that prevents overlapping parts. The lemma states symmetrically a similar condition for codomain containment.

Lemma 2. *Let R be a relation, and assume* $\mathsf{Entity}(P, E)$. *Then these implications hold*:

1.

$$\frac{\begin{array}{c}\mathsf{DomSub}(E, R)\ \ \mathsf{Entity}(P, E)\\ \mathsf{DomSub}(\delta^+E, \delta^+R)\ \ \mathsf{Entity}(P, \delta^+E)\\ (R \cap \mathbb{1};P \cap \delta^+R;\mathbb{1});\mathbb{1} \cap (E \cup \delta^+E);\mathbb{1} = \emptyset\end{array}}{\mathsf{DomSub}(E \cup \delta^+E, [R, \delta^+R]_P \cup \delta^+R)},$$

2.

$$\frac{\begin{array}{c}\mathsf{ImgSub}(F, R)\ \ \mathsf{Entity}(P, F)\\ \mathsf{ImgSub}(\delta^+F, \delta^+R)\ \ \mathsf{Entity}(P, \delta^+F)\\ \mathbb{1};(R \cap P;\mathbb{1} \cap \mathbb{1};\delta^+R) \cap \mathbb{1};(F \cup \delta^+F) = \emptyset\end{array}}{\mathsf{ImgSub}(F \cup \delta^+F, [R, \delta^+R]_P \cup \delta^+R)}.$$

Proof. Clearly,

$$(E \cup \delta^+E);\mathbb{1} \subseteq (R \cup \delta^+R);\mathbb{1},$$

and

$$(R \cup \delta^+ R);\mathbb{1} = ([R, \delta^+ R]_P \cup \delta^+ R);\mathbb{1} \cup A,$$

where

$$A := (R \cap \mathbb{1};P \cap \delta^+ R;\mathbb{1});\mathbb{1} \cup (R \cap P;\mathbb{1} \cap \mathbb{1};\delta^+ R);\mathbb{1}.$$

Now

$$\begin{aligned}(E \cup \delta^+ E);\mathbb{1} \cap A &= (E \cup \delta^+ E);\mathbb{1} \cap (R \cap P;\mathbb{1} \cap \mathbb{1};\delta^+ R);\mathbb{1}\\ &\subseteq (E \cup \delta^+ E);\mathbb{1} \cap P;\mathbb{1}\\ &= \emptyset.\end{aligned}$$

This establishes 1. In order to prove 2,

$$\mathbb{1};(R \cup \delta^+ R)$$

is decomposed similarly into

$$\mathbb{1};([R, \delta^+ R]_P \cup \delta^+ R)$$

and a part that is shown to be disjoint from $F \cup \delta^+ F$. □

In a similar way, we can make sure that the new relation maintains its properties as a single-valued map. The condition states that the part which is inserted represents a map, and that there exists no interference between the old, and the new part that might destroy the properties of a map. This will be formulated and proved now, together with the corresponding property for the inverse of a map, which will also be used later.

Lemma 3. *Let R be a relation, then the following implications hold*:

1.

$$\frac{\begin{array}{c}\mathsf{LUniq}(R)\ \mathsf{LUniq}(\delta^+ R)\\ [R, \delta^+ R] \subseteq \overline{\overline{\iota};\delta^+ R}\\ \delta^+ R \subseteq \overline{\overline{\iota};[R, \delta^+ R]}\end{array}}{\mathsf{LUniq}([R, \delta^+ R] \cup \delta^+ R)},$$

2.

$$\frac{\begin{array}{c}\mathsf{RUniq}(R)\ \mathsf{RUniq}(\delta^+ R)\\ [R, \delta^+ R] \subseteq \overline{\delta^+ R;\overline{\iota}}\\ \delta^+ R \subseteq \overline{[R, \delta^+ R];\overline{\iota}}\end{array}}{\mathsf{RUniq}([R, \delta^+ R] \cup \delta^+ R)}.$$

Proof. From $\cup$-distributivity it is inferred that

$$([R, \delta^+ R] \cup \delta^+ R);([R, \delta^+ R] \cup \delta^+ R)^{\smile} \subseteq \iota,$$

since Schröder's Rule implies

$$[R,\delta^+R];(\delta^+R)^\smile \subseteq \iota,$$
$$\delta^+R;([R,\delta^+R])^\smile \subseteq \iota.$$

This establishes property 1. The other property is proved similarly. □

It may be noted that both implications above can be reversed.

Inserting new elements may have an unfortunate effect on placeholders, as we have discussed in Section 5.3. Hence we have to make sure—among others—that in the insertion's result no placeholder appears twice as the first or the second component of a relation, and that no pair in a relation has placeholders on both sides. These properties are formulated now; for completeness, we state also conditions under which inheritance is preserved under insertions.

Use in what follows as abbreviations

$$\Gamma(A,B,C) \equiv_{\mathrm{Def}} A \cap (B \cap (C;\bar{\iota}));\mathbb{1} = \emptyset,$$
$$\Pi(A,B,C) \equiv_{\mathrm{Def}} A \cap (B;\mathbb{1} \cap C;\mathbb{1}) = \emptyset,$$
$$\Psi(A,B,C) \equiv_{\mathrm{Def}} A;B^\smile \cup B;A^\smile \cup B;B^\smile \subseteq C.$$

Lemma 4. *The following implications hold for any relation R*:

1.
$$\frac{\mathsf{NoTwice}(P,R)\ \ \mathsf{NoTwice}(P,\delta^+R,\delta^+R) \\ \Gamma(P,[R,\delta^+R]_P,\delta^+R)\ \ \Gamma(P,\delta^+R,[R,\delta^+R]_P)}{\mathsf{NoTwice}(P,[R,\delta^+R]_P \cup \delta^+R)},$$

2.
$$\frac{\mathsf{NoBoth}(P,R,S)\ \ \mathsf{NoBoth}(P,\delta^+R,\delta^+S) \\ \Pi(P,\delta^+R,[S,\delta^+S]_P)\ \ \Pi(P,[R,\delta^+R]_P,\delta^+S)}{\mathsf{NoBoth}(P,[R,\delta^+R]_P \cup \delta^+R,[S,\delta^+S]_P \cup \delta^+S)},$$

3.
$$\frac{\mathsf{NoSamePair}(P,R)\ \ \mathsf{NoSamePair}(P,\delta^+R)}{\mathsf{NoSamePair}(P,[R,\delta^+R]_P \cup \delta^+R)},$$

4.
$$\frac{\mathsf{NoDoubleFirst}(P,R)\ \ \mathsf{NoDoubleFirst}(P,\delta^+R) \\ \delta^+R;[R,\delta^+R]_P^\smile \subseteq \overline{P;\mathbb{1} \cap \mathbb{1};\overline{P}}}{\mathsf{NoDoubleFirst}(P,[R,\delta^+R]_P \cup \delta^+R)},$$

5.
$$\frac{\mathsf{NoDoubleSecond}(P,R)\ \ \mathsf{NoDoubleSecond}(P,\delta^+R) \\ [R,\delta^+R]_P^\smile;\delta^+R \subseteq \overline{\mathbb{1};P \cap \overline{P};\mathbb{1}}}{\mathsf{NoDoubleSecond}(P,[R,\delta^+R]_P \cup \delta^+R)},$$

6.

$$\frac{\mathsf{Inherits}(P,E,F)\ \ \mathsf{Entity}(P,\delta^+E)\ \ \mathsf{Entity}(P,\delta^+F)\ \ \delta^+E \subseteq F \cup \delta^+F}{\mathsf{Inherits}(P,E\cup\delta^+E,F\cup\delta^+F)}.$$

Proof. The proofs depend on the algebraic laws imposed for a relational algebra. We give prototypical examples for proving these implications.

Regarding 1, $\cup$-distributivity implies

$$\begin{aligned} P\cap(((X\cup Y)\cap(X\cup Y);\bar{\imath});\mathbb{1})) = {} & (P\cap((X\cap X;\bar{\imath});\mathbb{1}))) \\ & \cup(P\cap((X\cap Y;\bar{\imath});\mathbb{1}))) \\ & \cup(P\cap((Y\cap X;\bar{\imath});\mathbb{1}))) \\ & \cup(P\cap((Y\cap Y;\bar{\imath});\mathbb{1}))). \end{aligned}$$

Matching this against the definition, and against Γ yields the result. In a similar way 2 is established. The distributive law for $\cup$ implies 3 directly.

For establishing 4, put $\tau_\ell := P;\mathbb{1}\cap\mathbb{1};\overline{P}$ as an abbreviation, then the condition together with Schröder's Rule yields

$$\delta^+R;[R,\delta^+R]_P^{\smile}\cap\tau_\ell = \emptyset.$$

Consequently, by $\cup$-distributivity,

$$[R,\delta^+R]_P;\delta^+R^{\smile}\cap\tau_\ell = \emptyset$$

needs to be established. Schröder's Rule again shows this to be equivalent to

$$\tau_\ell;\delta^+R \subseteq \overline{[R,\delta^+R]_P},$$

which in turn may be seen from

$$\begin{aligned} \tau_\ell;\delta^+R & \subseteq P;\mathbb{1};\delta^+R\cap\mathbb{1};\overline{P};\delta^+R \\ & \subseteq P;\mathbb{1};\mathbb{1}\cap\mathbb{1};\mathbb{1};\delta^+R \\ & = P;\mathbb{1}\cap\mathbb{1};\delta^+R. \end{aligned}$$

The inference 5 is established in a very similar way. Finally, 6 is obvious. □

This lemma concludes the technical preparations for a characterization of the conditions under which weak validity is preserved under insertions.

We need to perform some index calculations on map letters. Define the set $\mathsf{Related}(t)$ as the smallest subset K of $\{T+1,\dots,T+S+B\}$ with these properties:

- $t \in K$,
- if $u \in K$ and $\langle u, v\rangle \in \mathsf{Up}$, then $v \in K$,
- if $\mathsf{r}_i \bullet\!\!-\!\!- \mathsf{r}_j$ or $\mathsf{r}_j -\!\!-\!\!\bullet\, \mathsf{r}_i$, then $i \in K$ iff $j \in K$.

Thus if we want to insert something into, say, entity E, and E corresponds to map letter r_i, then $\mathsf{Related}(i)$ contains the indices of exactly those entities and relations which are affected by this insertion.

Now let an entity or a relation correspond to map letter r_t. An insertion or a deletion is called *local at t* iff $\mathsf{r}_s = \emptyset$ whenever $s \in \{T+\Sigma,\dots,T+2\cdot\Sigma+1\}\setminus\mathsf{Related}(t)$. Introducing this guard prevents the insertion or the deletion from violating the invariants for the model by letting properties creeping in that are not really controlled through our safety measures.

From the instance $\mathscr{M}$ a new instance $\mathscr{M}'$ is generated by performing the insertions. Put for each $j \in \{1,\dots,S\}$

$$\mathsf{r}_{T+2\cdot\Sigma+j} := \mathsf{r}_{T+j} \cup \mathsf{r}_{T+\Sigma+j}.$$

This accounts for insertions into entities. As far as relations are concerned, we set for each index $j \in \{S+1,\dots,B\}$

$$\mathsf{r}_{T+2\cdot\Sigma+j} := [\mathsf{r}_{T+j}, \mathsf{r}_{T+\Sigma+j}] \cup \mathsf{r}_{T+\Sigma+j},$$

accounting for the peculiar way we insert into a relation.

Upon shifting the base address from T to $T+2\cdot\Sigma$, the weak validity of $\mathscr{M}'$ can be investigated:

Proposition 1. *Let $\mathscr{M}$ be a weakly valid ER model, assume that an insertion is local at some index t, then the ER model arising from the insertions is weakly valid, provided that the following conditions are all satisfied*:

1. $\&_{s\in\{1,\dots,S\}}\ \mathsf{Entity}(\pi, \mathsf{r}_{T+\Sigma+s})$,

2. $$\begin{aligned}&\&_{s\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\\ &\qquad \mathsf{DomSub}(\mathsf{r}_{\Sigma+\pi_1(\mathsf{Track}(s))}, \mathsf{r}_{\Sigma+s}) \,\&\, \mathsf{Entity}(\mathsf{r}_{\Sigma+\pi_1(\mathsf{Track}(s))})\\ &\qquad\quad \&\, \mathsf{ImgSub}(\mathsf{r}_{\Sigma+\pi_2(\mathsf{Track}(s))}, \mathsf{r}_{\Sigma+s}) \,\&\, \mathsf{Entity}(\mathsf{r}_{\Sigma+\pi_2(\mathsf{Track}(s))})\\ &\qquad\qquad \&\ (\mathsf{r}_s \cap \mathbb{1};\pi \cap \mathsf{r}_{\Sigma+s};\mathbb{1});\mathbb{1} \cap \mathsf{r}_{2\cdot\Sigma+\pi_1(\mathsf{Track}(s))};\mathbb{1} = \emptyset\\ &\qquad\qquad\quad \&\ \mathbb{1};(\mathsf{r}_s \cap \pi;\mathbb{1} \cap \mathbb{1};\mathsf{r}_{\Sigma+s}) \cap \mathbb{1};\mathsf{r}_{2\cdot\Sigma+\pi_1(\mathsf{Track}(s))} = \emptyset,\end{aligned}$$

3. $\&_{s\in\mathsf{LeftOne}\,\cap\,\mathsf{Related}(t)}\ \mathsf{r}_{\Sigma+s};\mathsf{r}^{\smile}_{\Sigma+s} \subseteq \boldsymbol{\iota} \,\&\, [\mathsf{r}_s, \mathsf{r}_{\Sigma+s}] \subseteq \overline{\overline{\boldsymbol{\iota}};\mathsf{r}_{\Sigma+s}} \,\&\, \mathsf{r}_{\Sigma+s} \subseteq \overline{\overline{\boldsymbol{\iota}};[\mathsf{r}_s, \mathsf{r}_{\Sigma+s}]}$,

4. $\&_{s\in\mathsf{RightOne}\cap\mathsf{Related}(t)}\ \mathsf{r}^{\smile}_{\Sigma+s};\mathsf{r}_{\Sigma+s} \subseteq \boldsymbol{\iota} \,\&\, [\mathsf{r}_s, \mathsf{r}_{\Sigma+s}] \subseteq \overline{\mathsf{r}_{\Sigma+s};\overline{\boldsymbol{\iota}}} \,\&\, \mathsf{r}_{\Sigma+s} \subseteq \overline{[\mathsf{r}_s, \mathsf{r}_{\Sigma+s}];\overline{\boldsymbol{\iota}}}$,

5. $$\begin{aligned}&\&_{s\in\mathsf{Related}(t)\cap\{T+S+1,\dots,T+S+B\}}\\ &\qquad \pi \cap (\mathsf{r}_{\Sigma+s} \cap \mathsf{r}_{\Sigma+s};\overline{\boldsymbol{\iota}};\mathbb{1}) = \emptyset \,\&\, \pi \cap (\mathsf{r}^{\smile}_{\Sigma+s} \cap (\mathsf{r}^{\smile}_{\Sigma+s};\overline{\boldsymbol{\iota}});\mathbb{1}) = \emptyset\\ &\qquad\quad \&\, \Gamma(\pi, [\mathsf{r}_s, \mathsf{r}_{\Sigma+s}], \mathsf{r}_{\Sigma+s}) \,\&\, \Gamma(\pi, [\mathsf{r}^{\smile}_s, \mathsf{r}^{\smile}_{\Sigma+s}], \mathsf{r}^{\smile}_{\Sigma+s})\\ &\qquad\qquad \&\, \Gamma(\pi, \mathsf{r}_{\Sigma+s}, [\mathsf{r}_s, \mathsf{r}_{\Sigma+s}]) \,\&\, \Gamma(\pi, \mathsf{r}^{\smile}_{\Sigma+s}, [\mathsf{r}^{\smile}_s, \mathsf{r}^{\smile}_{\Sigma+s}]),\end{aligned}$$

Table 4

Item #	Property addressed
1	Entities are preserved
2	Domain properties are preserved
3	Left uniqueness
4	Right uniqueness
5	NoTwice
6	NoBoth
7	NoSamePair
8	NoDoubleFirst
9	NoDoubleSecond
10	Inheritance is preserved

6.
$$\&_{s\in\mathsf{Related}(t)\cap\{T+S+1,\dots,T+S+B\}}\ \&_{v\in\mathsf{Related}(t)\cap\{s,\dots,T+S+B\}}\ \&\ \pi\cap(\mathsf{r}_{\Sigma+s};\mathbb{1})\cap(\mathsf{r}_{\Sigma+v};\mathbb{1})=\emptyset\ \ \Pi(\pi,\mathsf{r}_{\Sigma+s},[\mathsf{r}_v,\mathsf{r}_{\Sigma+v}])\ \&\ \Pi(\pi,[\mathsf{r}_s,\mathsf{r}_{\Sigma+s}],\mathsf{r}_v),$$

7. $\&_{s\in\mathsf{Related}(t)\cap\{T+S+1,\dots,T+S+B\}}\ \mathsf{r}_{\Sigma+s}\cap\pi;\mathbb{1}\cap\mathbb{1};\pi=\emptyset,$

8.
$$\&_{s\in\mathsf{Related}(t)\cap\{T+S+1,\dots,T+S+B\}}\ \mathsf{r}_{\Sigma+s};\breve{\mathsf{r}}_{\Sigma+s}\cap\pi;\mathbb{1}\cap\mathbb{1};\overline{\pi}=\emptyset\ \ \&\ \mathsf{r}_{\Sigma+s};[\mathsf{r}_s,\mathsf{r}_{\Sigma+s}]^{\smile}\subseteq\overline{\pi;\mathbb{1}\cap\mathbb{1};\overline{\pi}},$$

9.
$$\&_{s\in\mathsf{Related}(t)\cap\{T+S+1,\dots,T+S+B\}}\ \breve{\mathsf{r}}_{\Sigma+s};\mathsf{r}_{\Sigma+s}\cap\mathbb{1};\pi\cap\overline{\pi};\mathbb{1}=\emptyset\ \ \&\ [\mathsf{r}_s,\mathsf{r}_{\Sigma+s}]^{\smile};\mathsf{r}_{\Sigma+s}\subseteq\overline{\mathbb{1};\pi\cap\overline{\pi};\mathbb{1}},$$

10. $\&_{\langle i,j\rangle\in\mathsf{Up}\cap\mathsf{Related}(T)\times\mathsf{Related}(T)}\ \mathsf{Entity}(\pi,\mathsf{r}_{\Sigma+i})\ \&\ \mathsf{Entity}(\pi,\mathsf{r}_{\Sigma+j})\ \&\ \mathsf{r}_{\Sigma+i}\subseteq\mathsf{r}_j\cup\mathsf{r}_{\Sigma+j}.$

Proof. 0. This looks at first like a confusing bag of details. So let us sort them out by providing a table which permits stating a correspondence between the properties stated in the proposition, and the properties of an ER model (Table 4):

1. Property 1 establishes together with the assumption

$$\&_{T+1\leqslant j\leqslant T+S}\ \mathsf{Entity}(\mathsf{r}_j)$$

that

$$\&_{T+2\cdot\Sigma+1\leqslant j\leqslant T+2\cdot\Sigma+S}\ \mathsf{Entity}(\mathsf{r}_j)$$

is true.

2. Take $s \in \mathsf{Related}(t)$, and assume that $\langle i, j\rangle = \mathsf{Track}(s)$. Because $\mathscr{M}$ is weakly valid, we know that $\mathsf{DotDot}(\mathsf{r}_i, \mathsf{r}_s, \mathsf{r}_j)$ holds. In particular,

$$\mathsf{DomSub}(\mathsf{r}_i, \mathsf{r}_s) \ \& \ \mathsf{Entity}(\mathsf{r}_i)$$

are true. This implies together with

$$\mathsf{DomSub}(\mathsf{r}_{\Sigma+i}, \mathsf{r}_{\Sigma+s}) \ \& \ \mathsf{Entity}(\mathsf{r}_{\Sigma+i}) \ \& \ (\mathsf{r}_s \cap \mathbb{1};\pi \cap \mathsf{r}_{\Sigma+s};\mathbb{1});\mathbb{1} \cap \mathsf{r}_{2\cdot\Sigma+i};\mathbb{1} = \emptyset$$

through Lemma 2 (Property 1) that $\mathsf{DomSub}(\mathsf{r}_{2\cdot\Sigma+i}, \mathsf{r}_{2\cdot\Sigma+s})$ holds. In a similar way (by appealing to Lemma 2, part 2), $S\mathsf{ImgSub}(\mathsf{r}_{\Sigma+j}, \mathsf{r}_s)$ is established. Collecting things, we have established that

$$\mathsf{DotDot}(\mathsf{r}_{2\cdot\Sigma+i}, \mathsf{r}_{2\cdot\Sigma+s}, \mathsf{r}_{2\cdot\Sigma+j})$$

is true.

3. Let $s \in \mathsf{LeftOne} \cap \mathsf{Related}(t)$, then we know from $\mathscr{M}'s$ validity that $\mathsf{r}_s;\mathsf{r}_s \subseteq \imath$ holds. From Lemma 3, Property 1 we now see that

$$\mathsf{LUniq}(\mathsf{r}_{2\cdot\Sigma+s})$$

is true, provided 3 holds. In a very similar manner,

$$\mathsf{RUniq}(\mathsf{r}_{2\cdot\Sigma+s})$$

is deduced from 4 for $s \in \mathsf{RightOne} \cap \mathsf{Related}(t)$.

4. From 5 we infer that

$$\&_{T+2\cdot\Sigma+S+1 \leqslant j \leqslant T+2\cdot\Sigma+S+B} \ \mathsf{NoTwice}(\pi, \mathsf{r}_j) \ \& \ \mathsf{NoTwice}(\pi, \mathsf{r}_j^{\smile})$$

holds (where we use Lemma 1 to establish that the identity $[\mathsf{r}_{T+j}, \mathsf{r}_{T+\Sigma+j}]^{\smile} = [\mathsf{r}_{T+j}^{\smile}, \mathsf{r}_{T+\Sigma+j}^{\smile}]$ holds.

5. In similar ways one establishes the desired properties, resorting to Lemma 4 for establishing the necessary conditions. $\square$

To help the reader appreciate the content of the conditions above, we interpret the second and the last of them. Interpretations for the other conditions are quite similar and left to the reader. The first conditions states conditions under which

$$E \cup \delta^+ E \bullet\!\!- R \cup \delta^+ R -\!\!\bullet F \cup \delta^+ F$$

holds, i.e., under which conditions $E \cup \delta^+ E$ and $F \cup \delta^+ F$ remain the tight domain and the tight codomain, resp., of $[R, \delta^+ R] \cup \delta^+ R$, provided E was the tight domain, and F was the tight codomain of R before the insertion, i.e., provided

$$E \bullet\!\!- R -\!\!\bullet F$$

holds. The conditions state that $\delta^+ E$ needs to be an entity such that

$$\mathsf{dom}(\delta^+ E) \subseteq \mathsf{dom}(\delta^+ R)$$

is true, hence each element to be inserted into E should be the first component of a pair to be inserted into R. In the same way $\delta^+ F$ is required to be an entity such that

$$\mathsf{img}(\delta^+ F) \subseteq \mathsf{img}(\delta^+ R)$$

holds. In addition we make sure that the required conditions on place holders are not violated, so that

$$(R \cap \mathbb{1};P \cap \delta^+ R;\mathbb{1});\mathbb{1} \cap (E \cup \delta^+ E);\mathbb{1} = \emptyset.$$
$$\mathbb{1};(R \cap P;\mathbb{1} \cap \mathbb{1};\delta^+ R) \cap \mathbb{1};(F \cup \delta^+ F) = \emptyset$$

holds, as we have discussed above.

The last condition simply states that for $E \cup \delta^+ E$ to inherit from $R \cup \delta^+ R$ it is sufficient that E inherits from F, and that $\delta^+ E$ is a subset of $F \cup \delta^+ F$, and that the new sets are entities indeed.

8.3. Looking at attributes

We did neglect attributes for greater ease of discussion; now is the place for introducing conditions on them. Attributes are defined on entities (this is one of our restrictions, cf. Section 3.1), they come in different flavors, as we will discuss now. An attribute α on entity E is a partial map, so $\mathsf{RUniq}(\alpha)$ should be satisfied, and its domain should be contained in (the domain of) E, thus

$$\mathsf{dom}(\alpha) \subseteq \mathsf{dom}(E)$$

should hold. Moreover we assume attributes to have atomic values.

This requirement will be modelled as follows: We assume our universe $\mathscr{U}$ to be structured as

$$\mathscr{U} = \mathscr{A} \cup \mathscr{A}^*,$$

where $\mathscr{A} \neq \emptyset$ are the atomic values, and $\mathscr{A}^*$ denotes the set of all words over the alphabet $\mathscr{A}$, hence

$$\mathscr{A} \cap \mathscr{A}^* = \emptyset$$

with ε as the empty word; as usual, we put

$$\mathscr{A}^+ := \mathscr{A}^* \backslash \{\varepsilon\}.$$

We reserve a map letter $\varepsilon \in \{\mathsf{r}_1, \ldots, \mathsf{r}_T\}$ for representing ε (hence $\mathsf{Snglt}(\varepsilon)$ & $\mathsf{Coll}(\varepsilon)$) and permit only interpretations $\mathscr{I}$ that satisfy $\varepsilon^{\mathscr{I}} = \{\langle \varepsilon, \varepsilon \rangle\}$. The atomic entities in $\mathscr{A}$ are modelled through the map letter υ with

$$\mathsf{Coll}(\upsilon) \ \& \ \mathsf{NonVoid}(\upsilon) \ \& \ \upsilon \cap \varepsilon = \emptyset.$$

In addition we postulate that $\pi \subseteq \upsilon$ holds.

Interpretations are restricted further by postulating that

$$\upsilon^{\mathscr{I}} = \{\langle a,b\rangle \in \mathscr{A}^2 \mid a = b\}.$$

Moreover we assume the existence of canonic projections CAR and CDR separating the head from the tail of a non-empty word, hence

$$\mathsf{CAR} : \begin{cases} \mathscr{A}^+ & \to \mathscr{A}, \\ t_1 \dots t_k & \mapsto t_1 \end{cases}$$

and

$$\mathsf{CDR} : \begin{cases} \mathscr{A}^+ & \to \mathscr{A}, \\ t_1 \dots t_k & \mapsto t_2 \dots t_k. \end{cases}$$

These projections are represented through the map letters λ and ρ, corresponding to CAR and CDR, resp; their properties are discussed in Section 6.2. We abbreviate for later use the ith projection (hence the operation of extracting the ith component of a tuple) by

$$Z^{(i)} \equiv_{\mathrm{Def}} (i = 1 \ ?\ \mathsf{CAR}:\ Z^{(i-1)};\mathsf{CDR}),$$
$$\tau^{(i)} \equiv_{\mathrm{Def}} (i = 1 \ ?\ \lambda:\ \tau^{(i-1)};\rho),$$

the latter abbreviation preparing for the use of map letters later on.

Returning to attributes: a mandatory attribute α on entity E is characterized through

$$\mathsf{dom}(\alpha) = \mathsf{dom}(E)\ \&\ \mathbb{1};\pi \cap \mathsf{img}(\alpha) = \emptyset.$$

If $\{\alpha_0, \dots, \alpha_w\}$ is a collection of key attributes on E, then Section 4.2 shows that this property means

$$\mathsf{RUniq}\left(\bigcap_{i=0}^{w} Z^{(i+1)};\alpha_i^{\smile}\right)$$

to hold.

We proceed now in the same manner as in Section 8.2 by first formulating conditions which govern the preservation of the relevant properties, and begin with the property that being a map is preserved. The second part of the following Lemma reflects the postulate that insertions must preserve the domain: inserting into an attribute and inserting into its domain may not lead to the situation that the updated attribute has no longer the update of the domain as its domain.

Lemma 5. *The following properties hold*:

1. $$\frac{\mathsf{RUniq}(\alpha)\Psi(\alpha, \delta^+\alpha, \boldsymbol{\imath})}{\mathsf{RUniq}(\alpha \cup \delta^+\alpha)},$$

2. $$\frac{\begin{array}{c}\mathsf{dom}(\alpha) = \mathsf{dom}(E) \\ (\delta^+\alpha \backslash \alpha);\mathbb{1} = (\delta^+E;\mathbb{1})\backslash(E;\mathbb{1})\end{array}}{\mathsf{dom}(\alpha \cup \delta^+\alpha) = \mathsf{dom}(E \cup \delta^+E)}.$$

Proof. Both parts follows directly from $\cup$-distributivity. Note that the implication in the first part can be reversed. □

The conditions laid down in Lemma 5 permit stating conditions under which some attribute conditions persist under insertion. The exception is a condition which permits being a member of a family of key attributes stable under insertions. The criterion is formulated in Lemma 6. It requires some preparations.

Remember that in a relation algebra the equality

$$\bigcap_{i\in I}(A_{1,i}\cup A_{2,i}) = \bigcup_{J\subseteq I}\left(\bigcap_{j\in J}A_{1,j}\cap\bigcap_{j\notin J}A_{2,j}\right)$$

holds, whenever I is finite. This is so because a relation algebra is a Boolean algebra, in particular a distributive lattice. Abbreviate for the map expressions $A_0,\dots,A_k,B_0,\dots,B_k$ and for $J,K\subseteq\{0,\dots,k\}$

$$\begin{aligned}\Lambda(J,\langle A_0,\dots,A_k\rangle,\langle B_0,\dots,B_k\rangle) &\equiv_{\text{Def}} \bigcap_{j\in J}A_j;(Z^{(j+1)})^{\smile}\cap\bigcap_{j\notin J}(B_j\backslash A_j);(Z^{(j+1)})^{\smile},\\ \Gamma(J,K,\langle A_0,\dots,A_k\rangle,\langle B_0,\dots,B_k\rangle) &\equiv_{\text{Def}} \Lambda(J,\langle A_0,\dots,A_k\rangle,\langle B_0,\dots,B_k\rangle)\\ &\quad ;\ \Lambda(K,\langle A_0,\dots,A_k\rangle,\langle B_0,\dots,B_k\rangle)^{\smile}.\end{aligned}$$

With these notations we may formulate:

Lemma 6. *Invariance of a key under insertion is maintained by the following condition*:

$$\frac{\begin{array}{c}\&_{i=0}^{k}\,\mathsf{RUniq}(\alpha_i)\\ \&_{i=0}^{k}\,\mathsf{RUniq}(\delta^{+}\alpha_i)\\ \&_{J\subseteq\{0,\dots,k\}}\&_{K\subseteq\{0,\dots,k\}}\ \Gamma(J,K,\langle\alpha_0,\dots,\alpha_k\rangle,\langle\delta^{+}\alpha_0,\dots,\delta^{+}\alpha_k\rangle)\subseteq\boldsymbol{\iota}\end{array}}{\mathsf{RUniq}\left(\bigcap_{i=0}^{k}Z^{(i+1)};(\alpha_i\cup\delta^{+}\alpha_i)^{\smile}\right)}.$$

It should be noted that the formulation above requires

$$\langle\alpha_0,\dots,\alpha_k\rangle$$

as well as

$$\langle\delta^{+}\alpha_0\backslash\alpha_0,\dots,\delta^{+}\alpha_k\backslash\alpha_k\rangle$$

to have the properties of key attributes.

Proof. The distributive law (in the lattice), $\cup$-distributivity (with respect to composition), and Lemma 1 together show that

$$\left(\bigcap_{i=0}^{k}Z^{(i+1)};\ (\alpha_i\cup\delta^{+}\alpha_i)^{\smile}\right)^{\smile};\left(\bigcap_{i=0}^{k}Z^{(i+1)};\ (\alpha_i\cup\delta^{+}\alpha_i)^{\smile}\right)$$

equals

$$\bigcup_{J\subseteq\{0,\dots,k\}}\ \bigcup_{K\subseteq\{0,\dots,k\}} \Gamma(J,K,\langle\alpha_0,\dots,\alpha_k\rangle,\langle\delta^+\alpha_0,\dots,\delta^+\alpha_k\rangle).$$

This implies the desired result. □

The condition just formulated is exponential in the size of the key, consequently, it is not very convenient for practical purposes. On the other hand, it is exact, because a key can be extended if and only if the condition above is satisfied. It would be desirable to develop a more practical, if only sufficient condition for the invariance under insertions of the property being a key.

Now call an ER model $\mathscr{M}$ valid iff it is weakly valid, and if the conditions on attributes that have been laid down in the model's declaration are satisfied. Formally:

Definition 5. The ER model $\mathscr{M}$ is called valid iff
1. $\mathscr{M}$ is weakly valid,
2. the attributes satisfy

$$\&_{T+1\leqslant i\leqslant T+S}\ \&_{j\in\mathsf{Attributes}(i)}\ \mathsf{RUniq}(\mathsf{r}_j)\ \&\ \mathsf{dom}(\mathsf{r}_j)\subseteq\mathsf{dom}(\mathsf{r}_i)\ \&\ \mathsf{img}(\mathsf{r}_j)\subseteq\upsilon,$$
$$\&$$
$$\&_{T+1\leqslant i\leqslant T+S}\ \&_{j\in\mathsf{Mandatory}(i)}\ \mathsf{dom}(\mathsf{r}_j)=\mathsf{dom}(\mathsf{r}_i)\ \&\ \mathbb{1};\pi\cap\mathsf{img}(\mathsf{r}_j)=\emptyset,$$
$$\&$$
$$\&_{T+1\leqslant i\leqslant T+S}\ \textbf{\textit{let}}\ \{i_1,\dots,i_j\}=\mathsf{Key}(i)\ \textbf{\textit{in}}\ \mathsf{RUniq}\left(\bigcap_{\ell=1}^{j} Z^{(\ell+1)};\mathsf{r}_{i_\ell}^{\smile}\right).$$

We will state now conditions under which the attributes of the changes ER model $\mathscr{M}'$ will cater for the model's validity after the construction process is extended to attributes in the obvious way. Investigating validity requires us to exploit properties of the change sets $\delta^+\cdot$ for attributes in the context of their relations to the change sets for entities (note that we do for the time being without attributes on the relations on $\mathscr{M}$).

Proposition 2. *Suppose that the ER model $\mathscr{M}$ is valid, and that in addition to Properties* 1–10 *from Proposition* 1 *the following properties are satisfied, when performing an insertion that is local at some index t:*

1.
$$\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\&_{j\in\mathsf{Attributes}(i)}\ \mathsf{r}_{\Sigma+j}^{\smile};\mathsf{r}_{\Sigma+j}\subseteq\iota\ \&\ \Psi(\mathsf{r}_j,\mathsf{r}_{\Sigma+j},\iota)\ \&\ \mathbb{1};\mathsf{r}_{\Sigma+j}\subseteq\upsilon,$$
2.
$$\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\&_{j\in\mathsf{Mandatory}(i)}\ (\mathsf{r}_{\Sigma+j}\backslash\mathsf{r}_j);\mathbb{1}=(\mathsf{r}_{\Sigma+i};\mathbb{1})\backslash(\mathsf{r}_i;\mathbb{1}),$$
3.
$$\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\&_{j\in\mathsf{Mandatory}(i)}\ \mathbb{1};\pi\cap\mathbb{1};\mathsf{r}_{\Sigma+j}=\emptyset,$$
4.
$$\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}$$
$$\textbf{\textit{let}}\ \mathsf{Key}(i)=\{i_0,\dots,i_k\}\ \textbf{\textit{in}}$$
$$\&_{J\subseteq\{0,\dots,k\}}\&_{K\subseteq\{0,\dots,k\}}\ \Gamma(J,K,\langle\mathsf{r}_{i_0},\dots,\mathsf{r}_{i_k}\rangle,\langle\mathsf{r}_{\Sigma+i_0},\dots,\mathsf{r}_{\Sigma+i_k}\rangle)\subseteq\iota.$$

Then $\mathscr{M}'$ is a valid ER model.

Proof. Lemma 5 makes sure that condition 1 implies that we indeed obtain attributes, and that by condition 2 mandatory attributes remain mandatory. Condition 3 caters for banning place holders from the image of mandatory attributes. The last condition helps together with Lemma 6 in ascertaining the properties of keys in the new model. □

9. Deletions: validity

Deletions are treated in a similar fashion: we formulate conditions under which deletions maintain the validity of the ER model. We will again deal initially with entities and relations only, and then in a second step extend our considerations to attributes. This application of the principle of *Separation of Concerns* will again first formulate conditions under which weak validity is preserved, and then upgrade these conditions with the goal of finding criteria for unconstrained validity.

We will use the same initial setup of map letters as in Section 7, but now interpret the map letters between $T + \Sigma + 1$ and $T + 2 \cdot \Sigma$ as the place where we store the values to be deleted; they are now prefixed with δ^-. If entity E corresponds to map letter r_{T+i} with $\delta^- E$ corresponding to $\mathsf{r}_{T+\Sigma+i}$, then $E \backslash \delta^- E$ will be deposited at $\mathsf{r}_{T+2\cdot\Sigma+i}$. In the same linear way, proceeding in a block wise fashion, we deposit the changed values for relations and attributes. The reader may wish to consult Fig. 4 again.

9.1. Weak validity

The following observation shows that for maintaining weak validity we need not consider place holders separately, that left or right uniqueness of relations is of no concern, and that the defining property of key attributes remains intact, when deleting elements from the maps constituting the key. In other words, the properties of interest are downward closed.

Lemma 7. *The following implications hold*:

1.

$$\frac{R_1 \subseteq R_2 \ \ \mathsf{LUniq}(R_2)}{\mathsf{LUniq}(R_1)},$$

2.

$$\frac{R_1 \subseteq R_2 \ \ \mathsf{RUniq}(R_2)}{\mathsf{RUniq}(R_1)},$$

3.

$$\frac{\begin{array}{c}\&_{i=1}^{k} R_{1,i} \subseteq R_{2,i} \\ \mathsf{PlaceHolder}(P, \{R_{2,1}, \ldots, R_{2,k}\})\end{array}}{\mathsf{PlaceHolder}(P, \{R_{1,1}, \ldots, R_{1,k}\})},$$

4.

$$\frac{\begin{array}{c}\&_{i=1}^{k} R_{1,i} \subseteq R_{2,i} \\ \mathsf{RUniq}\left(\bigcap_{i=1}^{n} T^{(i+1)};R_{2,i}^{\smile}\right)\end{array}}{\mathsf{RUniq}\left(\bigcap_{i=1}^{n} T^{(i+1)};R_{1,i}^{\smile}\right)}.$$

Proof. Because the composition operator **;** is monotone in both arguments, the first two assertions are immediate. The monotonicity of the converse operator (which sends R to $R^{\smile}$) is used on top of that in establishing the third assertion. This is done by inspecting the auxiliary macros that constitute the conjunction defining the PlaceHolder-macro, and that are formulated in Section 5.3. Monotonicity of both operations is finally used to establish the last implication. □

Thanks to Lemma 7, the technical base for maintaining weak validity in the following statement (which corresponds to Lemma 4 for insertions), is rather easier to formulate:

Lemma 8. *The following implications hold*:

1.

$$\frac{\begin{array}{c}\mathsf{DomSub}(E,R) \\ \mathsf{Entity}(E)\ \mathsf{Entity}(\delta^{-}E) \\ \mathsf{DomSub}(R,\delta^{-}E \cup R\backslash\delta^{-}R)\end{array}}{\mathsf{DomSub}(E\backslash\delta^{-}E, R\backslash\delta^{-}R)},$$

2.

$$\frac{\begin{array}{c}\mathsf{ImgSub}(F,R) \\ \mathsf{Entity}(F)\ \mathsf{Entity}(\delta^{-}F) \\ \mathsf{DomSub}(R^{\smile},\delta^{-}F^{\smile} \cup (R\backslash\delta^{-}R)^{\smile})\end{array}}{\mathsf{ImgSub}(F\backslash\delta^{-}F, R\backslash\delta^{-}R)}.$$

Proof. Only the first implication needs to be established, since the second follows by inversion. Since

$$R;\mathbb{1} \subseteq (\delta^{-}E \cup (R\backslash\delta^{-}R));\mathbb{1},$$

an elementary calculation establishes

$$(R;\mathbb{1})\backslash(\delta^{-}E;\mathbb{1}) \subseteq (R\backslash\delta^{-}R);\mathbb{1}.$$

Consequently,

$$\begin{aligned}(E\backslash\delta^{-}E);\mathbb{1} &= (E;\mathbb{1})\backslash(\delta^{-}E;\mathbb{1}) \\ &\subseteq (R;\mathbb{1})\backslash(\delta^{-}E;\mathbb{1})\ (\text{since } E;\mathbb{1} \subseteq R;\mathbb{1}) \\ &\subseteq (R\backslash\delta^{-}R);\mathbb{1}.\end{aligned}$$

This establishes the claim. □

From the instance $\mathscr{M}$ a new instance $\mathscr{M}'$ is generated by performing the deletions. This is very similar to the insertion discussed above: Put for each $j \in \{1, \dots, S+B\}$

$$\mathsf{r}_{T+2\cdot\Sigma+j} := \mathsf{r}_{T+j} \backslash \mathsf{r}_{T+\Sigma+j}.$$

Upon shifting the base address from T to $T+2\cdot\Sigma$, the weak validity of $\mathscr{M}'$ can be investigated:

Proposition 3. *Let $\mathscr{M}$ be a weakly valid ER model, assume that a deletion is local at some index t, then the ER model arising from the deletions is weakly valid, provided the following conditions are all satisfied:*

1.

$$\begin{aligned}&\&_{s\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\\ &\qquad \mathsf{Entity}(\mathsf{r}_{\Sigma+\pi_1(\mathsf{Track}(s))}) \,\&\, \mathsf{Entity}(\mathsf{r}_{\Sigma+\pi_2(\mathsf{Track}(s))})\\ &\qquad\quad \&\, \mathsf{DomSub}(\mathsf{r}_s, \mathsf{r}_{\Sigma+\pi_1(\mathsf{Track}(s))} \cup (\mathsf{r}_s \backslash \mathsf{r}_{\Sigma+s}))\\ &\qquad\qquad \&\, \mathsf{DomSub}(\mathsf{r}_s^{\smile}, \mathsf{r}_{\Sigma+\pi_2(\mathsf{Track}(s))}^{\smile} \cup (\mathsf{r}_s \backslash \mathsf{r}_{\Sigma+s})^{\smile}),\end{aligned}$$

2. $\&_{\langle i,j\rangle\in\mathsf{Up}\cap\mathsf{Related}(t)\times\mathsf{Related}(t)}\ \mathsf{r}_{\Sigma+j} \subseteq \mathsf{r}_{\Sigma+i}$.

Proof. Because Condition 2 takes care of inheritance, and because of Lemma 7 we have to establish only $\mathsf{DotDot}(\mathsf{r}_{2\cdot\Sigma+\pi_1(\mathsf{Track}(s))}, \mathsf{r}_{2\cdot\Sigma+s}, \mathsf{r}_{2\cdot\Sigma+\pi_2(\mathsf{Track}(s))})$ for all indices $s \in \mathsf{Related}(t)$. But this follows through a straightforward calculation from the assumption together with Lemma 8. □

9.2. Adding attributes

Turning to attributes, we see that the functional character of attributes together with that of their domains is maintained when changing to a subset of each:

Lemma 9. *Let α be an attribute on entity E, then the following implications show how to maintain attribute conditions under deletion:*

1.

$$\frac{\begin{array}{c}\mathsf{RUniq}(\alpha)\ \mathsf{RUniq}(\delta^-\alpha)\\ \mathsf{Entity}(E)\ \mathsf{dom}(\alpha) \subseteq \mathsf{dom}(E)\\ (\alpha\backslash\delta^-\alpha)^{\smile};\delta^-E = \emptyset\end{array}}{\mathsf{dom}(\alpha\backslash\delta^-\alpha) \subseteq \mathsf{dom}(E\backslash\delta^-E)},$$

2.

$$\frac{\begin{array}{c}\mathsf{RUniq}(\alpha)\ \mathsf{RUniq}(\delta^-\alpha)\\ \mathsf{Entity}(E)\ \mathsf{dom}(\alpha) \subseteq \mathsf{dom}(E)\\ (\alpha\backslash\delta^-\alpha)^{\smile};\delta^-E = \emptyset\\ \mathsf{dom}(\delta^-E) \subseteq \mathsf{dom}(\delta^-\alpha)\end{array}}{\mathsf{dom}(\alpha\backslash\delta^-\alpha) = \mathsf{dom}(E\backslash\delta^-E)}.$$

Proof. 1. Schröder's Cycle Rule implies that $(\alpha\backslash\delta^-\alpha)^\smile;\delta^-E=\emptyset$ is equivalent to $\mathsf{dom}(\alpha\backslash\delta^-\alpha)\cap\mathsf{dom}(\delta^-E)=\emptyset$, thus

$$\begin{aligned}\mathsf{dom}(\alpha\backslash\delta^-\alpha)&\subseteq\mathsf{dom}(E)\backslash\mathsf{dom}(\delta^-E)\\&=\mathsf{dom}(E\backslash\delta^-E).\end{aligned}$$

This proves 1.

2. It remains to show that the domain of $\alpha\backslash\delta^-\alpha$ contains $E\backslash\delta^-E$ under the conditions from 2: Using Lemma 1, we see

$$\begin{aligned}\mathsf{dom}(E\backslash\delta^-E)&=\mathsf{dom}(E)\backslash\mathsf{dom}(\delta^-E)\\&\subseteq\mathsf{dom}(\alpha)\backslash\mathsf{dom}(\delta^-\alpha)\\&\subseteq\mathsf{dom}(\alpha\backslash\delta^-\alpha).\qquad\square\end{aligned}$$

Now we are able to state conditions under which deletions from an ER model maintain its validity:

Proposition 4. *Suppose that the ER model $\mathscr{M}$ is valid, and that in addition to the properties* 1 *and* 2 *from Proposition* 3 *the following properties are satisfied, when performing an insertion that is local at some index t:*

1. $\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\ \&_{j\in\mathsf{Attributes}(i)}\ \mathsf{r}_{\Sigma+j}^\smile;\mathsf{r}_{\Sigma+j}\subseteq\iota\ \&\ (\mathsf{r}_j\backslash\mathsf{r}_{\Sigma+j})^\smile;\mathsf{r}_{\Sigma+i}=\emptyset.$
2. $\&_{i\in\mathsf{Related}(t)\cap\{T+1,\dots,T+B\}}\ \&_{j\in\mathsf{Mandatory}(i)}\ \mathsf{r}_{\Sigma+i};\mathbb{1}\subseteq\mathsf{r}_{\Sigma+j};\mathbb{1}.$

Proof. Since the weak validity of $\mathscr{M}'$ is already being taken care of by Proposition 3, we have to cater for the integrity of the attributes. Condition 1 maintains together with Lemma 9 the condition under which the property of being an attribute is preserved, the second condition states when a mandatory attribute remains one; this also makes use of the same lemma. It was noted already in Lemma 7 that key attributes are not sensitive to deletions. $\square$

It comes as a surprise that deletions are so much easier to handle, than insertions. Some data structures (like binary search trees or heaps, see e.g. [6]) are rather sensitive to deletions. Hence a deletion requires much there more attention in maintaining the invariant, than an insertion does (and consequently makes the analysis a much harder and more unpleasant undertaking). In the case of the data structures just mentioned, however, intrinsic properties of the keys do not enter the discussion, while in the case considered here the monotonicity together with the downward closeness of some properties played a simplifying role.

10. Implementation issues

The reader may now wonder to what extent the ideas presented in this paper have led, or can lead, to concrete implementations.

In its most classical version in which we have relied, relation calculus has a charming simplicity which makes it attractive not only as a theoretical grounding but also from a computational point of view. To become practically usable, however, relation calculus needs a platform of computerized methods, able to effect translations from higher-level formalisms into it, providing automated support to reasoning, supplying algebraic simplification techniques, and the like. Among various platforms which are being developed concurrently, let us mention the PROLOG-based Metamorpho system: the portion of it which is already publicly available[1] supports various definitional extension mechanisms, namely flexible means to extend the native syntax of relation calculus. Some of the shortening definitions in this paper (cf., e.g., the one of $\mathsf{PlaceHolder}(P, Q)$ introduced in Section 5.3) are elaborate enough that they turned out to be ideal test cases for Metamorpho's definitional apparatus.

A user can load into Metamorpho one or several texts specifying the *primitive operators* and *derived constructs* of his/her version of relation calculus: thereby, the system acquires the ability to expand derived constructs in terms of the primitive ones, through abbreviating definitions ('macros', in a sense, of the kind illustrated before Lemma 1) which have been supplied by means of these files.

The user can then load his/her selection of *logical axioms* for relation calculus from another file: these are laws of the kind shown in the fourth item of Definition 1. The rationale of this is: since many competing axiomatic systems exist for dyadic relation algebras, this essential component of the logical apparatus should be customizable, very much like the selection of basic language constructs. The current Metamorpho system does not possess any autonomous capabilities to carry out equational reasoning; hence, at least on a temporary basis, it will be interfaced with an existing theorem-prover which can assist—even with some degree of autonomy—the user who is performing (or simply re-checking) deductions of the kind we have highlighted several times while proving the lemmas of Sections 8 and 9.

In the next step, the Metamorpho user will load *proper axioms* of a theory based on relation calculus. Unlike logical axioms, these are specific of the intended application. The formal statements of the properties of $\lambda, \varrho, \upsilon, \varepsilon$, by which we have set up in Section 4.2 the theory of flat tuples, are a typical example of proper axioms: it will be a subtheory of the theory associated with an ER-model in the way explained in Section 6. Like developing a sophisticated computer program, constructing a theory can at times be an overly complicated task, unless the environment provides mechanisms which support modularity conveniently: to cater for this, Metamorpho offers so-called 'templates' acting like procedures in the construction of theories.

Fig. 5 shows an excerpt of the definition files which one submits to Metamorpho in preparation for the treatment of ER-models. We do not enter into a full explanation here, because most of the notation introduced by this table agrees with what has been discussed at length already, both in its constructs and in their definitions (save for syntactic details and, perhaps, for minor semantic refinements). We could prolong this

[1] See under the URL http://costantini.dm.univaq.it/online.htm.

$$
\begin{array}{c}
\mathsf{mult}(P) \;=:\; P\cap P\mathbin{;}\bar{\imath} \\[4pt]
\begin{array}{rcl@{\qquad}rcl}
\mathsf{lA}(P) & =: & \mathbb{1}\mathbin{;}P & \mathsf{rA}(P) & =: & P\mathbin{;}\mathbb{1} \\
\mathsf{RUniq}(P,Q) & \leftrightarrow\mathbf{:} & \mathsf{mult}(Q)\cap\mathsf{rA}(P)=\emptyset & \mathsf{LUniq}(P,Q) & \leftrightarrow\mathbf{:} & \mathsf{RUniq}(P,QS^{\smile}) \\
\mathsf{RXcl}(P,Q) & \leftrightarrow\mathbf{:} & \mathsf{mult}(Q)\cap\mathsf{lA}(P)=\emptyset & \mathsf{LXcl}(P,Q) & \leftrightarrow\mathbf{:} & \mathsf{RXcl}(P,Q^{\smile}) \\
\mathsf{th}(L,R\,\|\,1) & =: & L & \mathsf{th}(L,R\,\|\,i+1) & =: & R\mathbin{;}\mathsf{th}(L,R,i)
\end{array} \\[4pt]
\mathsf{succth}(L,R\|N-1) \;=:\; \mathsf{th}(L,R,N)
\end{array}
$$

$$
\begin{array}{rcl}
\mathsf{lBoth}(S,Q) & =: & \mathsf{rA}(S)\cap\mathsf{rA}(Q) \\
\mathsf{NoLLBoth}(P,Q,S) & \leftrightarrow\mathbf{:} & \mathsf{lBoth}(Q,S)\cap P=\emptyset \\
\mathsf{NoLRBoth}(P,Q,S) & \leftrightarrow\mathbf{:} & \mathsf{NoLLBoth}(P,Q,S^{\smile}) \\
\mathsf{NoTogether}(P,Q) & \leftrightarrow\mathbf{:} & \mathsf{rA}(P)\cap\mathsf{lA}(P)\cap Q=\emptyset \\
\mathsf{NoTwice}([P]) & \boldsymbol{\Theta}\mathbf{:} & \mathsf{true}(P) \\
\mathsf{NoTwice}([P,Q|T]) & \boldsymbol{\Theta}\mathbf{:} & [\ \mathsf{RUniq}(P,Q),\qquad \mathsf{LUniq}(P,Q), \\
 & & \mathsf{NoTogether}(P,Q), \\
 & & \mathsf{RXcl}(P,Q),\qquad \mathsf{LXcl}(P,Q) \\
 & & |\mathsf{NoTwice}([P|T])\ ] \\
\mathsf{NoLRBoth}([P,Q]) & \boldsymbol{\Theta}\mathbf{:} & \mathsf{true}([P,Q]) \\
\mathsf{NoLRBoth}([P,Q,S|T]) & \boldsymbol{\Theta}\mathbf{:} & [\ \mathsf{NoLLBoth}(P,Q,S),\qquad \mathsf{NoLRBoth}(P,Q,S), \\
 & & \mathsf{NoLRBoth}(P,S,Q) \\
 & & |\mathsf{NoLRBoth}([P,Q|T])] \\
\mathsf{NoBoth}([P]) & \boldsymbol{\Theta}\mathbf{:} & \mathsf{true}(P) \\
\mathsf{NoBoth}([P,Q|T]) & \boldsymbol{\Theta}\mathbf{:} & [\ \mathsf{NoLRBoth}(P,Q,Q) \\
 & & |\{\mathsf{NoLRBoth}([P,Q|T]),\quad \mathsf{NoBoth}([P|T])\}\ ] \\
\mathsf{PlaceHolders}([P,Q|T]) & \boldsymbol{\Theta}\mathbf{:} & \{\mathsf{NoTwice}([P,Q|T]),\qquad \mathsf{NoBoth}([P,Q|T])\} \\
\mathsf{keyFunc}([A],L,R\,\|\,I) & =: & \mathsf{succth}(L,R,I)\mathbin{;}A^{\smile} \\
\mathsf{keyFunc}([A,B|T],L,R\,\|\,J-1) & =: & \mathsf{th}(L,R,J)\mathbin{;}A^{\smile}\cap\mathsf{keyFunc}([B|T],L,R,J) \\
\mathsf{Key}([L,R|S]) & \leftrightarrow\mathbf{:} & \mathsf{RUniq}(\mathsf{keyFunc}(S,L,R,0)) \\
\mathsf{IsA}([Y,P,F]) & \boldsymbol{\Theta}\mathbf{:} & [\boldsymbol{\imath}][F\subseteq Y, F\cap P=\emptyset] \\
\mathsf{IsA}([Y,P,E,F|T]) & \boldsymbol{\Theta}\mathbf{:} & [\boldsymbol{\imath}][E\subseteq F|\ \mathsf{IsA}([Y\cap P,F|T])] \\
\mathsf{IsAChains}([Y,P]) & \boldsymbol{\Theta}\mathbf{:} & \mathsf{true}([Y,P]) \\
\mathsf{IsAChains}([Y,P,R|S]) & \boldsymbol{\Theta}\mathbf{:} & [\ \mathsf{true}(Y)\ |\ \{\mathsf{IsA}([Y,P|R]),\mathsf{IsAChains}([Y,P|S])\}\ ] \\
\mathsf{DotDots}([P]) & \boldsymbol{\Theta}\mathbf{:} & \mathsf{true}(P) \\
\mathsf{DotDots}([P,E,R,F|D]) & \boldsymbol{\Theta}\mathbf{:} & [\ R\subseteq\mathsf{rA}(E-P)\mathbin{;}(F-P) \\
 & & |\mathsf{DotDots}([P|D])\ ]
\end{array}
$$

Fig. 5. Some macros and templates submitted to Metamorpho for treatment of ER-models.

sequence of definitions, to culminate in a template definition

$$
\begin{array}{ll}
\mathsf{ERmodel}(& \mathit{IsAChains},\ \mathit{Relations},\ \mathit{DotDots}, \\
 & \mathit{LeftUniques},\ \mathit{RightUniques}, \mathit{Attributes},\ \mathit{Keys}, \\
 & \mathit{PlHolders},\ \mathit{Left},\ \mathit{Right},\ \mathit{Individuals},\ \mathit{NullTup}, \\
 & \mathit{PN},\ \mathit{LN},\ \mathit{RN},\ \mathit{YN},\ \mathit{NN}\ ) \qquad\qquad \boldsymbol{\Theta}\mathbf{:}\quad \cdots
\end{array}
$$

The ERmodel template, when invoked with actual parameters, will generate a relational theory reflecting the semantics of a specific ER-model. An example of invocation is the following (not very significant, indeed, as an ER-model):

```
ERmodel( [[r11, r15], [r12, r13, r14]], -- IsA-chains
         [r6, r7, r8, r9, r10, r16],    -- Relations and Attributes
                                        -- DotsDots:
         [r11, r10, r12,                -- r10 (total on both sides) relates r11 with r12
           r15, r16, r14],              -- r16 (total on both sides) relates r15 with r14
         [r16],                         -- LeftUniques (left-functional relations),
         [r10],                         -- RightUniques (functional relations).
                                        -- Attributes:
         [r9, r11, 0,                   -- r9 is an optional attribute on r11-entities
           r8, r15, 1,                  -- r8 is a mandatory attribute on r15-entities
           r7, r15, 1,                  -- r7 is a mandatory attribute of entity r15
           r6, r14, 1],                 -- r6 is a mandatory attribute of entity r14
                                        -- Keys:
         [[r7, r8],                     -- r7,r8 form a key of r15
           [r6]],                       -- r6 is a key of r14
                                        -- PlaceHolders,Left,Right,Individuals,NullTup:
         r5,                            -- r5 represents the collection of all placeholders...
         r1, r2,                        -- ...in a universe with projections r1,r2
         r3,                            -- ...with individuals r3
         r4,                            -- ...and with void tuple r4
         π, λ, ϱ, υ, ε                  -- Corresponding external names
).
```

When a text containing this single invocation is loaded into Metamorpho as proper axiom file (after the whole sequence of definitional macros and templates has been fed into it), a theory consisting of 5 definitions (namely, $\pi =: r_5, \lambda =: r_1, \ldots, \varepsilon =: r_4$) and 133 equations will arise: a few of the equalities axiomatize the flat tuple domain, all others translate the given ER-model.

11. Further work

The obvious line of attack for further work is removing some restrictions we imposed for technical reasons. This work was performed under some simplifying assumptions: we did assume that we work only with attributes on entities, and that we have a rather scant selection of cardinality restrictions. Both assumptions are not essential for our approach, and we feel that they should be removed. Another technical issue addresses the fact that we work with binary relations only. The discussions concerning projections shows, however, that it should not be too difficult to extend our set up for incorporating n-ary relations (although the notation then becomes slightly unbearable).

From a modelling point of view, we work here in a somewhat untyped environment: we do not have sorts for different entities, but rather assume that one sort fits all. This is fairly problematic in applications, and not entirely practical. Introducing sorts is another step we feel should be undertaken (along with a more detailed comparison of both approaches). This together with further developing the Metamorpho system (cf. [10]) will be some challenging tasks for the future.

Acknowledgements

This research was in part supported through grants from the Exchange Programme for Scientists between Italy and Germany from the Italian Ministry of Foreign Affairs & Deutscher Akademischer Austauschdienst, from which both authors were funded at different times during Fall 2000 and Summer 2001. The first author's work was additionally partially funded through a grant from *Progetto speciale I.N.D.A.M./GNIM* "*Nuovi paradigmi di calcolo*: *Linguaggi e Modelli*" while visiting what is now the Dipartimento di Informatica at the University of L'Aquila. The second author was partly funded by the MURST/MIUR 40% project *Aggregate- and number-reasoning for computing*: *from decision algorithms to constraint programming with multisets*, *sets*, *and maps*. This work also benefited from the collaboration fostered by the European COST Action 274 (TARSKI, cf. `http://www.tarski.org`).

The referees' comments and suggestions were very helpful and are appreciated.

References

[1] R. Berghammer, G. Schmidt, Relational specifications. in: C. Rauszer (Ed.), Proc. XXXVIII Stefan Banach Seminar on Algebraic Methods in Logic and Their Computer Science Applications, Banach Center Publications, Vol. 28, Warsaw, 1993, Institute of Mathematics, Polish Academy of Sciences, pp. 167–190.

[2] C. Brink, W. Kahl, G. Schmidt (Eds.), Relational Methods in Computer Science, Advances in Computing Science, Springer, Wien, New York, 1997.

[3] D. Cantone, E.G. Omodeo, A. Policriti, Set Theory for Computing, Springer, Berlin, 2001.

[4] I. Claßen, M. Löwe, S. Waßerroth, J. Wortmann, Static and dynamic semantics of entity-relationship models based on algebraic methods, Technical Report, Department of Computer Science, Technical University, Berlin, 1994.

[5] E.F. Codd, A relational model for large shared data banks, Comm. ACM 13 (6) (1970) 377–387.

[6] E.-E. Doberkat, Deleting the root of a heap, Acta Inform. 17 (1982) 244–265.

[7] E.-E. Doberkat, Generating an algebraic specification from an ER-model, Internat. J. Software Eng. Knowledge Eng. 7 (4) (1997) 525–552.

[8] E.-E. Doberkat, The categorial semantics of ER-models, Technical Report, Chair for Software Technology, University of Dortmund, 1999.

[9] E.-E. Doberkat, E.G. Omodeo, Algebraic semantics of ER-models in the context of the calculus of relations, Part II: dynamic view, in: H.M. de Swart (Ed.), Relational Methods in Computer Science, Lecture Notes in Computer Science, Vol. 2561, Springer, Berlin, December 2002, pp. 50–65.

[10] A. Formisano, E.G. Omodeo, M. Temperini, Goals and benchmarks for automated map reasoning, J. Symbolic Comput. 29 (2) (2000) 259–297.

[11] M. Gogolla, U. Hohenstein, Towards a semantic view of an extended entity-relationship model, ACM Trans. Database Systems 16 (1991) 369–416.

[12] R. Hettler, Zur Übersetzung von E/R-Schemata nach SPECTRUM, Technical Report TUM I-9333, Technical University, Munich, 1993.

[13] U. Hohenstein, Formale Semantik eines erweiterten Entity-Relationship Modells, Teubner, Stuttgart und Leipzig, 1993.

[14] A. Jaoua, N. Belkhiter, H. Ounalli, T. Moukam, Databases, in: C. Brink, W. Kahl, G. Schmidt (Eds.), Relational Methods in Computer Science, Advances in Computing Science, Springer, Wein, New York, 1997, pp. 197–210.

[15] B. Meyer, Object-oriented Software Construction, 5th Edition, Prentice-Hall, Englewood Cliffs, NJ, 1998.

[16] E.G. Omodeo, E.-E. Doberkat, Algebraic semantics of ER-models in the context of the calculus of relations, Part I: static view, Electron. Notes Theoret. Comput. Sci. 44 (3) (2003).

[17] M. Page-Jones, Fundamentals of Object-Orientied Design in UML, Dorset House Publishing & Addison-Wesley, New York and Boston, 2000.

[18] G. Schmidt, C. Hattensperger, M. Winter, Heterogeneous relation algebra, in: C. Brink, W. Kahl, G. Schmidt (Eds.), Relational Methods in Computer Science, Advances in Computing Science, Springer, Wein, New York, 1997, pp. 39–53.

[19] E. Schröder, Vorlesungen über die Algebra der Logik (exakte Logik), Vol. 2.1, Teubner, Stuttgart, 1891. Reprinted by Chelsea Publishing Co., New York, 1966.

[20] E. Schröder, Vorlesungen über die Algebra der Logik (exakte Logik), in: Algebra und Logic der Relative, Vol. 3, part 1, Teubner, Leipzig, 1895. Reprinted by Chelsea Publishing Co., New York, 1966.

[21] A. Tarski, S. Givant, A formalization of set theory without variables, in: Vol. 41, Colloquium Publications, American Mathematical Society, Providence, RI, 1987.

[22] B. Thalheim, Entity-Relationship Modeling: Foundations of Database Technology, Springer, Berlin, 2000.