The Dynamics of Database Views

Marcelo Arenas and Leopoldo Bertossi

P. Universidad Católica de Chile, Escuela de Ingeniería, Departamento de Ciencia de Computación, Casilla 306, Santiago 22, Chile. {marenas,bertossi}@ing.puc.cl

Abstract. The dynamics of relational database can be specified by means of Reiter's formalism based on the situation calculus. The specification of transaction based database updates is given in terms of Successor State Axioms (SSAs) for the base tables of the database. These axioms completely describe the contents of the tables at an arbitrary state of the database that is generated by the execution of a legal primitive transaction, and thus solve the frame problem for databases. In this paper we show how to derive action–effect based SSAs for views from the SSAs for the base tables. We prove consistency properties for those axioms. In addition, we establish the relationship between the derived SSA and the view definition as a static integrity constraint of the database. We give applications of the derived SSAs to the problems of view maintenance, and checking, proving, and enforcement of integrity constraints.

1 Introduction

The situation calculus (SC) [MH1] is a family of many sorted languages of predicate logic that contain domain individuals, actions, and situations (states[1]) at the same first order object level. So, first order quantifications over all these sorts of individuals are allowed. Since those languages are designed to represent knowledge and reason about actions and dynamic properties subject to discrete change due to action executions, it is natural to think of applying them to specify the dynamics of a relational database. This was done by Reiter in [Re2] on the basis of his formalism for solving the frame problem [Re1], that is the problem of obtaining a succinct specification of the properties that do not change when actions are executed. In [Be1] the main features of an automated system able to reason with and from those database specifications are presented.

According to Reiter, the specification of transaction based database updates can be given in terms of *Successor State Axioms* (SSAs) for the base tables of the database. These axioms completely describe the contents of the tables at an arbitrary state of the database that is generated by the execution of a legal primitive transaction[2], and thus solve the frame problem for databases.

[1] In this paper we do not make any distinction between situations and states.

[2] To be precise, a primitive transaction is said legal if its preconditions at the execution state are satisfied.

The SSAs usually come from *Effect Axioms* by means of a compilation process that materializes at the object level the assumption that the effect axioms provide all the conditions under which an entry in a table may change its truth value [Re1]. Then, they have a very particular syntactical form and we call them *action-effect based SSAs*. These kind of SSAs provide a lot of information about the dynamics of the tables.

In database theory and praxis, views are tables that are defined in logical terms from the base tables of the database. They are usually virtual tables, but they can be *materialized* as physical tables for certain purposes, e.g. in data warehousing [CD1]. They are an important component of the database. In this paper we show how to derive action–effect based SSAs for views from the SSAs for the base tables. This problem was originally formulated in [Re2]. We also prove consistency properties for those axioms. This is done in section 5, after having made our problem precise in section 4. In section 6, we establish the relationship between the derived SSA and the view definition as a static integrity constraint of the database.

Having appropriate SSAs for views allows us to solve some reasoning and computational tasks by processing the information contained in the axioms. It is in this way that we provide applications of the derived SSAs for views to the solution of some interesting problems in databases. This is done on the assumption that the specification of the database dynamics is given in terms of SSAs. In section 7 we present an algorithm for view maintenance derived from the SSAs for the view. In section 8 we apply SSAs for views to the problem of integrity constraints checking/proving. In section 9 we consider the problem of embedding desired integrity constraints in a specification based on SSAs.

In section 10, we draw some conclusions and discuss possible connections of our work to other interesting problems in databases that are worth being further explored. In section 2 we review Reiter's formalism, which is further discussed in section 3.

The proofs of some of the propositions in this paper are rather long and technical. For this reason they are included in an appendix. Nevertheless, we decided to leave the statements of the propositions, sometimes also a little technical, in the main body of the paper because they provide mechanisms for computing SSAs for some typical forms of views definitions.

2 The Situation Calculus and Database Updates

Characteristic ingredients of a particular language $\mathcal{L}$ of the situation calculus, besides the usual symbols of predicate logic, are: (1) Sorts *action*, *situation*, and *individual* (this last one for the individuals in the domain; this sort could be split into subsorts if necessary); (2) Predicate symbols of the sort (*individual*, ... , *individual*, *situation*) to denote tables. These are predicates that depend on the state of the world and can be thought as the tables in a relational database[3]; (3)

[3] In the AI literature they are called fluents.

Operation symbols of the sort $(individual, \ldots, individual) \rightarrow action$ for denoting actions with individuals as parameters (or applied to individuals), for example, $enroll(\cdot)$ may be an operation, and $enroll(john)$ becomes an action term (or a term of sort *action*. Actions correspond to the primitive transactions of the database; (4) A constant, S_0, to denote the initial state; (5) An operation symbol *do* of sort $(action, situation) \rightarrow situation$, that executes an action at a given state producing a successor state.

In these languages there are first order variables for individuals of each sort, so it is possible to quantify over individuals, actions, and situations. They are usually denoted by $\forall \bar{x}$, $\forall a$, $\forall s$, respectively.

The specification of a dynamically changing world, by means of an appropriate language of the situation calculus, consists in stating the laws of evolution of the world. This is typically done by specifying: (1) Fixed, state independent, but domain dependent knowledge about the individuals of the world; (2) Knowledge about the state of the world at the initial situation S_0 given in terms of formulas that do not mention any state besides S_0; (3) Preconditions for performing the different actions (or making their execution possible). We introduce a predicate *Poss* in $\mathcal{L}$ of sort $(action, situation)$, so that $Poss(a, s)$ says that the execution of action a is possible in state s; (4) The immediate (positive or negative) effects of actions in terms of the tables whose truth values we know are changed by their execution.

In Reiter's formalism, the knowledge contained in items (1) and (2) above is considered the initial database Σ_0. The information given in item (3) is formalized by means of action precondition axioms (*APA*s) of the form $Poss(A(\bar{x}), s) \equiv \pi_A(\bar{x}, s)$, for each action name A, where $\pi_A(\bar{x}, s)$ is a SC formula that is *simple in* s, that is, it contains no state term other than s, in particular, no *do* symbol, no quantifications on states, and no occurrences of the *Poss* predicate [LR2] (later on we will give the precise definition). Finally, item (4) is expressed by effect axioms for pairs (primitive transaction, table):

Positive Effects Axioms: For some pairs formed by a table F and an action name A, an axiom of the form:

$$\forall(\bar{x}, \bar{y}, s)[Poss(A(\bar{y}), s) \wedge \varphi_F^+(\bar{y}, \bar{x}, s) \supset F(\bar{x}, do(A(\bar{y}), s))]. \tag{1}$$

Intuitively, if the named primitive transaction A is possible, and the preconditions on the database, in particular, on the table F, represented by the metaformula $\varphi_F^+(\bar{y}, \bar{x}, s)$ are true at state s, then the statement $F(x)$ becomes true at the successor state $do(A(\bar{y}), s)$ obtained after execution of A at state s. Here, $\bar{x}, \bar{y}$ are parameters for the table and action. Notice that in general we have to kind of conditions: (a) preconditions for action executions, independently from any table they may affect; they are axiomatized by the *Poss* predicate; and (b) preconditions on the database for pairs table/action which make the changes possible (given that the action is already possible). These preconditions are represented by $\varphi_F^+(\bar{y}, \bar{x}, s)$.

Negative Effects Axioms: For some pairs formed by a table F and an action name A, an axiom of the form:

$$\forall(\bar{x}, \bar{y}, s)[Poss(A(\bar{y}), s) \wedge \varphi_F^-(\bar{y}, \bar{x}, s) \supset \neg F(\bar{x}, do(A(\bar{y}), s))]. \tag{2}$$

This is the case where action A makes table F to become false of $\bar{x}$ in the successor state.

Example 1. Consider an educational database as in [Re2], with the following ingredients. Tables: 1. $Enrolled(stu, c, s)$, student stu is enrolled in course c in the state s. 2. $Grade(stu, c, g, s)$, the grade of student stu in course c is g in the state s. Primitive Transactions: 1. $register(stu, c)$, register student stu in course c. 2. $change(stu, c, g)$, change the grade of student stu in course c to g. 3. $drop(stu, c)$, student stu drops the course c.

Action Precondition Axioms:

$$\forall(stu, c, s)[Poss(register(stu, c), s) \equiv \neg Enrolled(stu, c, s)].$$
$$\forall(stu, c, g, s)[Poss(change(stu, c, g), s) \equiv \exists g'\, Grade(stu, c, g', s)].$$
$$\forall(stu, c, s)[Poss(drop(stu, c), s) \equiv Enrolled(stu, c, s)].$$

Effect Axioms:

$$\forall(stu,c, s)[Poss(register(stu, c), s) \supset Enrolled(stu, c, do(register(stu, c), s))]$$
$$\forall(stu,c, s)[Poss(drop(stu, c), s) \supset \neg Enrolled(stu, c, do(drop(stu, c), s))]$$
$$\forall(stu,c, g, s)[Poss(change(stu, c, g), s) \supset Grade(stu, c, g, do(change(stu, c, g), s))]$$
$$\forall(stu,c, g, g', s)[Poss(change(stu, c, g'), s) \wedge g \neq g' \supset \neg Grade(stu, c, g, do(change(stu, c, g'), s))]$$

□

Since the situation calculus extends usual languages of predicate logic used to describe static and extensional aspects of relational databases with terms for denoting primitive transactions and database states, we gain in expressiveness, having now the possibility of representing knowledge and reasoning about the database dynamics and explicitly about the transactions involved in it. This is achieved in a single language that accommodates all the aspects of databases mentioned before. Nevertheless, as correctly pointed out in [Go1], this flexibility raises a new problem that we can already detect in the specification in the example above: it does not mention the usually many things (entries in tables) that do not change when a specific action is executed. So, we face the so called *frame problem*, consisting of providing a short, succinct, specification of the properties that persist after actions are performed.

An undesirable and naive (non) solution to this problem consists in writing down directly and explicitly all and the many properties that remain unchanged as primitive transactions are executed[4]. This would lead to large number of axioms. Fortunately, Reiter [Re1] discovered a simple solution to the frame problem as it appears in the situation calculus. It allows to construct a first order specification that accounts both for effects as non effects, from a specification that contains descriptions of effects only, as in the example above. We sketch this solution in the rest of this section.

For illustration, assume that we have only two positive effects laws for table F: (1) and

$$\forall(\bar{x}, \bar{z}, s)[Poss(A'(\bar{z}), s) \wedge \psi_F^+(\bar{z}, \bar{x}, s) \supset F(\bar{x}, do(A'(\bar{z}), s))]. \tag{3}$$

We may combine them into one general positive effect axiom for table F:

$$\forall(a, \bar{x}, s)[Poss(a, s) \wedge [\exists\bar{y}(a = A(\bar{y}) \wedge \varphi_F^+(\bar{y}, \bar{x}, s)) \vee \\ \exists\bar{z}(a = A'(\bar{z}) \wedge \psi_F^+(\bar{z}, \bar{x}, s))] \supset F(\bar{x}, do(a, s))].$$

In this form we obtain, for each table F, a general positive effect law of the form:

$$\forall(a, \bar{x}, s)[Poss(a, s) \wedge \gamma_F^+(a, \bar{x}, s) \supset F(\bar{x}, do(a, s))].$$

Analogously, we obtain, for each table F, a general negative effect axiom:

$$\forall(a, \bar{x}, s)[Poss(a, s) \wedge \gamma_F^-(a, \bar{x}, s) \supset \neg F(\bar{x}, do(a, s))].$$

In this way we have represented, for each table F, in one single axiom all the actions and the corresponding conditions on the database that can make $F(\bar{x})$ to be true at an arbitrary successor state obtained by executing a legal action. In the same way we can describe when $F(\bar{x})$ becomes false. We will see later on how to combine these two axioms general axioms for table F to give a full account of its dynamics.

Example 2. (cont'd) In the educational example we obtain the following general effect axioms for the table *Grade*:

$$\forall(a, stu, c, g, s)[Poss(a, s) \wedge a = change(stu, c, g) \supset Grade(stu, c, g, do(a, s))]$$
$$\forall(a, stu, c, g, s)[Poss(a, s) \wedge \exists g'(a = change(stu, c, g') \wedge g \neq g') \\ \supset \neg Grade(stu, c, g, do(a, s))].$$

□

The basic assumption underlying Reiter's solution to the frame problem is that the general effect axioms, both positive and negative, for a given table F, contain

[4] We consider "action" and "primitive transaction" as synonymous.

all the possibilities for table F to change its truth value from a state to a successor state. Actually, for each table F we generate its Successor State Axiom:

$$\forall(a,s)Poss(a,s) \supset \forall\bar{x}[F(\bar{x}, do(a,s)) \equiv (\gamma_F^+(a,\bar{x},s) \vee (F(\bar{x},s) \wedge \neg\gamma_F^-(a,\bar{x},s)))]. \quad (4)$$

Here, γ^+ and γ^- are of the form $\bigvee_{some\ A's} \exists\bar{u}(a = A(\bar{u}) \wedge \varphi(\bar{u},\bar{x},s))$, meaning that action A, under condition φ, makes $F(\bar{x}, do(A,s))$ true, in the case of γ^+, and false, in the case of γ^-. Thus, the SSA says that if action a is possible, then F becomes true at the successor state that results from the execution of action a if and only if a is one of the actions causing F to be true (and for which the corresponding preconditions, φ, are true), or F was already true before executing a and this action is not one of the actions that falsify F.

Example 3. (cont'd) In our running example, we obtain the following SSAs for the tables in the database:

$$\forall(a,s)Poss(a,s) \supset \forall(stu,c)[Enrolled(stu,c,do(a,s)) \equiv a = register(stu,c) \vee Enrolled(stu,c,s) \wedge a \neq drop(stu,c)]$$
$$\forall(a,s)Poss(a,s) \supset \forall(stu,c,g)[Grade(stu,c,g,do(a,s)) \equiv a = change(stu,c,g) \vee Grade(stu,c,g,s) \wedge \neg\exists g'(a = change(stu,c,g') \wedge g' \neq g)].$$

□

Notice that, provided there is complete knowledge about the contents of the tables at the initial state, the SSAs completely describe the contents of the tables at every state that can be reached by executing a finite sequence of legal primitive transactions (that is for which the corresponding $Poss$ conditions are satisfied). The SSAs have a nice inductive structure that makes some reasoning tasks easy, at least in principle.

Reiter's SSAs materialize at the object level a meta assumption on explanation closure: the changes caused on a table are only those that are explicitly represented in the effects axioms. This construction of the SSAs has some similarity with negation as failure in logic programming: if we want to determine if the truth value of the statement $F(\bar{x})$ persists, we just check the general effect axioms to find out if there are any reasons for change, if we do not find anything there, we establish that $F(\bar{x})$ does not change. Also in this line of reasoning, the construction process of the SSAs is reminiscent of Clark's predicate completion in logic programming [Cl1]. As in that case, for the new specification to work properly one needs to makes some additional ontological assumptions about the models of the situation calculus. Following [LR1], we will assume that the following *Foundational Axioms of the Situation Calculus* (*FA*s) underlie any database specification: 1. Unique Names Axioms for Actions (*UNAA*): $A_i(\bar{x}) \neq A_j(\bar{y})$, for all different action names A_i, A_j; and $\forall(\bar{x},\bar{y})[A(\bar{x}) = A(\bar{y}) \supset \bar{x} = \bar{y}]$, for every action name A. 2. Unique Names Axioms for States: $S_0 \neq do(a,s)$, $do(a_1,s_1) = do(a_2,s_2) \ \supset \ a_1 = a_2 \wedge s_1 = s_2$.

For some reasoning tasks we need an Induction Axiom on States:

$$\forall P\ [P(S_0) \wedge \forall s \forall a\ (P(s) \supset P(do(a,s))) \quad \supset \quad \forall s\ P(s)],$$

that has the effect of restricting the domain of situations to the one containing the initial situation and the situations that can be obtained by executing a finite number of actions. In this way, no "ghost" situations may appear. The axiom is second order, but for some reasoning tasks, like proving integrity constraints, reasoning can be done at the first order level [LR1, Be1].

Finally, we will be usually interested in reasoning about states that are accessible from the initial situation by executing a finite sequence of legal actions. This *accessibility relation* on states, $\leq$, can be defined from the induction axiom plus the conditions $\neg s < S_0,\ \ s < do(a,s') \equiv Poss(a,s') \wedge s \leq s'$, which we call *(less-axioms)*.

Summarizing, a specification Σ, in the SC, of transaction based database updates consists of the sets: $\Sigma_0 \cup APAs \cup SSAs \cup FAs \cup (less - axioms)$.

3 The Structure of SSAs

In [Re2] Reiter considers database specifications [5] whose tables have SSAs of the general form

$$\forall(a,s)Poss(a,s) \supset \forall \bar{x}[F(\bar{x}, do(a,s)) \ \equiv\ \Phi_F(a,\bar{x},s)], \tag{5}$$

where Φ_F is an arbitrary formula that is simple in s, in particular, it does not contain the *do* symbol. Notice that SSAs of the form (4) are particular cases of (5) that appear in specifications where the dynamics of the tables were originally defined in terms of effect axioms only. This is the most usual and interesting case, and we call those SSAs of the form (4) "action–effect based". In the rest of the paper we will concentrate on this kind of SSAs. To make their processing correct and more clear, we need to be more precise about their components. We will also agree on a sort of canonical form for them. We will everywhere assume that we have a specification Σ for the database updates as introduced at the end of section 2.

Definition 1. *A database table P has an action-effect based Successor State Axiom (aeSSA) wrt Σ iff Σ contains a formula of the following form as a SSA for P:*

$$\forall(a,s)Poss(a,s) \supset \forall \bar{x}[P(\bar{x}, do(a,s)) \equiv \gamma^{+}(a,\bar{x},s) \vee (P(\bar{x},s) \wedge \neg\gamma^{-}(a,\bar{x},s))],$$

[5] In this paper, whenever we refer to a "database specification", we are talking about a specification of the dynamics of a database as presented before.

with

$$\Sigma \models \forall(a,\bar{x},s)[\gamma^+(a,\bar{x},s) \supset \neg\gamma^-(a,\bar{x},s)], \tag{6}$$

$$\gamma^+(a,\bar{x},s) = \bigvee_{i=1}^{m} \exists\bar{x}_{A_i}(a = A_i(\bar{x}_{A_i}) \wedge \varphi^{\gamma^+}_{A_i}(\bar{x}_{A_i},\bar{x},s)).$$

$$\gamma^-(a,\bar{x},s) = \bigvee_{i=1}^{n} \exists\bar{x}_{B_i}(a = B_i(\bar{x}_{B_i}) \wedge \varphi^{\gamma^-}_{B_i}(\bar{x}_{B_i},\bar{x},s)).$$

where $A_i \neq A_j$ $(1 \leq i < j \leq m)$, $B_i \neq B_j$ $(1 \leq i < j \leq n)$, *and the* $\varphi^{\gamma^+}_{A_i}(\bar{x}_{A_i},\bar{x},s)$ *and* $\varphi^{\gamma^-}_{B_i}(\bar{x}_{B_i},\bar{x},s)$ *are formulas simple in s that do not contain any action term* [6].

Here, (6) is a *consistency condition*, first considered in [Re1], which ensures that a same action cannot have both a positive and a negative effect on a same entry in a table.

Example 4. (cont'd) The SSAs for *Enrolled* and *Grade* in our example can be easily transformed in order to meet all the requirements in definition 1[7]:

$$\begin{aligned}\forall(a,s)Poss(a,s) \supset \quad & \forall(stu,c)[\mathit{Enrolled}(stu,c,do(a,s)) \equiv \gamma^+(a,stu,c,s) \vee \\ & \qquad \mathit{Enrolled}(stu,c,s) \wedge \neg\gamma^-(a,stu,c,s)] \\ \forall(a,s)Poss(a,s) \supset \; & \forall(stu,c,g)[\mathit{Grade}(stu,c,g,do(a,s)) \equiv \delta^+(a,stu,c,g,s) \vee \\ & \qquad \mathit{Grade}(stu,c,g,s) \wedge \neg\delta^-(a,stu,c,g,s)],\end{aligned}$$

with

$$\begin{aligned}\gamma^+(a,stu,c,s) &= \exists(x_1,x_2)(a = \mathit{register}(x_1,x_2) \wedge \varphi^{\gamma^+}_{\mathit{register}}(x_1,x_2,stu,c,s)), \\ \gamma^-(a,stu,c,s) &= \exists(x_3,x_4)(a = \mathit{drop}(x_3,x_4) \wedge \varphi^{\gamma^-}_{\mathit{drop}}(x_3,x_4,stu,c,s)), \\ \delta^+(a,stu,c,g,s) &= \exists(y_1,y_2,y_3)(a = \mathit{change}(y_1,y_2,y_3) \wedge \\ &\qquad\qquad \varphi^{\delta^+}_{\mathit{change}}(y_1,y_2,y_3,stu,c,g,s)) \\ \delta^-(a,stu,c,g,s) &= \exists(y_4,y_5,y_6)(a = \mathit{change}(y_4,y_5,y_6) \wedge \\ &\qquad\qquad \varphi^{\delta^-}_{\mathit{change}}(y_4,y_5,y_6,stu,c,g,s)).\end{aligned}$$

---

[6] For simplicity, we will sometimes denote an aeSSA as follows:

$$\begin{aligned}\forall(a,s)Poss(a,s) \supset \forall\bar{x}[P(\bar{x},do(a,s)) \equiv & \bigvee_A \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi^{\gamma^+}_A(\bar{x}_A,\bar{x},s)) \\ & \vee \; P(\bar{x},s) \; \wedge \neg \bigvee_B \exists\bar{x}_B(a = B(\bar{x}_B) \wedge \varphi^{\gamma^-}_B(\bar{x}_B,\bar{x},s))].\end{aligned}$$

[7] It should be clear that this transformation can be automated.

and

$$\varphi^{\gamma^+}_{register}(x_1, x_2, stu, c, s) = x_1 = stu \wedge x_2 = c,$$
$$\varphi^{\gamma^-}_{drop}(x_3, x_4, stu, c, s) = x_3 = stu \wedge x_4 = c,$$
$$\varphi^{\delta^+}_{change}(y_1, y_2, y_3, stu, c, g, s) = y_1 = stu \wedge y_2 = c \wedge y_3 = g,$$
$$\varphi^{\delta^-}_{change}(y_4, y_5, y_6, stu, c, g, s) = y_4 = stu \wedge y_5 = c \wedge y_6 \neq g.$$

□

Action-effect based SSAs are much more natural and informative than general axioms of the form (5), since we can read directly from them which are the actions that can make a table change its truth values and under which conditions. In fact, when a table has an aeSSA, it is possible to create a simple procedure for updating it when actions are executed. To be precise, suppose that we have a table P with a SSA as in definition 1, and we execute an action $A(\bar{c})$ in a state s. For updating P, we consider this table as a set of tuples of the form $\bar{x}$, without the state parameter. Thus, if we want to compute the tuples in P in the state $do(A(\bar{c}), s)$, we need to compute the tuples that must be added to the table (Δ^+_P) and the tuples that must be removed from this table (Δ^-_P). These sets can be computed using the procedures shown in Figure 1.

Once we have Δ^+_P and Δ^-_P, it is possible to compute P in $do(A(\bar{c}), s)$ by adding to the tuples present in P at state s, the set Δ^+_P and removing those from Δ^-_P, i.e. by computing $(P \cup \Delta^+_P) - \Delta^-_P$.

function Add($A(\bar{y})$:action)
begin
  $\Delta^+_P := \emptyset$; $i := 1$;
  while $(A \neq A_i \wedge i \leq m)$ $i = i + 1$;
  if $i \leq m$ **then**
    for each tuple $\bar{x}$ **do**
      if $\varphi^{\gamma^+}_{A_i}(\bar{y}, \bar{x}, s)$ holds **then**
        $\Delta^+_P = \Delta^+_P \cup \{\bar{x}\}$;
  return(Δ^+_P);
end

function Remove($A(\bar{y})$:action)
begin
  $\Delta^-_P := \emptyset$; $i := 1$;
  while $(A \neq B_i \wedge i \leq n)$ $i = i + 1$;
  if $i \leq n$ **then**
    for each tuple $\bar{x}$ **do**
      if $\varphi^{\gamma^-}_{B_i}(\bar{y}, \bar{x}, s)$ holds **then**
        $\Delta^-_P = \Delta^-_P \cup \{\bar{x}\}$;
  return(Δ^-_P);
end

Fig. 1. Functions for computing Δ^+_P and Δ^-_P.

We can now rewrite the consistency condition (6) as $\Delta^+_P \cap \Delta^-_P = \emptyset$. Thus, this condition ensures that when we execute an action there are no tuples that must be both added and removed, a natural and desirable property. Additionally, this condition ensures that it is possible to compute the table P, in a successor state, using any of these procedures: $(P \cup \Delta^+_P) - \Delta^-_P$ or $(P - \Delta^-_P) \cup \Delta^+_P$, i.e. it is not important if we add and then remove tuples from P or remove and then add tuples to P.

4 SSAs for Views?

Given the specification of the dynamics of a database, with SSAs for its base tables, it is natural to consider the introduction of database views, that is, of new, usually virtual, tables that are defined in terms of the base tables by means of the logical formulas. To be precise, in our context, a view will be a new table $V(\bar{x}, s)$ with an explicit definition of the form:

$$\forall(\bar{x}, s)[V(\bar{x}, s) \ \equiv_{df} \ \psi(\bar{x}, s)], \tag{7}$$

where $\psi(\bar{x}, s)$ is a SC formula that is simple in s and mentions base tables only.

To be precise, a formula simple in situation s is an SC formula that can be constructed by means of the following rules:

1. φ does not mention any situation and action terms.
2. φ is of the form $F(\bar{t}, s)$, where F is a table name, and $\bar{t}$ is a tuple of terms for individuals.
3. Formulas simple in s can be constructed by means of the usual propositional connectives plus quantifications on individuals from other formulas simple in s[8].

In order to update a view its definition could be used in terms of the already updated base tables. Nevertheless, it is natural to consider the possibility of having a special SSA for that view. In this case we could reason about the view evolution directly from its SSA (see sections 7 and 10 for more on this issue).

In [Re2], Reiter considers the problem of generating SSAs for views, and he gives a solution based on his *regression operator* $\mathcal{R}$[Re1], which, applied to a formula evaluated at a successor state, returns an equivalent formula evaluated at the current state. This is done by appealing to the SSAs of the tables involved in the formula. More precisely, if table F, appearing in a formula φ, has a SSA of the form

$$\forall(a, s)Poss(a, s) \supset \forall(x_1, x_2 \ldots, x_n)[F(x_1, x_2, \ldots, x_n, do(a, s)) \equiv \Phi_F(x_1, x_2, \ldots, x_n, a, s)],$$

then the operator, $\mathcal{R}$, applied to a formula φ, replaces each occurrence of a formula of the form $F(t_1, t_2, \ldots, t_n, do(A, S))$ by $\Phi_F|^{x_1, x_2, \ldots, x_n, a, s}_{t_1, t_2, \ldots, t_n, A, S}$.

Applying the regression operator to (7), where s was replaced by $do(a, s)$, one obtains a SSA for V:

$$\forall(a, s)Poss(a, s) \supset \forall\bar{x}[V(\bar{x}, do(a, s)) \equiv \mathcal{R}[\psi(\bar{x}, do(a, s))]]. \tag{8}$$

As the application of $\mathcal{R}$ on the RHS of (8) eliminates the *do* symbol, we obtain a SSA of the general form (5). Nevertheless, this procedure does not necessarily produce a SSA of the form (4).

[8] We are following the definition given in [LR2], except for the fact that there action terms are allowed, something we do not need here.

In the next sections we show how to generate *action–effect based* SSAs (of the form (4)) for views when we have such axioms for the base tables. We will also verify that the resulting SSAs satisfy the consistency condition, if the SSAs for the participating base tables do.

5 Generating SSAs for Views

In this section we will address the problem of constructing SSAs of the canonical form presented in definition 1 for views whose definitions are of the form (7). This will be done by induction on the structure of formulas simple in s that where introduced in the preceding section, and then we will be in position to establish the following

Theorem 1. *Let Σ be a database specification, all whose base tables have aeSSAs. Let $\psi(\bar{x}, s)$ be a formula in the language of the specification that is simple in s. Then there are formulas $\gamma^+(a, \bar{x}, s)$ and $\gamma^-(a, \bar{x}, s)$, such that*

$$\Sigma \models \forall(a, s) Poss(a, s) \supset \forall \bar{x}[\psi(\bar{x}, do(a, s)) \equiv \gamma^+(a, \bar{x}, s) \vee (\psi(\bar{x}, s) \wedge \neg\gamma^-(a, \bar{x}, s))], \quad (9)$$

where γ^+ and γ^- have the form required by definition 1. In particular, they satisfy the consistency condition.

The proof of this theorem can be obtained from particular construction steps for the definition of the view. We consider the steps corresponding to those cases appearing in the transformation of the view definition into a prenex disjunctive normal form, whose basic conjuncts are tables, negations of tables and formulas without state terms. More specifically, we consider, according to the bottom–up construction of the normal form, the following cases and in the following order (1) the negation of a table, (2) the conjunction of formulas having a aeSSA, (3) the conjunction of a formula having a aeSSAwith a formula without state terms, (4) the disjunction of formulas having aeSSAs, but treated in terms of negations and conjunctions, (5) the quantifications on individuals.

Notice that this way of proving the theorem does not entail that in order to obtain SSA for a view, a preliminary transformation into prenex disjunctive normal form is necessary. A better way of achieving this would be to have a list of the resulting aeSSAs for all possible logical combinations of tables and formulas without state terms. Most of the cases are given below, and the remaining cases could be computed from them.

Before considering the inductive steps we mentioned before, we need a technical definition.

Definition 2. *Given a SC formula γ, we will denote by $Acc(\gamma)$ the set of action names appearing in it.*

Example 5. (cont'd) Consider the aeSSA for *Enrolled* and *Grade* shown in example 4. In that case, $Acc(\gamma^+) = \{register\}$, $Acc(\gamma^-) = \{drop\}$, $Acc(\delta^+) = \{change\}$, $Acc(\delta^-) = \{change\}$. □

This definition will be used in connection with aeSSAs, that, according to definition 1 and without loss of generality, require that each action name appears at most once in each of the γs, with all its arguments being existentially quantified variables and not shared with other actions in the same γ. For example, if the formula $a = enroll(john) \vee a = enroll(mary) \vee a = register(x)$ were a candidate to appear as a γ in an aeSSA, it could be replaced by $\exists y(a = enroll(y) \wedge (y = john \vee y = mary)) \vee \exists u(a = register(u) \wedge u = x)$.

Proposition 1. *If a table P has the following aeSSA in the specification Σ*

$$\forall(a,s)Poss(a,s) \supset \forall\bar{x}\ [P(\bar{x}, do(a,s)) \equiv \gamma^+(a,\bar{x},s) \vee P(\bar{x},s) \wedge \neg\gamma^-(a,\bar{x},s)], \quad (10)$$

then the following holds:

$$\Sigma \models \forall(a,s)Poss(a,s) \supset \forall\bar{x}[\neg P(\bar{x}, do(a,s)) \equiv \gamma^-(a,\bar{x},s) \vee \neg P(\bar{x},s) \wedge \neg\gamma^+(a,\bar{x},s)], \quad (11)$$

$$\Sigma \models \forall(a,\bar{x},s)[\gamma^-(a,\bar{x},s) \supset \neg\gamma^+(a,\bar{x},s)]. \quad (12)$$

We can see from (11) that the derived predicate $\neg P$ inherits an aeSSA from the aeSSA for table P. By (12), it also inherits the satisfaction of the consistency condition.

Proposition 2. *If specification Σ contains the following aeSSAs for the tables $P(\bar{x},s)$ and $Q(\bar{y},s)$*

$$\forall(a,s)Poss(a,s) \supset \forall\bar{x}\ [P(\bar{x}, do(a,s)) \equiv \gamma^+(a,\bar{x},s) \vee P(\bar{x},s) \wedge \neg\gamma^-(a,\bar{x},s)], \quad (13)$$

$$\forall(a,s)Poss(a,s) \supset \forall\bar{y}\ [Q(\bar{y}, do(a,s)) \equiv \delta^+(a,\bar{y},s) \vee Q(\bar{y},s) \wedge \neg\delta^-(a,\bar{y},s))], \quad (14)$$

then

$$\Sigma \models \forall(a,s)Poss(a,s) \supset \ \forall(\bar{x},\bar{y})[P(\bar{x}, do(a,s)) \wedge Q(\bar{y}, do(a,s)) \equiv \xi^+(a,\bar{x},\bar{y},s) \vee (P(\bar{x},s) \wedge Q(\bar{y},s) \wedge \neg\xi^-(a,\bar{x},\bar{y},s))] \quad (15)$$

and

$$\Sigma \models \forall(a,\bar{x},\bar{y},s)[\xi^+(a,\bar{x},\bar{y},s) \supset \neg\xi^-(a,\bar{x},\bar{y},s)]. \quad (16)$$

Here $\xi^+(a,\bar{x},\bar{y},s)$ is equal to

$$\bigvee_{A \in Acc(\gamma+) \cap Acc(\delta+)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s))$$

$$\bigvee$$

$$\bigvee_{A \in Acc(\gamma+) - Acc(\delta^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s))$$

$$\bigvee$$

$$\bigvee_{A \in Acc(\gamma+) \cap Acc(\delta^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s))$$

$$\bigvee$$

$$\bigvee_{A \in Acc(\delta+) - Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s) \wedge P(\bar{x}, s))$$

$$\bigvee$$

$$\bigvee_{A \in Acc(\delta+) \cap Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s) \wedge \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s) \wedge P(\bar{x}, s))$$

and $\xi^-(a, \bar{x}, \bar{y}, s)$ *is equal to* $\gamma^-(a, \bar{x}, s) \vee \delta^-(a, \bar{y}, s)$.

Although the derived SSA has not a simple form, it has a clear intuitive contents. In order for $P \wedge Q$ to be true at the successor state, there are five possible cases corresponding to the five disjuncts in the formula above: (a) both P and Q became true, (b) P became true and Q was true and did not change because the action cannot falsify it, (c) P became true and Q was true and did not change because, even when the action could falsify it, the precondition for this falsification was not satisfied, (d) and (e) are the same as (b) and (c), resp., but exchanging the roles of P and Q.

Proposition 3. *If in* Σ *there is the following aeSSA for a table* $P(\bar{x}, s)$

$$\forall (a, s) Poss(a, s) \supset \forall \bar{x} [P(\bar{x}, do(a, s)) \equiv \gamma^+(a, \bar{x}, s) \vee (P(\bar{x}, s) \wedge \neg \gamma^-(a, \bar{x}, s))],$$

and $\psi(\bar{x})$ *is a formula that contains no state or actions terms, then it holds*

$$\Sigma \models \forall (a, s) Poss(a, s) \supset \forall \bar{x} [P(\bar{x}, do(a, s)) \wedge \psi(\bar{x}) \equiv$$
$$\bigvee_A \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \psi(\bar{x})) \vee (P(\bar{x}, s) \wedge \psi(\bar{x}) \wedge \neg \gamma^-(a, \bar{x}, s))].$$

$$\Sigma \models \forall (a, \bar{x}, s) [\bigvee_A \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \psi(\bar{x})) \supset \neg \gamma^-(a, \bar{x}, s)].$$

This proposition has a straightforward proof. It considers the case of a view defined by a selection according to the condition ψ. The last statement in it corresponds to the consistency condition. Notice that this case cannot be obtained from proposition 2, because there is no aeSSAfor ψ. Nevertheless, if one artificially introduces a situational argument in it and considers a trivial SSA of the form $Poss(a, s) \supset \psi(\bar{x}, do(a, s)) \equiv \psi(\bar{x}, s)$, this case can be derived from proposition 2.

For the case of the universal quantification of some of the variables in a table, we have the following

Proposition 4. *If the table P has the following aeSSA in the specification Σ*

$$\forall(a,s)Poss(a,s) \supset \forall\bar{x}[P(\bar{x},do(a,s)) \equiv \gamma^+(a,\bar{x},s) \vee \\ P(\bar{x},s) \wedge \neg\gamma^-(a,\bar{x},s)], \quad (17)$$

where $\bar{x} = (x_1,\ldots,x_{i-1},x,x_{i+1},\ldots,x_n)$. Let $\bar{x}' := (x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)$. It holds

$$\Sigma \models \forall(a,s)Poss(a,s) \supset \forall\bar{x}'[\forall xP(\bar{x},do(a,s)) \equiv \xi^+(a,\bar{x}',s) \vee \\ \forall xP(\bar{x},s) \wedge \neg\xi^-(a,\bar{x}',s)] \quad (18)$$

$$\Sigma \models \forall(a,\bar{x}',s)[\xi^+(a,\bar{x}',s) \supset \neg\xi^-(a,\bar{x}',s)], \quad (19)$$

where $\xi^+(a,\bar{x}',s)$ is equal to

$$\bigvee_{A\in Acc(\gamma^+)-Acc(\gamma^-)} \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \forall x(\neg P(\bar{x},s) \supset \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)))$$
$$\bigvee$$
$$\bigvee_{A\in Acc(\gamma^+)\cap Acc(\gamma^-)} \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \forall x[(\neg P(\bar{x},s) \supset \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s))$$
$$\bigwedge$$
$$(P(\bar{x},s) \supset \neg\varphi_A^{\gamma^-}(\bar{x}_A,\bar{x},s))]),$$

and $\xi^-(a,\bar{x}',s)$ is equal to

$$\bigvee_{A\in Acc(\gamma^-)} \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \exists x\varphi_A^{\gamma^-}(\bar{x}_A,\bar{x},s)).$$

The obtained SSA has also a natural intuitive contents. It is possible to understand formula ξ^+ by considering its two cases: (a) $\forall x\ P$ becomes true at the successor state if the action can only make P true and for every individual for which P was false, the action made P true, (b) the action can make P true or false, but for every individual for which P was false, it made P true, and for every individual for which P was true, the action did not change P. In the case of ξ^-, $\forall x\ P$ becomes false at the successor state if the action can make P false and at least for one individual this happens.

Other logical cases can be handled as usual, in terms of the previous cases. In particular, the case of a projection as the existential quantification of a table is covered by the following

Corollary 1. *Let P be a table in a specification Σ with aeSSA:*

$$\forall(a,s)Poss(a,s) \supset \forall\bar{x}[P(\bar{x},do(a,s)) \equiv \gamma^+(a,\bar{x},s) \vee \\ P(\bar{x},s) \wedge \neg\gamma^-(a,\bar{x},s)], \quad (20)$$

where $\bar{x} = (x_1,\ldots,x_{i-1},x,x_{i+1},\ldots,x_n)$. Let $\bar{x}' := (x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n)$. It holds

$$\Sigma \models \forall(a,s)Poss(a,s) \supset \forall\bar{x}'[\exists xP(\bar{x},do(a,s)) \equiv \xi^+(a,\bar{x}',s) \vee \\ \exists xP(\bar{x},s) \wedge \neg\xi^-(a,\bar{x}',s)] \quad (21)$$

$$\Sigma \models \forall(a,\bar{x}',s)[\xi^+(a,\bar{x}',s) \supset \neg\xi^-(a,\bar{x}',s)], \quad (22)$$

where $\xi^+(a,\bar{x}',s)$ *is equal to*

$$\bigvee_{A\in Acc(\gamma^+)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \exists x \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)),$$

and $\xi^-(a,\bar{x}',s)$ *is equal to*

$$\begin{array}{c}\displaystyle\bigvee_{A\in Acc(\gamma^-)-Acc(\gamma^+)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \forall x(P(\bar{x},s) \supset \varphi_A^{\gamma^-}(\bar{x}_A,\bar{x},s)))\\ \bigvee \\ \displaystyle\bigvee_{A\in Acc(\gamma^-)\cap Acc(\gamma^+)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \forall x[(P(\bar{x},s) \supset \varphi_A^{\gamma^-}(\bar{x}_A,\bar{x},s))\\ \bigwedge \\ (\neg P(\bar{x},s) \supset \neg\varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s))]).\end{array}$$

Views defined by joins of tables can also be reduced to the already considered cases.

Example 6. It is interesting to notice that the consistency condition still holds for the aeSSA of an "inconsistent" view, that is an empty view of the form $V(\bar{x},s) \equiv_{df} P(\bar{x},s) \wedge \neg P(\bar{x},s)$. In fact, if P has the SSA

$$Poss(a,s) \supset P(\bar{x},do(a,s)) \equiv \gamma^+(\bar{x},a,s) \vee P(\bar{x},s) \wedge \neg\gamma^-(\bar{x},a,s),$$

and we apply the above methodology to the table $P(\bar{x},s)\wedge\neg P(\bar{x},s)$, we generate a SSA of the form

$$\begin{aligned}Poss(a,s) \supset P(\bar{x},do(a,s)) \wedge \neg P(\bar{x},do(a,s)) \equiv \delta^+(\bar{x},a,s) \vee\\ P(\bar{x},s) \wedge \neg P(\bar{x},s) \wedge \neg\delta^-(\bar{x},a,s),\end{aligned} \quad (23)$$

with $\delta^+(\bar{x},a,s) = \gamma^+(\bar{x},a,s) \wedge \gamma^-(\bar{x},a,s)$. By the consistency condition for P, we have that δ^+ is an inconsistent formula, and since the consistency condition for $P \wedge \neg P$ is of the form $\delta^+ \supset \neg\delta^-$, we may conclude that the consistency condition for $P \wedge \neg P$ also holds. □

Theorem 1 can be obtained from the propositions above. The next example illustrates the theorem and the combination of the different cases considered in the given propositions.

Example 7. (cont'd) We want to store the information about courses that are passed by officially registered students. We define a new table, V_1, as follows:

$$\begin{aligned}\forall(stu,c,s)[V_1(stu,c,s) \equiv\\ \exists g(Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge g \geq 50)].\end{aligned} \quad (24)$$

For including this view in our specification, we add to the initial database, Σ_0, the sentence:

$$\begin{aligned}\forall(stu,c)[V_1(stu,c,S_0) \equiv\\ \exists g(Enrolled(stu,c,S_0) \wedge Grade(stu,c,g,S_0) \wedge g \geq 50)].\end{aligned} \quad (25)$$

Next we compute an aeSSA for V_1. For this, we apply first proposition 2 to $Enrolled(stu,c,s) \wedge Grade(stu,c,g,s)$. In this case $Acc(\gamma^+) \cap Acc(\delta^+) = \emptyset$, $Acc(\gamma^+) - Acc(\delta^-) = \{register\}$, $Acc(\gamma^+) \cap Acc(\delta^-) = \emptyset$, $Acc(\delta^+) - Acc(\gamma^-) = \{change\}$, $Acc(\delta^+) \cap Acc(\gamma^-) = \emptyset$, then from the specification it follows that, for every action a that is possible in a state s:

$$
\begin{aligned}
& Enrolled(stu,c,do(a,s)) \wedge Grade(stu,c,g,do(a,s)) \equiv \\
& \quad [\exists(x_1,x_2)(a = register(x_1,x_2) \wedge x_1 = stu \wedge x_2 = c \wedge grade(stu,c,g,s)) \vee \\
& \quad \exists(y_1,y_2,y_3)(a = change(y_1,y_2,y_3) \wedge y_1 = stu \wedge y_2 = c \\
& \qquad\qquad \wedge y_3 = g \wedge Enrolled(stu,c,s)) \vee \\
& \qquad (Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge \\
& \quad \neg(\exists(x_3,x_4)(a = drop(x_3,x_4) \wedge x_3 = stu \wedge x_4 = c) \vee \\
& \quad \exists(y_4,y_5,y_6)(a = change(y_4,y_5,y_6) \wedge y_4 = stu \wedge y_5 = c \wedge y_6 \neq g)))].
\end{aligned}
$$

Next, we apply proposition 3 to $Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge g \geq 50$. Using the previous formula we obtain that for every legal action a in a state s:

$$
\begin{aligned}
& Enrolled(stu,c,do(a,s)) \wedge Grade(stu,c,g,do(a,s)) \wedge g \geq 50 \equiv \\
& \quad [\exists(x_1,x_2)(a = register(x_1,x_2) \wedge x_1 = stu \wedge x_2 = c \wedge \\
& \qquad\qquad Grade(stu,c,g,s) \wedge g \geq 50) \vee \\
& \quad \exists(y_1,y_2,y_3)(a = change(y_1,y_2,y_3) \wedge y_1 = stu \wedge y_2 = c \wedge \\
& \qquad\qquad y_3 = g \wedge Enrolled(stu,c,s) \wedge g \geq 50) \vee \\
& \qquad (Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge g \geq 50 \wedge \\
& \quad \neg(\exists(x_3,x_4)(a = drop(x_3,x_4) \wedge x_3 = stu \wedge x_4 = c) \vee \\
& \qquad \exists(y_4,y_5,y_6)(a = change(y_4,y_5,y_6) \wedge y_4 = stu \wedge y_5 = c \wedge y_6 \neq g)))].
\end{aligned}
$$

Finally, if we apply corollary 1 to $\exists g(Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge g \geq 50)$ in the previous formula, we obtain the following aeSSA for V_1:

$$
\begin{aligned}
& \forall(a,s) Poss(a,s) \supset \forall(stu,c)[V_1(stu,c,do(a,s)) \equiv \\
& \quad \exists(x_1,x_2)(a = register(x_1,x_2) \wedge \exists g(x_1 = stu \wedge x_2 = c \wedge \\
& \qquad\qquad Grade(stu,c,g,s) \wedge g \geq 50)) \vee \\
& \quad \exists(y_1,y_2,y_3)(a = change(y_1,y_2,y_3) \wedge \exists g(y_1 = stu \wedge y_2 = c \wedge \\
& \qquad\qquad y_3 = g \wedge Enrolled(stu,c,s) \wedge g \geq 50)) \vee \\
& \qquad V_1(stu,c,s) \wedge \\
& \quad \neg(\exists(x_3,x_4)(a = drop(x_3,x_4) \wedge \forall g((Enrolled(stu,c,s) \wedge \\
& \qquad\qquad Grade(stu,c,g,s) \wedge g \geq 50) \supset x_3 = stu \wedge x_4 = c) \vee \\
& \quad \exists(y_4,y_5,y_6)(a = change(y_4,y_5,y_6) \wedge \forall g((Enrolled(stu,c,s) \wedge \\
& \qquad\qquad Grade(stu,c,g,s) \wedge g \geq 50) \supset y_4 = stu \wedge y_5 = c \wedge y_6 \neq g) \wedge \\
& \quad \forall g(\neg(Enrolled(stu,c,s) \wedge Grade(stu,c,g,s) \wedge g \geq 50) \supset \\
& \qquad\qquad \neg(y_4 = stu \wedge y_5 = c \wedge y_6 = g)))))].
\end{aligned}
$$

$\square$

6 View Definitions as Integrity Constraints

In the previous section we considered the problem of obtaining suitable SSAs for views from the definition of the view. Nevertheless, the way in which we will use that derived information is as follows: Starting from the original specification, Σ, we will not consider explicit definitions of the views, but we will add to Σ the new SSAs plus definitions of the new virtual tables at the initial state. And then we hope that the right (and intended) logical relations between the views defined by SSAs and the base tables can be ensured along the database evolution.

This way of proceeding is consistent with Reiter's approach of not considering explicit state (integrity) constraints in the specifications [Re2]: State constraints should be entailed by the specification (see [LR1, Pi1] for more on this issue). Since view definitions can be seen as state constraints, the definitions should be implicit in the new specification.

More precisely, a static integrity constraint that is satisfied through the evolution of the database specified by Σ is a SC formula, $\varphi(s)$, that is simple in s, contains no action terms and no free variables for domain individuals, and $\Sigma \models \forall s(S_0 \leq s \supset \varphi(s))$. Since usual view definitions, i.e. of the form (7), could be seen as static ramifications constraints, we would expect to derive this explicit definition as an integrity constraint of the specification enriched with the new SSAs for the views.

Theorem 2. *Let $V(\bar{x}, s)$ be a view defined by a formula $\psi(\bar{x}, s)$, and let $(9)_\psi$, the axiom obtained from (9) by replacing ψ by V everywhere, be added to the specification of the database as the SSA for V. It holds*

$$\Sigma \cup \{(9)_\psi\} \cup \{\forall \bar{x}[V(\bar{x}, S_0) \equiv \psi(\bar{x}, S_0)]\} \models \forall s(S_0 \leq s \supset V(\bar{x}, s) \equiv \psi(\bar{x}, s)).$$

Formula $(9)_\psi$ tells us that the aeSSA for the view $V(\bar{x}, s)$ defined by the formula $\psi(\bar{x}, s)$ is:

$$\forall (a, s) Poss(a, s) \supset \forall \bar{x}[V(\bar{x}, do(a, s)) \equiv \xi^+(a, \bar{x}, s) \vee V(\bar{x}, s) \wedge \neg \xi^-(a, \bar{x}, s)], \quad (26)$$

where ξ^+ and ξ^- are constructed from the SSAs for the base tables appearing in ψ as shown in section 5. For following applications, we stress the fact that formulas $\xi^+(a, \bar{x}, s)$ and $\xi^-(a, \bar{x}, s)$ contain only base tables (or other previously defined views) evaluated at the execution state s, (in)equalities between domain individuals, and equalities between the action variable a and instantiations of actions names existing in the original language. By the $UNAA$, any instantiation of a in those formulas by means of an action name will make all the (in)equalities between actions disappear.

The theorem establishes that the intended logical condition for a view, actually its explicit definition, is a state constraint that is implicit in the original specification of the database dynamics extended with the new SSAs. As such, it holds at all legal states of the database. This can be proved by induction on states. For this purpose, the following derived induction principle [Re3] can be applied

$$\forall P([P(S_0) \ \wedge \ \forall s \forall a(P(s) \wedge Poss(a,s) \ \supset \ P(do(a,s))] \ \supset \ \forall s(S_0 \leq s \ \supset P(s))).$$

7 View Maintenance Supported by SSAs

We could materialize the views by keeping extensional physical tables containing the entries that satisfy the view definitions. View maintenance is the problem of updating such a materialized views [GS1].

From our derived specification of the evolution of a database view V in terms of a aeSSA of the form (26), we can produce an algorithm for the automated maintenance of V. First we start with a materialization of the view at the initial state, that is, using the view definition and the initial base tables. Now, the aeSSA for V explicitly shows which are the primitive transactions that may affect the view and in what conditions. Then, in the presence of a given primitive transaction that updates some of the base tables, the algorithm can check whether the transaction appears among the candidate transactions to change the view, what can be read from the SSAs, and, in the positive case, whether the associated conditions are met, in which case the update in the view is computed as indicated by the SSA.

This algorithm reacts to the *transactions* updating the base tables, rather than to the *changes* in the base tables. Actually, changes in the base tables are not considered, but only the transactions, the base tables, and the view itself at the previous (execution) state. In that sense this is not an incremental view maintenance algorithm [GS1].

7.1 An Example from Relational Algebra

We will illustrate with an example from relational algebra how to apply the results obtained in section 5 to the generation of procedures for view maintenance.

It is possible to rewrite every relational expression using first order logic. Actually, it can be seen as a view defined in the SC. Thus, we can obtain an aeSSA for this new virtual table if we have aeSSAs for the base tables. In the most simple cases of relational databases we are allowed to insert and remove tuples from base tables. Then such a table, P, will have an aeSSA mentioning $insert_P$ and $delete_P$ as primitive transactions:

$$\begin{aligned}\forall(a,s)Poss(a,s) \supset \forall\bar{x}[P(\bar{x},do(a,s)) \equiv \exists\bar{u}(a = insert_P(\bar{u}) \wedge \bar{u} = \bar{x}) \vee \\ P(\bar{x},s) \wedge \neg\exists\bar{v}(a = delete_P(\bar{v}) \wedge \bar{v} = \bar{x})].\end{aligned} \tag{27}$$

Now, suppose a view is defined using the relational expression $P(\bar{x}) \bowtie_{\theta(\bar{x},\bar{y})} P(\bar{y})$, where $\bar{x}$ and $\bar{y}$ do not have variables in common. By the results obtained in section 5, we obtain the following aeSSA for the new table $(P \bowtie_\theta P)(\bar{x}, \bar{y}, s)$:

$$
\begin{aligned}
&\forall(a,s) Poss(a,s) \supset \forall(\bar{x},\bar{y})[(P \bowtie_\theta P)(\bar{x},\bar{y},do(a,s)) \equiv \\
&\qquad \exists \bar{u}(a = insert_P(\bar{u}) \wedge [(\bar{u} = \bar{x} \wedge \bar{u} = \bar{y} \wedge \theta(\bar{x},\bar{y})) \vee \\
&\qquad (\bar{u} = \bar{x} \wedge P(\bar{y},s) \wedge \theta(\bar{x},\bar{y})) \vee (\bar{u} = \bar{y} \wedge P(\bar{x},s) \wedge \theta(\bar{x},\bar{y}))]) \vee \\
&\qquad (P \bowtie_\theta P)(\bar{x},\bar{y},s) \wedge \neg \exists \bar{v}(a = delete_P(\bar{v}) \wedge (\bar{v} = \bar{x} \vee \bar{v} = \bar{y}))].
\end{aligned}
$$

From this axiom it is possible to derive a maintenance procedure for the new table. It is shown in figure 2. Notice that the translation of the aeSSA for $(P \bowtie_\theta P)$ into the maintenance procedure is direct and uses much information taken from the different components of the SSA. For example, it shows which base tables are necessary to consider for updating $(P \bowtie_\theta P)$ when we add a tuple or when we remove a tuple from a base table. The procedure (and the SSA) show that the table $(P \bowtie_\theta P)$ is *self-maintainable* [GS1] with respect to deletions.

Clearly, the language of situation calculus allow us to consider more interesting primitive transactions than the ones in this example, for example, $update_P(\bar{x}, \bar{y})$, that is "update $\bar{x}$ to $\bar{y}$ in table P", or more complex primitive transactions specified by the user; and also more complex view definition constructs.

procedure Maintenance(A:action)
begin
    if A= $insert_P(\bar{w})$ **then**
        if $\theta(\bar{w}, \bar{w})$ holds **then**
            Add the tuple $(\bar{w}, \bar{w})$ to $(P \bowtie_\theta P)$;
        for each tuple $\bar{y}$ such that $P(\bar{y}) \wedge \theta(\bar{w}, \bar{y})$ holds **do**
            Add the tuple $(\bar{w}, \bar{y})$ to $(P \bowtie_\theta P)$;
        for each tuple $\bar{x}$ such that $P(\bar{x}) \wedge \theta(\bar{x}, \bar{w})$ holds **do**
            Add the tuple$(\bar{x}, \bar{w})$ to $(P \bowtie_\theta P)$;
    else if $A = delete_P(\bar{w})$ **then**
        Remove each tuple $(\bar{x}, \bar{w})$ and $(\bar{w}, \bar{y})$ from $(P \bowtie_\theta P)$;
end

Fig. 2. Procedure for maintaining the view $(P \bowtie_\theta P)$.

8 Checking and Proving Integrity Constraints

One interesting relationship between views and integrity constraints has to do with using view maintenance to check integrity constraint satisfaction (see references in [GS1]). We will illustrate how this problem connects with our treatment of views by considering the case of a static integrity constraint of the form $\forall \bar{x} \varphi(\bar{x}, s)$, where φ is a formula simple in s and quantifier free. For example, a functional dependency is of this form. Now, let us define a view by

$\forall \bar{x}[V(\bar{x}, s) \equiv \neg\varphi(\bar{x}, s)]$. We expect $V(\bar{x}, s)$ to be always an empty table, i.e. $V(s) = \emptyset$. Using our previous results, we can derive an aeSSA for V of the form (26). Let us denote it by SSA_V. Then it holds $\Sigma \models \forall s(S_0 \leq s \supset \forall \bar{x}\varphi(\bar{x}, s))$ iff $\Sigma \cup \{SSA_V, \forall \bar{x}[V(\bar{x}, S_0) \equiv \neg\varphi(\bar{x}, S_0)]\} \models \forall s(S_0 \leq s \supset \neg\exists \bar{x} V(\bar{x}, s))$.

It is clear that we can *check* the IC by checking whether V is empty, but we can also *prove* that the IC holds at every accessible state by proving that $V(s)$ is empty at every such state. This second case will be possible iff the original specification Σ entails the IC. In the case of IC checking, even if the IC is not entailed by the specification, we can be interested in checking it when concrete transactions are executed. For both the IC checking problem and the IC proving problem, we assume the the IC is satisfied at the execution state. This corresponds in the IC proving case to a proof by induction on states for arbitrary transactions supported by V's SSA: If this is (26), then in the inductive step we can assume that $V(s)$ is empty, so in order to check $V(\bar{x}, do(a, s))$, we need to check that $\xi^+(a, \bar{x}, s)$ in (26) is empty., i.e. no $\bar{x}$ makes it true. Because of the form of ξ^+ we can do case analysis on the (finite number of) transaction names appearing in it, verifying that for each of them we do not obtain any tuples $\bar{x}$ satisfying ξ^+. In the case of a successful IC proof the $\xi^+(a, \bar{x}, s)$ will be identically false for any possible action a. In the case of a successful IC checking step, it will become false of the particular executed transaction.

8.1 The Case of Functional Dependencies

We will consider the particular case of a functional dependency of the form[9]

$$\forall(x, y, z, s)[S_0 \leq s \supset (F(x, y, s) \wedge F(x, z, s) \supset y = z)]. \tag{28}$$

The problem of proving it reduces to the problem of proving that the view

$$\forall(x, y, z, s)[V(x, y, z, s) \equiv_{def} F(x, y, s) \wedge F(x, z, s) \wedge y \neq z] \tag{29}$$

is empty at all accessible states. That is, if V has a SSA of the form:

$$\begin{aligned} Poss(a, s) \supset \forall(x, y, z)[V(x, y, z, do(a, s)) \equiv \xi^+(a, x, y, z, s) \vee \\ V(x, y, z, s) \wedge \neg\xi^-(a, x, y, z, s)], \end{aligned}$$

then all we need is to prove that if $Poss(a, s)$ holds, then $\xi^+(a, x, y, z, s)$ is identically false. With the results of the previous sections, we can obtain such a SSA for V: If table F's SSA is

$$Poss(a, s) \supset \forall(x, y)[F(x, y, do(a, s)) \equiv \gamma^+(a, x, y, s) \vee (F(x, y, s) \wedge \neg\gamma^-(a, x, y, s)),$$

[9] We are grateful to Pablo Saez for helping us with this section.

then the expression $\xi^+(a,x,y,z,s)$ in V's SSA is:

$$
\begin{gathered}
[\bigvee_{A\in Acc(\gamma^+)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,x,z,s)) \\
\vee \\
\bigvee_{A\in Acc(\gamma^+)\cap Acc(\gamma^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge ((\varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \neg\varphi_A^{\gamma^-}(\bar{x}_A,x,z,s) \wedge \\
F(x,z,s)) \vee (\varphi_A^{\gamma^+}(\bar{x}_A,x,z,s) \wedge \neg\varphi_A^{\gamma^-}(\bar{x}_A,x,y,s) \wedge F(x,y,s)))) \\
\vee \\
\bigvee_{A\in Acc(\gamma^+)-Acc(\gamma^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge ((\varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge F(x,z,s)) \vee \\
(\varphi_A^{\gamma^+}(\bar{x}_A,x,z,s) \wedge F(x,y,s))))] \wedge y \neq z.
\end{gathered}
$$

We can prove that this expression is identically false by case analysis, that is, by instantiating the expression with every action symbol $A(\bar{x}_A)$ (in a generic way), and then applying $UNAA$. In this way, and assuming that $A \in Acc(\gamma^+) \cap Acc(\gamma^-)$, we get the following expression for $\xi^+(A(\bar{x}_A),x,y,z,s)$:

$$
\begin{aligned}
&[\varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,x,z,s) \vee \\
&\ \varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \neg\varphi_A^{\gamma^-}(\bar{x}_A,x,z,s) \wedge F(x,z,s) \vee \\
&\ \varphi_A^{\gamma^+}(\bar{x}_A,x,z,s) \wedge \neg\varphi_A^{\gamma^-}(\bar{x}_A,x,y,s) \wedge F(x,y,s))] \wedge y \neq z.
\end{aligned} \tag{30}
$$

If instead $A \in Acc(\gamma^+) - Acc(\gamma^-)$, we can make $\varphi_A^{\gamma^-}(\bar{x}_A,x,z,s)$ identically false, getting the same result as (30).

We need to prove that (30) is identically false, that is:

$$\varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,x,z,s) \supset y = z \tag{31}$$

and:

$$\varphi_A^{\gamma^+}(\bar{x}_A,x,y,s) \wedge \neg\varphi_A^{\gamma^-}(\bar{x}_A,x,z,s) \wedge F(x,z,s) \supset y = z. \tag{32}$$

A third subgoal should be equivalent to the second one.

We can conclude that, in order to prove (28), we can prove (31) and (32) for every action symbol A in $Acc(\gamma^+)$, assuming the hypothesis $Poss(A(\bar{x}_A), s)$. This can be carried out if we have a concrete SSA for F and concrete precondition axioms.

Example 8. We will illustrate the previous ideas with an example of a database with a table of objects with their colors. Here $Color(x,y,s)$ means that object x has color y at the state s. We also have a primitive transaction $paint(x,y)$, that modifies x's color to y. Assume that the following effect axioms are available (or can be recovered from already existing aeSSAs):

$$
\begin{gathered}
\forall(x,y,s)[Poss(paint(x,y),s) \supset Color(x,y,do(paint(x,y),s))], \\
\forall(x,y,y',s)[Poss(paint(x,y'),s) \wedge y \neq y' \supset \neg Color(x,y,do(paint(x,y'),s))].
\end{gathered}
$$

That is, we have the following aeSSA for *Color*:

$$\forall(a,s) Poss(a,s) \supset \forall(x,y)[Color(x,y,do(a,s)) \equiv \\ \exists(u,v)(a = paint(u,v) \wedge u = x \wedge v = y) \vee \\ Color(x,y,s) \wedge \neg\exists(u,v)(a = paint(u,v) \wedge u = x \wedge v \neq y)].$$

Let us further assume that it is always possible to execute action *paint*:

$$\forall(x,y,s)[Poss(paint(x,y),s) \equiv \mathit{True}]. \tag{33}$$

We expect to satisfy the constraint stating that each object has only one color in every state:

$$\forall s[S_0 \leq s \supset \forall(x,y,z)\ (Color(x,y,s) \wedge Color(x,z,s) \supset y = z)].$$

To prove this integrity constraint we instantiate (31) and (32) using the aeSSA for *Color*, obtaining:

$$(u = x \wedge v = y \wedge u = x \wedge v = z) \supset y = z, \tag{34}$$

$$(u = x \wedge v = y \wedge \neg(u = x \wedge v \neq z) \wedge Color(x,y,s)) \supset y = z. \tag{35}$$

Thus, we need to prove that the specification entails:

$$(u = x \wedge v = y \wedge v = z) \supset y = z, \tag{36}$$

$$(u = x \wedge v = y \wedge v = z \wedge Color(x,y,s)) \supset y = z, \tag{37}$$

what is straightforward. □

The kind of analysis we have presented for proving/checking ICs is reminiscent of Nicolas' methodology for checking an IC on the assumption that the IC is already satisfied before the transaction execution [Ni1], what allows to simplify the condition to be checked. A detailed analysis of Nicolas' methodology in the context of our specifications of database updates and a proof that its reconstruction in the SC setting coincides with the methodology presented in this section can be found in [Sa1]. Direct methodologies for proving ICs by induction in our SC scenario which have similar characteristics are reported in [Be2, Sa1].

9 Embedding Integrity Constraints

Both [LR1] and [Pi1] consider the problem of embedding a desirable static IC, say $\forall s\ (S_0 \leq s \supset \psi(s))$, where $\psi(s)$ has no free variables for individuals, into the given specification in such a way that it does not appear explicitly in the specification, but implicitly in the sense that it logically follows from the specification. The new specification should still have the form indicated at the end of section 2.

The IC can be solved à la qualification, that is, action preconditions are made stronger, and this is always possible as shown in [LR1]. ICs can also be solved à la ramification, that is by modifying the effect axioms that are implicit but traceable in a specification based on aeSSAs. There is no general solution for this option, but some syntactic cases of ICs, that is of $\psi(s)$ above, e.g. functional dependencies, are solved in [Pi1].

We can show that our treatment of views can be used to give an easy and general alternative solution to the problem of embedding an IC á la qualification. This is done by defining a "propositional" view $\psi(s)$. If the specification Σ is such that all base tables have aeSSAs, then it is possible to derive an aeSSA for $\psi(s)$:

$$\begin{aligned}\forall(a,s)Poss(a,s) \supset [\psi(do(a,s)) \equiv \bigvee_A \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, s))\\ \vee\ \ \psi(s)\ \ \wedge \neg \bigvee_B \exists \bar{x}_B(a = B(\bar{x}_B) \wedge \varphi_B^{\gamma^-}(\bar{x}_B, s))].\end{aligned}$$

Since we want $\psi(s)$ to true for all accessible states, we have to further constraint the actions that can make it false. Then, for each action B mentioned in en γ^-, we change its precondition axiom from

$$\forall(\bar{x}_B, s)[Poss(B(\bar{x}_B), s) \equiv \Pi_B(\bar{x}_B, s)].$$

to

$$\forall(\bar{x}_B, s)[Poss(B(\bar{x}_B), s) \equiv \Pi_B(\bar{x}_B, s) \wedge \neg\varphi_B^{\gamma^-}(\bar{x}_B, s)].$$

In this way we generate a new specification Σ', which entails $\forall s[S_0 \leq s \supset \psi(s)]$.

Example 9. Let us consider the specification given in the example 8. Suppose that we expect to satisfy the constraint saying that two different objects cannot have the same color:

$$\forall s[S_0 \leq s \supset \forall(x_1, x_2, y)\ (Color(x_1, y, s) \wedge Color(x_2, y, s) \supset x_1 = x_2)].$$

In order to embed this IC in the specification, we generate the following view:

$$\forall s[V(s) \equiv \forall(x_1, x_2, y)(Color(x_1, y, s) \wedge Color(x_2, y, s) \supset x_1 = x_2)],$$

which, by the previous results, turns out to have the following SSA:

$$\forall(a, s)Poss(a, s) \supset V(do(a, s)) \equiv \gamma^+(a, s) \vee (V(s) \wedge \neg\gamma^-(a, s)),$$

where

$$\begin{aligned}\gamma^-(a, s) = \ \neg\exists(u, v)[a = paint(u, v) \wedge \exists(x_1, x_2, y)(v = y \wedge x_1 \neq x_2 \wedge\\ ((u = x_1 \wedge Color(x_2, y, s)) \vee (u = x_2 \wedge Color(x_1, y, s)))].\end{aligned}$$

Finally, considering the γ^- part of this SSA for the view and formula (33), we obtain the following new precondition axiom for the primitive transaction *paint*:

$$\begin{aligned}\forall(u, v, s)[Poss(paint(u, v), s) \equiv \neg\exists(x_1, x_2, y)(v = y \wedge x_1 \neq x_2 \wedge\\ ((u = x_1 \wedge Color(x_2, y, s)) \vee (u = x_2 \wedge Color(x_1, y, s))))].\end{aligned}$$

□

For embedding ICs of the form $\forall s(S_0 \leq s \supset \forall \bar{x}\varphi(\bar{x}, s))$, like functional dependencies, an alternative methodology could be based on further constraining the actions that may introduce tuples into the hopefully empty view $V(\bar{x}, s) \equiv_{df} \neg\varphi(\bar{x}, s)$, as in section 8. In the extended version of this paper we compare our methodology with that of Lin and Reiter.

10 Conclusions, Related Issues and Outlook

1. We have managed to derive action–effect based successor state axioms for database views from similar axioms for the base tables. Having such SSAs has several advantages: (1) We can update the view without appealing to the changes in other tables. (2) If we compute the new extension of a view from the other tables using only its definition, we are not really *updating* explicitly the view, but *recomputing* the whole contents of the table. And, if we do not want to proceed this way, we need to derive algorithms for computing differences of tables, including the view [GS1]. The approach based on aeSSAs does not have these problems. (3) All this becomes more complicated if the base tables are only virtually updated, or only the update transaction logs are kept. This situation appear frequently in applications of Reiter's formalism [AB1]. (4) Having an explicit action–effect based SSA for a view makes a *materialization* of the view as a physical table easy. We might be interested in doing this for the usual reasons in databases, and also for *rolling forward* [LR2] the virtually updated database. As said before, the materialization does not use the updated base tables, but the tables at the previous state. So, this is a form of *deferred* materialization. (5) Having such a SSA available makes possible doing hypothetical reasoning about the view. In general, all the possibilities of reasoning based on SSAs that are open to the base tables become available for the views as well [AB1]. (6) It might be the case that, for security reasons or by limitations of the query language [Im1], a user has a restricted access to the database through particular views. So, without seeing the whole database, that user would be interested in knowing how transactions affect his/her particular views and in reasoning about their changes. This becomes possible with appropriate SSAs for the views. (7) At the knowledge representation level, we have an explicit solution to the frame problem for views and we know exactly how this solution is inherited from the solution for the base tables.

2. We proved that the derived SSAs satisfy a natural consistency condition if the SSAs for the base tables do.

3. We established that adding the right SSAs for the views to the database specification allows us to obtain the intended and explicit view definitions as static integrity constraints of the database.

4. We have seen how to produce an algorithm for view maintenance that relies on the derived aeSSA for the view, the executed transactions and the contents

of the database at the execution state. We also related the problem of view updates with the problems of integrity constraint checking and proving. We also gave a new solution to the problem of embedding ICs into the specification.

5. There are several open problems related to the subjects in this paper that have not been addressed in this paper and should be subjects of further research. Some of them are the following:

(1) We have not considered the problem of generating aeSSA for recursive views. To consider an example, we cannot expect the transitive closure TC_R of a relation R to have a aeSSA of the form (26). If this were the case, then we could start with an empty relation R at the initial state, an empty TC_R, and execute at S_0 an action A that populates the table R. In this case, we would have $\forall \bar{x}(TC_R(\bar{x}, do(A, S_0)) \equiv \xi^+(A, \bar{x}, S_0)$ (because TC_R is empty at S_0). By the $UNAA$, A disappears from the RHS, and becomes a formula in terms of base tables, i.e. R, (and S_0) only, obtaining a first order definition of the transitive closure, what we know is impossible [GV1].

(2) In this paper, views are explicitly defined by means of arbitrary first order SC formulas. We have not considered the issue of *safeness* of the views definitions [Ul1, VT1]. From the pure logical point of view, all the propositions we gave wrt to derivation of aeSSAs are correct, but some of them, e.g. proposition 1, consider cases that may lead to unsafe view definitions in databases. Some syntactical subclasses of safe views definitions should be treated in more detail from the general propositions we provided.

(3) Aggregate views and other complex views appearing in data warehousing and on–line analytical processing (OLAP) [CD1, Zh1] should also be treated. Also views encoding historical information [Ch1, AB1].

(4) It would be interesting to relate our treatment of views with some work done on answering queries in terms of database views, as in [Le1, Ra1].

(5) More importantly, we have the impression that our work could have some interesting consequences on the problem of updating a database through its views [FC1]. There is at least one case in which we know how to update the base tables when a view is updated by means of one of the primitive transactions appearing in its derived aeSSA: All we need to do is to detect which base tables that action effects and how, by reading their aeSSAs. So, the other case appears when we add to a view's aeSSA new actions that may change a view and we have to propagate the occurrences of those actions to the aeSSAs for the base tables, while taking into account the view definition as an integrity constraint.

(6) Also the relationship between our approach and the problem of determining updates that are relevant to a view [Bl1, Be1, SB1] should be explored.

(7) At the knowledge representation level, it would be interesting to explore the derivation, with a methodology as presented in [LR1], of a new SSAs based specification from the explicit definitions of the views, seen as static ramification constraints. It is not clear that a solution to the problem of deriving aeSSAs for views can be found in this way, because there is no

general solution so far for the problem of embedding ICs in the specification via ramification. Along this line, [LR1] already contains an example about the transitive closure considered as a ramification constraint.

Acknowledgments

This research has been partially supported by FONDECYT (Grants # 1971304 and # 1980945). Part of this work was done while the second author was on sabbatical at the Technical University of Berlin. He is grateful to Ralf Kutsche and all the people in the "Computergestüzte Informationssysteme" (CIS) group for their support and hospitality; and to the Graduierten Kolleg "Verteilte Informationssysteme", the DAAD and the Catholic University of Chile (DIPUC) for their financial support. The authors are grateful to Javier Pinto and Jorge Baier for their generous help.

References

[AB1] Arenas, M., Bertossi, L.: Hypothetical temporal queries in databases. In *Proc. of the ACM SIGMOD/PODS 5th Workshop on Knowledge Representation meets databases (KRDB'98)*, Borgida, A., Chaudhuri, V., Staudt, M. (eds.) (1998) (http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-10/).

[Be1] Bertossi, L., Arenas, M., Ferretti, C.: SCDBR: An automated reasoner for specifications of database updates. Journal of Intelligent Information Systems **10(3)** (1998).

[Be2] Bertossi, L., Pinto, J., Saez, P., Kapur, D., Subramaniam, M.: Automating Proofs of Integrity Constraints in the Situation Calculus. In *Foundations of Intelligent Systems. Proc. Ninth International Symposium on Methodologies for Intelligent Systems (ISMIS'96).* Springer LNAI 1079 (1996) 212–222.

[Bl1] Blakeley, J., Coburn, N., Larson, P.: Updating derived relations: Detecting irrelevant and autonomously computable updates". ACM Transactions on Database Systems **14** (1989) 369–400.

[CD1] Chaudhuri, S., Dayal, U.: An overview of datawarehousing and OLAP technology. ACM SIGMOD Record **26** (1997) 65–74.

[Ch1] Chomicki, J.: Efficient checking of temporal integrity constraints using bounded history encoding. ACM Transactions on Database Systems **20** (1995) 149–186.

[Cl1] Clark, K.: Negation as failure. In Gallaire, H., Minker, J. (eds.) *Logic and Data Bases.* Plenum Press (1978) 292–322.

[FC1] Furtado, A., Casanova, M.: Updating relational views. In Kim, W., Reiner, D., Batory, D. (eds.) *Query Processing in Database Systems.* Springer (1985).

[GV1] Gaifman, H., Vardi, M.: A simple proof that connectivity of finite graphs is not first-order definable. Bulletin EATCS **26** (1985) 44–45.

[Go1] Godfrey, P., Grant, J., Gryz, J., Minker, J.: Integrity constraints: Semantics and applications. In Chomicki, J., Saake, G. (eds.) *Logics for Databases and Information Systems.* Kluwer (1998).

[GS1] Gupta, A., Singh Mumick, I.: Maintenance of materialized views: Problems, techniques, and applications. IEEE-CS Data Engineering Bulletin **18** (1995) 3–18. Special issue on materialized views and data warehousing.

[Im1] Imielinski, T.: Relative knowledge in distributed database (extended abstract). In *Proc. Symposium on Principles of Database Systems (PODS'87)*, ACM Press (1987) 197-209.

[Le1] Levy, A., Mendelzon, A., Sagiv, Y., Srivastava, D.: Answering queries using views. In *Proc. Symposium on Principles of Database Systems (PODS'95)*, ACM Press (1995) 95–104.

[LR1] Lin, F., Reiter, R.: State constraints revisited. Journal of Logic and Computation **4** (1994) 655–678. Special issue on action and processes.

[LR2] Lin, F., Reiter, R.: How to progress a database. Artificial Intelligence **92** (1995) 131–167.

[MH1] McCarthy, J., Hayes, P.: Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie (eds.), *Machine Intelligence* **4**, Edinburgh University Press (1969) 463–502.

[Ni1] Nicolas, J-M.: Logic for improving integrity checking in relational data bases. Acta Informatica **18** (1982) 227–253.

[Pi1] Pinto, J.: Temporal reasoning in the situational calculus. PhD thesis, Department of Computer Science, University of Toronto (1994).

[Ra1] Rajaraman, A., Sagiv, Y., Ullman, J.: Answering queries using templates with binding patterns. In *Proc. Symposium on Principles of Database Systems (PODS'95)*, ACM Press (1995) 105–112.

[Re1] Reiter, R.: The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, V. (ed.) *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, Academic Press (1991) 359–380.

[Re2] Reiter, R.: On specifying database updates. Journal of Logic Programming **25** (1995) 53–91.

[Re3] Reiter, R.: Proving properties of states in the situation calculus. Artificial Intelligence **64** (1993) 337–351.

[Sa1] Saez, P.: Automated proofs of database integrity constraints. PhD thesis, Catholic University of Chile, School of Engineering, Department of Computer Science. In preparation.

[SB1] Siu, B., Bertossi, L.: Answering historical queries in databases (extended abstract). In Zelkowitz, M., Straub, P. (eds.) *Proc. XVI International Conference of the Chilean Computer Science Society (SCCC'96)* (1996).

[Ul1] Ullman, J.: Principles of database and knowledge-base systems, Vol. I. Computer Science Press, 1988.

[VT1] Van Gelder, A., Topor, R.: Safety and correct translation of relational calculus formulas. In *Proc. ACM Symposium on Database Systems (PODS)*, ACM Press (1987) 313–327.

[Zh1] Zhuge, Y., Garcia-Molina, H., Widom, J.: View maintenance in a datawarehousing environment. In *Proc. Symposium on Principles of Database Systems (PODS'96)*, ACM Press (1996) 316–327.

A Some Proofs and Intermediate Results

In order not to overload the formulas we will adopt some natural conventions, e.g. instead of writing

$$\gamma^+(a,\bar{x},s) = \bigvee_{A \in Acc(\gamma^+)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)),$$

we will simply write

$$\gamma^+(a,\bar{x},s) = \bigvee_{A} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)).$$

We will usually denote with $\mathcal{M}$ a model of the situation calculus. $|\mathcal{M}|$ will denote the universe of structure $\mathcal{M}$. Then, in this many sorted universe we will find domain individuals, action and states.

Lemma 1. *If*

$$\gamma^+(a,\bar{x},s) = \bigvee_{A} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)) \tag{38}$$

$$\delta^+(a,\bar{y},s) = \bigvee_{B} \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^+}(\bar{y}_B,\bar{y},s)), \tag{39}$$

where for the action names A and B, $\bar{x}_A$ and $\bar{y}_B$ do not share variables, then

$$UNAA \models \forall(a,\bar{x},\bar{y},s)[\gamma^+(a,\bar{x},s) \wedge \delta^+(a,\bar{y},s) \equiv \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^+)} \exists \bar{x}_A \; (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s) \wedge \varphi_A^{\delta^+}(\bar{x}_A,\bar{y},s))]$$

Proof: Let $\mathcal{M} \models UNAA$, and $C \in |\mathcal{M}|$ be an action, $\bar{e}$ an l-tuple of objects in $|\mathcal{M}|$, where l is the number of arguments of action C. Let S be a state in $|\mathcal{M}|$, $\bar{b}$ an r-tuple of objects in $|\mathcal{M}|$, where r is the number of variables in $\bar{x}$, $\bar{d}$ a u-tuple of objects in $|\mathcal{M}|$, where u is the number of de variables in $\bar{y}$. Finally, let $\sigma: \{\frac{C(\bar{e})}{a}, \frac{\bar{b}}{\bar{x}}, \frac{\bar{d}}{\bar{y}}, \frac{S}{s}\}$, be an assignment for the variables.

By definition of γ^+ and δ^+ one has that:

$$\langle \mathcal{M},\sigma\rangle \models \gamma^+(a,\bar{x},s) \wedge \delta^+(a,\bar{y},s) \equiv \bigvee_{A,B} [\exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A,\bar{x},s)) \wedge \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^+}(\bar{y}_B,\bar{y},s))]. \tag{40}$$

As $\bar{x}_A$ and $\bar{y}_B$ do not share variables, in the RHS of the previous formula we can leave those quantifiers outside, obtaining the equivalent formula:

$$\bigvee_{A,B} \exists \bar{x}_A \exists \bar{y}_B \ (a = A(\bar{x}_A) \wedge a = B(\bar{y}_B) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_B^{\delta^+}(\bar{y}_B, \bar{y}, s)). \quad (41)$$

But this formula can be rewritten as

$$\begin{aligned} &\bigvee_{A=B} \exists \bar{x}_A \exists \bar{y}_B \ (a = A(\bar{x}_A) \wedge a = B(\bar{y}_B) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_B^{\delta^+}(\bar{y}_B, \bar{y}, s)) \vee \\ &\bigvee_{A\neq B} \exists \bar{x}_A \exists \bar{y}_B \ (a = A(\bar{x}_A) \wedge a = B(\bar{y}_B) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_B^{\delta^+}(\bar{y}_B, \bar{y}, s)). \end{aligned} \quad (42)$$

We know that if $A \neq B$, where A and B are action names in the language, then $UNAA \models \forall(\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) \neq B(\bar{y}_B)]$, thus $\mathcal{M} \models \forall(\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) \neq B(\bar{y}_B)]$, and then

$$\begin{aligned} \langle \mathcal{M}, \sigma \rangle \models \neg \bigvee_{A\neq B} \exists \bar{x}_A \exists \bar{y}_B \ (a = A(\bar{x}_A) \wedge \\ a = B(\bar{y}_B) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_B^{\delta^+}(\bar{y}_B, \bar{y}, s)). \end{aligned} \quad (43)$$

From (40), (41), (42), (43) and considering that the condition $A{=}B$ is equivalent to the condition $A \in Acc(\gamma^+) \cap Acc(\delta^+)$, we conclude that

$$\begin{aligned} \langle \mathcal{M}, \sigma \rangle \models \gamma^+(a, \bar{x}, s) \wedge \delta^+(a, \bar{y}, s) \equiv \bigvee_{A \in Acc(\gamma^+) \cap Acc(\gamma^-)} \exists \bar{x}_A \exists \bar{y}_A \ (a = A(\bar{x}_A) \wedge \\ \wedge\, a = A(\bar{y}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge \varphi_A^{\delta^+}(\bar{y}_A, \bar{y}, s)). \end{aligned} \quad (44)$$

As A is an action name in the language, we have that $UNAA \models \forall(\bar{x}_A, \bar{y}_A)[A(\bar{x}_A) = A(\bar{y}_A) \supset \bar{x}_A = \bar{y}_A]$, thus $\mathcal{M} \models \forall(\bar{x}_A, \bar{y}_A)[A(\bar{x}_A) = A(\bar{y}_A) \supset \bar{x}_A = \bar{y}_A]$, and then in the model $\langle \mathcal{M}, \sigma \rangle$ the formula on the RHS of (44) is equivalent to:

$$\bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^+)} \exists \bar{x}_A \ (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s)),$$

from this we conclude the lemma.

Lemma 2. *If*

$$\gamma^+(a, \bar{x}, s) = \bigvee_A \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \quad (45)$$

$$\delta^-(a, \bar{y}, s) = \bigvee_B \exists \bar{y}_B (a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)), \quad (46)$$

where every action in $Acc(\gamma^+) \cup Acc(\delta^-)$ is mentioned in UNAA and $Q(\bar{y}, s)$ is a fluent, then from UNAA it is possible to conclude:

$$\begin{aligned}&\forall(a, \bar{x}, \bar{y}, s)[(\gamma^+(a, \bar{x}, s) \wedge Q(\bar{y}, s) \wedge \neg\delta^-(a, \bar{y}, s)) \equiv \\ &\bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)) \vee \\ &\bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s))]\end{aligned}$$

Proof: Let $\mathcal{M} \models UNAA$, and $C \in |\mathcal{M}|$ be an action, $\bar{e}$ an l-tuple of objects in $|\mathcal{M}|$, where l is the number of arguments of action C. Let S be a state in $|\mathcal{M}|$, $\bar{b}$ an r-tuple of objects in $|\mathcal{M}|$, where r is the number of variables in $\bar{x}$, $\bar{d}$ an u-tuple of objects in $|\mathcal{M}|$, where u is the number of variables in $\bar{y}$. Finally, let $\sigma : \{\frac{C(\bar{e})}{a}, \frac{\bar{b}}{\bar{x}}, \frac{\bar{d}}{\bar{y}}, \frac{S}{s}\}$, be the assignment for variables.

By definition of γ^+ and δ^- we have:

$$\begin{aligned}\langle\mathcal{M}, \sigma\rangle \models \gamma^+(a, \bar{x}, s) \wedge Q(\bar{y}, s) \wedge \neg\delta^-(a, \bar{y}, s) \equiv \\ \bigvee_A \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ \bigwedge_B \neg\exists\bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)).\end{aligned} \tag{47}$$

Applying distribution in the RHS of the equivalence above, and considering that $Acc(\gamma^+) = (Acc(\gamma^+) - Acc(\delta^-)) \cup (Acc(\gamma^+) \cap Acc(\delta^-))$, we obtain:

$$\begin{aligned}\langle\mathcal{M}, \sigma\rangle \models &\gamma^+(a, \bar{x}, s) \wedge Q(\bar{y}, s) \wedge \neg\delta^-(a, \bar{y}, s) \equiv \\ &\bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} [\exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ &\bigwedge_B \neg\exists\bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))] \vee \\ &\bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} [\exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ &\bigwedge_B \neg\exists\bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))].\end{aligned} \tag{48}$$

In consequence, in order to prove the lemma, we need to consider some cases:

(I) Let $A \in Acc(\gamma^+) - Acc(\delta^-)$. Assume that $\langle\mathcal{M}, \sigma\rangle \models \exists\bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s))$. Since $A \notin Acc(\delta^-)$ and $\mathcal{M} \models UNAA$, we have that for each index $B \in Acc(\delta^-)$, $\mathcal{M} \models \forall(\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) \neq B(\bar{y}_B)]$, and then: $\langle\mathcal{M}, \sigma\rangle \models \bigwedge_B \neg\exists\bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))$, Thus, is possible to conclude that for every action symbol $A \in Acc(\gamma^+) - Acc(\delta^-)$ that

$$\begin{aligned}\langle \mathcal{M}, \sigma\rangle \models\ & \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)) \wedge \\ & \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)) \wedge \\ & \qquad \bigwedge_B \neg \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)),\end{aligned}$$

and then

$$\begin{aligned}\langle \mathcal{M}, \sigma\rangle \models & \bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)) \supset \\ & \bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} [\exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ & \bigwedge_B \neg \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))].\end{aligned}$$

But, $\alpha \wedge \beta \supset \alpha$ is a logical truth, so we conclude:

$$\begin{aligned}\langle \mathcal{M}, \sigma\rangle \models & \bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)) \equiv \\ & \bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} [\exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ & \bigwedge_B \neg \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))].\end{aligned} \tag{49}$$

(II) Let $A \in Acc(\gamma^+) \cap Acc(\delta^-)$. In this case we will prove that $\langle \mathcal{M}, \sigma\rangle$ satisfies

$$\begin{aligned}& \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s)) \equiv \\ & \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} [\exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \wedge Q(\bar{y}, s) \wedge \\ & \qquad \qquad \bigwedge_B \neg \exists \bar{y}_B(a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))].\end{aligned} \tag{50}$$

For this we need to consider two cases:

(II.I) Assume that

$$\begin{aligned}\langle \mathcal{M}, \sigma\rangle \models & \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \\ & \qquad \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s)).\end{aligned}$$

Let $A \in Acc(\gamma^+) \cap Acc(\delta^-)$ and let $\bar{d}_A$ be such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A) \land \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \land \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \land Q(\bar{y}, s). \quad (51)$$

Let $B \in Acc(\delta^-)$. If $A = B$, by (51) and given that $\mathcal{M} \models UNAA$, then $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \forall \bar{y}_B(a = B(\bar{y}_B) \supset \bar{d}_A = \bar{y}_B)$. But since $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s)$, by (51), we conclude that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\rangle \models \forall \bar{y}_B(a = B(\bar{y}_B) \supset \neg\varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))$, or equivalently

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\rangle \models \neg\exists \bar{y}_B(a = B(\bar{y}_B) \land \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)). \quad (52)$$

If $A \neq B$, as $\mathcal{M} \models UNAA$, we obtain that $\langle \mathcal{M}, \sigma\rangle \models \forall(\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) \neq B(\bar{y}_B)]$. And then, by (51), we obtain that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\rangle \models \neg\exists \bar{y}_B(a = B(\bar{y}_B) \land \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))$. Thus, from this result and (52) we conclude the implication $\supset$ in (50).

(II.II) Assume that

$$\langle \mathcal{M}, \sigma\rangle \models \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} [\exists \bar{x}_A(a = A(\bar{x}_A) \land \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \land Q(\bar{y}, s) \land \\ \bigwedge_B \neg\exists \bar{y}_B(a = B(\bar{y}_B) \land \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s))].$$

Let $A \in Acc(\gamma^+) \cap Acc(\delta^-)$ and $\bar{d}_A$ be such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{d}_A) \land \varphi_A^{\gamma^+}(\bar{d}_A, \bar{x}, s) \land Q(\bar{y}, s) \land \\ \bigwedge_B \neg\exists \bar{y}_B(a = B(\bar{y}_B) \land \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)). \quad (53)$$

In order to prove that

$$\langle \mathcal{M}, \sigma\rangle \models \exists \bar{x}_A(a = A(\bar{x}_A) \land \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \land \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \land Q(\bar{y}, s)), \quad (54)$$

we only need to prove that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s)$. As $A \in Acc(\delta^-)$, by (53) we have that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\exists \bar{y}_A(a = A(\bar{y}_A) \land \varphi_A^{\delta^-}(\bar{y}_A, \bar{y}, s))$. But as $\mathcal{M} \models UNAA$, we obtain that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \forall(\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) = A(\bar{y}_A) \supset \bar{x}_A = \bar{y}_A]$. Thus, from the previous results and (53) is possible to conclude that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s)$, and from this (54) is satisfied. We can conclude then that the implication $\subset$ en (50) is true.

Proof of Proposition 2: Let $\mathcal{M} \models \Sigma$, and $C \in |\mathcal{M}|$ an action, $\bar{e}$ an l-tuple of objects in $|\mathcal{M}|$, where l is the number of arguments of action C. Let S be a state in $|\mathcal{M}|$, $\bar{b}$ an r-tuple of objects in $|\mathcal{M}|$, where r is the number of variables in $\bar{x}$, $\bar{d}$ an u-tuple of objects in $|\mathcal{M}|$, where u is the number of variables in $\bar{y}$. Finally, let $\sigma: \{\frac{C(\bar{e})}{a}, \frac{\bar{b}}{\bar{x}}, \frac{\bar{d}}{\bar{y}}, \frac{S}{s}\}$, be an assignment for the variables.

Assume that $\langle \mathcal{M}, \sigma \rangle \models Poss(a, s)$. By (13) and (14), and using the distributivity laws, we conclude that

$$\begin{aligned}\langle \mathcal{M}, \sigma \rangle \models P(\bar{x}, do(a, s)) \wedge Q(\bar{y}, do(a, s)) \equiv \\ &\gamma^+(a, \bar{x}, s) \wedge \delta^+(a, \bar{y}, s) \vee \\ &\gamma^+(a, \bar{x}, s) \wedge Q(\bar{y}, s) \wedge \neg\delta^-(a, \bar{y}, s) \vee \\ &\delta^+(a, \bar{y}, s) \wedge P(\bar{x}, s) \wedge \neg\gamma^-(a, \bar{x}, s) \vee \\ &P(\bar{x}, s) \wedge Q(\bar{y}, s) \wedge \neg(\gamma^-(a, \bar{x}, s) \vee \delta^-(a, \bar{y}, s)).\end{aligned} \tag{55}$$

By lemmas 1 and 2, (55) and the definitions of ξ^+ and ξ^-, we conclude that (15) holds.

Now, assume that $\langle \mathcal{M}, \sigma \rangle \models \xi^+(a, \bar{x}, \bar{y}, s)$. In order to show that $\langle \mathcal{M}, \sigma \rangle \models \neg\xi^-(a, \bar{x}, \bar{y}, s)$ we need to consider five cases.

(I) If

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^+)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s)),$$

then $\langle \mathcal{M}, \sigma \rangle \models \gamma^+(a, \bar{x}, s) \wedge \delta^+(a, \bar{y}, s)$. Therefore, by (13) and (14), $\langle \mathcal{M}, \sigma \rangle \models \neg(\gamma^-(a, \bar{x}, s) \vee \delta^-(a, \bar{y}, s))$. Thus, by definition of ξ^-, we conclude $\langle \mathcal{M}, \sigma \rangle \models \neg\xi^-(a, \bar{x}, \bar{y}, s)$.

(II) If

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^+) - Acc(\delta^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge Q(\bar{y}, s)).$$

then $\langle \mathcal{M}, \sigma \rangle \models \gamma^+(a, \bar{x}, s)$, but given that $A \notin Acc(\delta^-)$ and $\langle \mathcal{M}, \sigma \rangle \models UNAA$ we conclude that $\langle \mathcal{M}, \sigma \rangle \models \neg\delta^-(a, \bar{y}, s)$. Additionally, as P has a aeSSA, we conclude that $\langle \mathcal{M}, \sigma \rangle \models \neg\gamma^-(a, \bar{x}, s)$. Thus, by definition of ξ^- we obtain $\langle \mathcal{M}, \sigma \rangle \models \neg\xi^-(a, \bar{x}, \bar{y}, s)$.

(III) If

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\delta^+) - Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s) \wedge P(\bar{x}, s)),$$

then, analogously to (II), we conclude that $\langle \mathcal{M}, \sigma \rangle \models \neg\xi^-(a, \bar{x}, \bar{y}, s)$.

(IV) Assume that

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^+) \cap Acc(\delta^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \\ \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s)).$$

Let $A \in Acc(\gamma^+) \cap Acc(\delta^-)$ and $\bar{d}_A$ be such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s) \wedge \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s) \wedge Q(\bar{y}, s). \quad (56)$$

It holds $\langle \mathcal{M}, \sigma \rangle \models \gamma^+(a, \bar{x}, s)$. As P has an aeSSA, we conclude $\langle \mathcal{M}, \sigma \rangle \models \neg \gamma^-(a, \bar{x}, s)$. Let $B \in Acc(\delta^-)$. If $A = B$, as $\mathcal{M} \models \mathit{UNAA}$, it holds $\langle \mathcal{M}, \sigma \rangle \models \forall (\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) = B(\bar{y}_B) \supset \bar{x}_A = \bar{y}_B]$, thus, by (56), we conclude that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \forall \bar{y}_B [a = B(\bar{y}_B) \supset \bar{x}_A = \bar{y}_B]$. As $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg \varphi_A^{\delta^-}(\bar{x}_A, \bar{y}, s)$, it follows that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \forall \bar{y}_B [a = B(\bar{y}_B) \supset \neg \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)]$, or equivalently

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg \exists \bar{y}_B [a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)]. \quad (57)$$

If $A \neq B$, given that $\mathcal{M} \models \mathit{UNAA}$, we obtain $\langle \mathcal{M}, \sigma \rangle \models \forall (\bar{x}_A, \bar{y}_B)[A(\bar{x}_A) \neq B(\bar{y}_B)]$. Thus by (56), it follows that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg \exists \bar{y}_B [a = B(\bar{y}_B) \wedge \varphi_B^{\delta^-}(\bar{y}_B, \bar{y}, s)]. \quad (58)$$

From (57) and (58), we obtain $\langle \mathcal{M}, \sigma \rangle \models \neg \delta^-(a, \bar{y}, s)$. Again, from the definition of ξ^-, we conclude that $\langle \mathcal{M}, \sigma \rangle \models \neg \xi^-(a, \bar{x}, \bar{y}, s)$.

(V) If

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\delta^+) \cap Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \\ \wedge \varphi_A^{\delta^+}(\bar{x}_A, \bar{y}, s) \wedge \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s) \wedge P(\bar{x}, s)),$$

then $\mathcal{M} \models \neg \xi^-(a, \bar{x}, \bar{y}, s)$, analogously to (IV).

Proof of Proposition 4: Let $\mathcal{M} \models \Sigma$, and $C \in |\mathcal{M}|$ be an action, $\bar{e}$ an l-tuple of objects in $|\mathcal{M}|$, where l is the number of arguments of action C. Let S be a state in $|\mathcal{M}|$, such that $\langle \mathcal{M}, \sigma \rangle \models \mathit{Poss}(a, s)$. Let $d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n$ be individuals in $|\mathcal{M}|$ and let $\sigma : \{\frac{C(\bar{e})}{a}, \frac{d_1}{x_1}, \ldots, \frac{d_{i-1}}{x_{i-1}}, \frac{d_{i+i}}{x_{i+1}}, \ldots, \frac{d_n}{x_n}, \frac{S}{s}\}$, be a substitution.

We want to show that $\langle \mathcal{M}, \sigma \rangle$ satisfies

$$\forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s)) \equiv \\ \xi^+(a, \bar{x}', s) \vee (\forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s) \wedge \neg \xi^-(a, \bar{x}', s)). \quad (59)$$

($\Rightarrow$) First, we assume that

$$\langle \mathcal{M}, \sigma \rangle \models \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s)). \tag{60}$$

We have to consider four cases.

(I) If for each action name $A \in Acc(\gamma^+) \cup Acc(\gamma^-)$, $\langle \mathcal{M}, \sigma \rangle \models \neg \exists \bar{x}_A [a = A(\bar{x}_A)]$, we have that $\langle \mathcal{M}, \sigma \rangle \models \neg \gamma^+(a, \bar{x}, s)$ and $\langle \mathcal{M}, \sigma \rangle \models \neg \gamma^-(a, \bar{x}, s)$. By (60) we conclude $\langle \mathcal{M}, \sigma \rangle \models \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$. Thus, $\langle \mathcal{M}, \sigma \rangle$ satisfies the RHS of (59).

(II) Assume there is an action name $A \in Acc(\gamma^+) - Acc(\gamma^-)$, and a tuple $\bar{d}_A$, of objects in $|\mathcal{M}|$, such that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A)$.

(II.I) If $\langle \mathcal{M}, \sigma \rangle \models \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, then

$$\begin{aligned} \langle \mathcal{M}, \sigma \rangle \models \forall (x, \bar{x}_A)(\neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s) \supset \\ \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)). \end{aligned}$$

Therefore, $\langle \mathcal{M}, \sigma \rangle \models \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \forall x (\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)))$. Thus, $\langle \mathcal{M}, \sigma \rangle \models \xi^+(a, \bar{x}', s)$ and $\langle \mathcal{M}, \sigma \rangle$ satisfies the RHS of (59).

(II.II) Let $\langle \mathcal{M}, \sigma \rangle \models \neg \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, and d be an object in $|\mathcal{M}|$. On one hand, if we assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, we have

$$\begin{aligned} \langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s) \supset \\ \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s). \end{aligned} \tag{61}$$

On the other hand, if we assume that

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s),$$

then, given that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A)$, P has an aeSSA, $\mathcal{M} \models UNAA$ and (60) holds, we conclude

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s).$$

Thus,

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s). \tag{62}$$

By (61) and (62)

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \forall x(\neg P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, s, x_{i+1}, \dots, x_n, d)),$$

therefore $\langle \mathcal{M}, \sigma\rangle \models \xi^+(a, \bar{x}', s)$ and $\langle \mathcal{M}, \sigma\rangle$ satisfies the RHS of (59).

(III) Assume that there is an action name $A \in Acc(\gamma^+) \cap Acc(\gamma^-)$, and a tuple, $\bar{d}_A$, of individuals in $|\mathcal{M}|$, such that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A)$. Let d an individual in $|\mathcal{M}|$.

(III.I) Assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$. If $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \varphi_A^{\gamma^-}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$, then, since P has an aeSSA and $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A)$, we obtain

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s).$$

Thus, $\langle \mathcal{M}, \sigma \cup \{\frac{x}{d}\}\rangle \models \neg P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, do(a, s))$. But this contradicts (60). So, we conclude that

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\gamma^-}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$$

and

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s) \supset \neg\varphi_A^{\gamma^-}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s).$$

(III.II) Assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$. From $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A)$, $\mathcal{M} \models UNAA$ and (60), we obtain $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$, thus

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s).$$

From (III.I) and (III.II) we conclude that $\langle \mathcal{M}, \sigma\rangle$ satisfies

$$\begin{gathered}\exists \bar{x}_A(a = A(\bar{x}_A) \\ \bigwedge \\ \forall x[(\neg P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)) \\ \bigwedge \\ (P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s) \supset \neg\varphi_A^{\gamma^-}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s))]).\end{gathered}$$

Therefore, $\langle \mathcal{M}, \sigma\rangle \models \xi^+(a, \bar{x}', s)$ and $\langle \mathcal{M}, \sigma\rangle$ satisfies the RHS of (59).

(IV) Assume there is an action name $A \in Acc(\gamma^-) - Acc(\gamma^+)$, and a tuple $\bar{d}_A$, of individuals in $|\mathcal{M}|$, such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A). \tag{63}$$

If there exists an individual d in $|\mathcal{M}|$ such that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, then by (63) and $\mathcal{M} \models UNAA$, there is not an action name $B \in Acc(\gamma^+)$ such that $\langle \mathcal{M}, \sigma\rangle \models \exists \bar{x}_B(a = B(\bar{x}_B))$. Thus, $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg\gamma^+(a, \bar{x}, s)$, and, in consequence,

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s)),$$

but this contradicts (60). Therefore

$$\langle \mathcal{M}, \sigma\rangle \models \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s). \tag{64}$$

Assume that

$$\mathcal{M} \models \bigvee_{B \in Acc(\gamma^-) - Acc(\gamma^+)} \exists \bar{x}_B (a = B(\bar{x}_B) \wedge \\ \wedge \exists x \varphi_B^{\gamma^-}(\bar{x}_B, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)).$$

By (63) and $\mathcal{M} \models UNAA$, $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \exists x \varphi_A^{\gamma^-}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$. Thus there exists an individual d in $|\mathcal{M}|$ such that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \varphi_A^{\gamma^-}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$. Therefore, $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \exists \bar{x}_A(a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^-}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s))$. But P has an aeSSA, thus $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s))$, but this contradicts (60). It follows

$$\langle \mathcal{M}, \sigma\rangle \models \neg \bigvee_{B \in Acc(\gamma^-) - Acc(\gamma^+)} \exists \bar{x}_B (a = B(\bar{x}_B) \wedge \exists x \varphi_B^{\gamma^-}(\bar{x}_B, \bar{x}, s)). \tag{65}$$

Finally, by (63) and $\mathcal{M} \models UNAA$, we conclude

$$\langle \mathcal{M}, \sigma\rangle \models \neg \bigvee_{B \in Acc(\gamma^+) \cap Acc(\gamma^-)} \exists \bar{x}_B (a = B(\bar{x}_B) \wedge \exists x \varphi_B^{\gamma^-}(\bar{x}_B, \bar{x}, s)). \tag{66}$$

From (65) and (66), we obtain

$$\langle \mathcal{M}, \sigma\rangle \models \neg \bigvee_{B \in Acc(\gamma^-)} \exists \bar{x}_B (a = B(\bar{x}_B) \wedge \exists x \varphi_B^{\gamma^-}(\bar{x}_B, \bar{x}, s)). \tag{67}$$

By (64) and (67) we conclude that $\langle \mathcal{M}, \sigma \rangle \models \forall x P(\bar{x}, s) \wedge \neg \xi^{-}(a, \bar{x}', s)$. Thus $\langle \mathcal{M}, \sigma \rangle$ satisfies the RHS of (59).

($\Leftarrow$) Assume that $\langle \mathcal{M}, \sigma \rangle \models \xi^{+}(a, \bar{x}', s) \vee \forall x P(\bar{x}, s) \wedge \neg \xi - (a, \bar{x}', s)$. In this case we have to consider three cases.

(V) Assume that

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^{+}) - Acc(\gamma^{-})} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \forall x (\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^{+}}(\bar{x}_A, \bar{x}, s))).$$

Let $\bar{d}_A$ a tuple of individuals in $|\mathcal{M}|$, such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A) \wedge \forall x (P(\bar{x}, s) \supset \varphi_A^{\gamma^{+}}(\bar{x}_A, \bar{x}, s)), \tag{68}$$

for some action symbol $A \in Acc(\gamma^{+}) - Acc(\gamma^{-})$. Let d an individual in $|\mathcal{M}|$. On one hand, if we assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, then, by $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A)$, $\mathcal{M} \models UNAA$ and $A \in Acc(\gamma^{+}) - Acc(\gamma^{-})$, we conclude that for every action name $B \in Acc(\gamma^{-})$, $\langle \mathcal{M}, \sigma \rangle \models \neg \exists \bar{x}_B (a = B(\bar{x}_B))$, thus $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models \neg \gamma^{-}(a, \bar{x}, s)$. Therefore,

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s)). \tag{69}$$

On the other hand, if we assume that

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models \neg P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s),$$

then, by (68), we obtain $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \varphi_A^{\gamma^{+}}(\bar{x}_A, x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, s)$, thus $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^{+}}(\bar{x}_A, \bar{x}, s)$. In consequence,

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\} \rangle \models P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s)). \tag{70}$$

From (69) and (70), it follows that $\langle \mathcal{M}, \sigma \rangle \models \forall x P(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_n, do(a, s))$.

(VI) Assume that

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^{+}) \cap Acc(\gamma^{-})} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge$$
$$\forall x [(\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^{+}}(\bar{x}_A, \bar{x}, s)) \wedge (P(\bar{x}, s) \supset \neg \varphi_A^{\gamma^{-}}(\bar{x}_A, \bar{x}, s))]).$$

Let $\bar{d}_A$ be a tuple of individuals in $|\mathcal{M}|$ such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \forall x \neg P(\bar{x}, s) \supset \varphi_A^{\gamma^{+}}(\bar{x}_A, \bar{x}, s), \tag{71}$$

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \forall x P(\bar{x}, s) \supset \neg \varphi_A^{\gamma^{-}}(\bar{x}_A, \bar{x}, s), \tag{72}$$

for some action name $A \in Acc(\gamma^+) \cap Acc(\gamma^-)$. Let d an individual in $|\mathcal{M}|$. On one hand, if we assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models P(\bar{x}, s)$, then, by (72), $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \neg\varphi_A^{\gamma^-}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$. From $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A)$, $\mathcal{M} \models UNAA$ and the fact that P has an aeSSA we conclude that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg\gamma^-(a, \bar{x}, s)$. Then

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, do(a, s)). \tag{73}$$

On the other hand, if we assume that $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg P(\bar{x}, s)$, by (71) and $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \varphi_A^{\gamma^+}(\bar{x}_A, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, s)$, we obtain $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A) \wedge \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)$. In consequence,

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models P(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n, do(a, s)). \tag{74}$$

From (73) and (74), it follows $\langle \mathcal{M}, \sigma\rangle \models \forall x P(\bar{x}, do(a, s))$.

(VII) Assume that

$$\langle \mathcal{M}, \sigma\rangle \models \forall x P(\bar{x}, s), \tag{75}$$

$$\langle \mathcal{M}, \sigma\rangle \models \neg \bigvee_{A \in Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \wedge \exists x \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)). \tag{76}$$

We have to consider two cases:

(VII.I) For every action name $A \in Acc(\gamma^-)$ it holds $\langle \mathcal{M}, \sigma\rangle \models \neg\exists \bar{x}_A (a = A(\bar{x}_A))$. In this case, for every object d in $|\mathcal{M}|$ we conclude $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg\gamma^-(a, \bar{x}, s)$, and then, by (75), we obtain $\langle \mathcal{M}, \sigma\rangle \models \forall x P(\bar{x}, do(a, s))$.

(VII.II) Assume that there exists an action name $A \in Acc(\gamma^-)$ and a tuple of individuals $\bar{d}_A$ such that

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models a = A(\bar{x}_A). \tag{77}$$

By (76), we have

$$\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\}\rangle \models \forall x \neg\varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s).$$

and then, by (77) and $\mathcal{M} \models UNAA$, we conclude that for every individual d in $|\mathcal{M}|$, $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}\}\rangle \models \neg\gamma^-(a, \bar{x}, s)$, in consequence, by (75), we obtain $\langle \mathcal{M}, \sigma\rangle \models \forall x P(\bar{x}, do(a, s))$. This finishes the proof of (59).

Now we will prove (19). Assume that $\langle \mathcal{M}, \sigma\rangle \models \xi^+(a, \bar{x}', s)$. There are two possibilities:

(I) If

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^+) - Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \forall x (\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s))).$$

then $\langle \mathcal{M}, \sigma \rangle \models \neg \xi^-(a, \bar{x}', s)$, because $\mathcal{M} \models UNAA$ and when $A \in Acc(\gamma^+) - Acc(\gamma^-)$, it holds $A \notin Acc(\gamma^-)$.

(II) Assume

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^+) \cap Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \forall x [(\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)) \land (P(\bar{x}, s) \supset \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s))]). \quad (78)$$

First, we can conclude that

$$\langle \mathcal{M}, \sigma \rangle \models \neg \bigvee_{A \in Acc(\gamma^-) - Acc(\gamma^+)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \exists x \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)), \quad (79)$$

because $\mathcal{M} \models UNAA$ and when $A \in Acc(\gamma^+) \cap Acc(\gamma^-)$, it holds $A \notin Acc(\gamma^-) - Acc(\gamma^+)$. Second, if we assume that

$$\langle \mathcal{M}, \sigma \rangle \models \bigvee_{A \in Acc(\gamma^-) \cap Acc(\gamma^+)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \exists x \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)),$$

then there exists a tuple of individuals $\bar{d}_A$, in $|\mathcal{M}|$, and an individual d in $|\mathcal{M}|$, such that for some action name $A \in Acc(\gamma^+) \cap Acc(\gamma^-)$

$$\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A) \land \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s). \quad (80)$$

From (78), $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A)$ and $\mathcal{M} \models UNAA$, it follows that $\langle \mathcal{M}, \sigma \cup \{\frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \forall x [(P(\bar{x}, s) \supset \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)) \land (\neg P(\bar{x}, s) \supset \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s))]$. Therefore, if $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models P(\bar{x}, s)$, then $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)$. This contradicts (80). If $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg P(\bar{x}, s)$, then $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \varphi_A^{\gamma^+}(\bar{x}_A, \bar{x}, s)$. But P has an aeSSA, and then, by $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models a = A(\bar{x}_A)$, we can conclude $\langle \mathcal{M}, \sigma \cup \{\frac{d}{x}, \frac{\bar{d}_A}{\bar{x}_A}\} \rangle \models \neg \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)$. But this again contradicts (80). In consequence,

$$\langle \mathcal{M}, \sigma \rangle \models \neg \bigvee_{A \in Acc(\gamma^-) \cap Acc(\gamma^+)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \exists x \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)), \quad (81)$$

From (79) and (81), it follows

$$\langle \mathcal{M}, \sigma \rangle \models \neg \bigvee_{A \in Acc(\gamma^-)} \exists \bar{x}_A (a = A(\bar{x}_A) \land \exists x \varphi_A^{\gamma^-}(\bar{x}_A, \bar{x}, s)).$$

Therefore, $\langle \mathcal{M}, \sigma \rangle \models \neg \xi^-(a, \bar{x}', s)$. By (I) and (II), (19) holds.