Reasoning Aspects in Information Systems and Databases

Paula Gouveia, Cristina Sernadas

Departamento de Matemática - Instituto Superior Técnico
Av. Rovisco Pais, 1096 Lisboa Codex, PORTUGAL
&
INESC
Apartado 10105, 1017 Lisboa Codex, PORTUGAL
Phone: 351-1-3100322 Fax: 351-1-525843 E-mail: mpg@inesc.pt

Deductive approaches are very important namely for reasoning and detecting inconsistencies in conceptual schema definition. Object-oriented approaches are very efficient in providing structuring mechanisms to conceptual schema development. We bring the two together adopting a logic framework for defining objects as well as their interactions and we present a tableau system that allows the reasoning with objects as well as with object communities. This system is easy to use and is the basis of a theorem prover prototype.

1. Introduction

Among the formal approaches to conceptual schema definition we can refer to the object-oriented and the deductive perspectives. Following the object-oriented point of view [SernadasA *et al* 89,Wieringa 90,Booch 91,Junglaus *et al* 91,Loucopoulos and Zicari 92], the conceptual schema is a collection of objects that can interact with each other. Each object [SernadasA *et al* 91] corresponds to an entity in the universe of discourse and includes a set of local observations (attributes) and a set of actions (events or methods). On the other hand, following the deductive approach [Gallaire *et al* 84,Reiter 84,Demolombe *et al* 90,Olivé 90], the conceptual schema is defined as a theory in some logic framework. The formulae in the theory represent assertions that are relevant to the entities of the universe of discourse. Among the main advantages of the deductive approach we can refer to the possibility of reasoning namely for proving properties and checking inconsistencies. The main advantage of the object-oriented point of view is the extensive use of structuring primitives which makes the conceptual schema easier to understand as well as more structured [SernadasC *et al* 90].

Hence, it would be highly desirable to have the niceties of both worlds: that is to say define the conceptual schema in a structured way by identifying the relevant objects and their interactions but defining each object as a theory. Moreover, the interaction mechanisms themselves should also be described using the adopted logic framework.

Of course, one of the problems is the selection of the logic framework to be adopted. We decided to start with a many-sorted first-order linear temporal logic [Emerson 90,Manna and Pnuelli 92| following a strong school in the area of information systems and databases [SernadasA 80,Saake and Lipeck 89,SernadasC *et al* 92]. On the other hand, being an object a dynamic entity that evolves through the occurrence of actions, its life (life-cycle or trajectory) can be identified as a sequence of actions and each state can be identified with the sequence of actions that have occurred so far. Once again the temporal formalism seems to be adequate to specify these dynamic aspects.

In most cases temporal formalisms are adopted for specification purposes. In a few cases temporal axiomatic systems are used for reasoning with specifications. That is to say, a lot of effort and expertise has to be put in producing proofs of assertions. It is then easy to argue that clausal-like formalisms are more interesting namely because of the operational style of proofs. These proofs are made easily and moreover they can automated.

Our objective in this paper is to present a two-level system for reasoning with objects based on tableaux. In the first-level, given an object description (specification) and an assertion to be verified we can build a tableau using the inference rules of the calculus. In the second-level we can reason about the conceptual schema itself using the objects as well as interaction mechanisms. Our tableau system is based on the prefixed tableau systems for modal logic presented in [Fitting 83]. This kind of systems avoid the construction of sets of interdependent tableaux as in [Li and SernadasA 91]. Tableaux are comparable to the operational flavour of clausal-like approaches in the sense they can be used in a simple way and moreover they can also be automated. For our calculus we adopt the logic approach to object descriptions presented in [SernadasA *et al* 92].

The tableau systems incorporates different aspects represented by different collections of rules namely, the usual collection of rules related to connectives and quantifiers, rules for dealing with the temporal operators adopting [Gouveia 92,Gouveia *et al* 93], rules related to sequences of states and rules that are specific of objects (related to birth, effects of actions on observations, occurrence of actions and enabling of observations and actions). We do not detail here the rules for arithmetic, equational and inequational reasoning. At the moment we have a preliminary version of a theorem prover prototype. This prototype is based on the tableau system presented here and includes a module for arithmetic, equational and inequational reasoning following the work of [Nelson and Oppen 79,Nelson and Oppen 80].

The paper is organized as follows. In section 2, we present the syntax and the semantics of object descriptions. In section 3, we introduce the syntax and semantics of prefixed formulae, the tableau system, relevant results and examples. In section 4, we outline reasoning aspects related with interaction mechanisms.

2. Objects

In this section we present the syntax and semantics of object descriptions. We assume fixed a set of data types, i. e., a data signature DT=(S,OP) including the sort bool and a data specification $SPEC_{DT}$=(DT,EQ).

2.1 Syntax

An object description includes a signature and a set of formulae. The signature introduces the action and the observation symbols which are related, respectively, with the dynamic and static aspects of the description. In order to specify the constraints on the object behaviour we use formulae from a temporal linear, many-sorted first order logic with equality suitable extended to deal with object specific features [SernadasA *et al* 92].

DEFINITION 2.1.1 : OBJECT DESCRIPTION SIGNATURE

An *object description signature* is a tuple Σ_{ob}=(ACT,OBS) where

- ACT is a S*-indexed family of finite sets whose elements are called action symbols
- OBS is a S-indexed family of finite sets whose elements are called observation symbols

EXAMPLE 2.1.2:

Let us consider an example related with a library application where we have objects like books and users. For instance, each book has a name and a registration number and it can be taken and returned by an user. The object description signature is Σ_{book}=(ACT,OBS) with

$\text{ACT}_{\text{string nat}}$={new} $\text{OBS}_{\text{string}}$={name}

ACT_{user}={tak, ret} OBS_{bool}={avail}

OBS_{nat}={reg}.

DEFINITION 2.1.3: TERMS

Given a S-indexed family of sets of variables X, the family of *terms* associated with DT, $T_{DT}(X)$, is defined as usual:

- if $x \in X_s$ then $x \in T_{DT}(X)_s$
- if $op \in OP_{s1,...,sn,s}$ and $ti \in T_{DT}(X)_{si}$, n≥0, then $op(t1,...,tn) \in T_{DT}(X)_s$
- $T_{DT}(X)$ has no more terms.

DEFINITION 2.1.5: FORMULAE

Given a object description signature Σ_{ob}=(ACT,OBS) and a S-indexed family of sets of variables X we define the *set of formulae*, F_{ob}, associated with Σ_{ob} as follows:

- $\Diamond a(t1,...,tn)$, $\nabla a(t1,...,tn) \in F_{ob}$ for every $a \in ACT_{s1,...,sn}$ and terms t1,...,tn of sorts s1,...,sn, respectively, n≥0
- $\Diamond o(t) \in F_{ob}$ for every $o \in OBS_s$ and term t of sort s
- $op(t1,...,tn) \in F_{ob}$ for every $op \in OP_{s1,...,sn,bool}$ and terms t1,...,tn of sorts s1,...,sn, respectively, n≥0
- $t1=t2 \in F_{ob}$ for every terms t1, t2 of of the same sort s
- if $f1 \in F_{ob}$ and $f2 \in F_{ob}$ then $\neg f1 \in F_{ob}$, $f1 \wedge f2 \in F_{ob}$, $\exists x f1$ ($x \in X_s$), $Ff1 \in F_{ob}$, $Pf1 \in F_{ob}$ and $Xf1 \in F_{ob}$
- F_{ob} has no more formulae.

Informally, we can say that the formula $\nabla a(t1,...,tn)$ refers to the occurrence of an a-action and formulae as $\Diamond a(t1,...,tn)$ and $\Diamond o(t)$ indicate that an a-action or an o-observation is enabled. Intuitively we say that an action is enabled when it is allowed (but not forced) to occur and that an observation is enabled when the value of its argument is the current value of the slot.

The temporal operators are as usual: F is the *some time in the future* operator, P is the *some time in the past* operator and X is the *next* operator.

We denote by t[x | t'] (f[x | t']) the term (formula) obtained from t (f) when x is substituted for t' in t (f). The notion of substitution is as usual and, in particular, when dealing with formulae is assumed that t' is free for x in f.

DEFINITION 2.1.5: ABBREVIATIONS

We consider the usual abbreviations:

- $f1 \vee f2 =_{def} \neg((\neg f1) \wedge (\neg f2))$
- $f1 \Leftrightarrow f2 =_{def} (f1 \Rightarrow f2) \wedge (f2 \Rightarrow f1)$
- $Hf =_{def} \neg P \neg f$
- $\exists \vec{x} f =_{def} \exists x1 \ldots \exists xnf$ if $\vec{x} = x1,\ldots,xn$
- $f1 \Rightarrow f2 =_{def} (\neg f1) \vee f2$
- $\forall xf =_{def} \neg \exists x \neg f$
- $Gf =_{def} \neg F \neg f$

We consider also the following abbreviations related with action and observation symbols.

DEFINITION 2.1.6: BIRTH FORMULA

A *birth formula* for the action symbol a is the following abbreviation

$$\circledast a(\vec{x}) =_{def}$$

$$(\forall \vec{x}_1 \ldots \forall \vec{x}_n (\nabla a_1(\vec{x}_1) \vee \ldots \vee \nabla a_n(\vec{x}_n)) \Rightarrow P \exists \vec{x} \nabla a(\vec{x}))$$

$$\wedge$$

$$\forall \vec{x} (\nabla a(\vec{x}) \Rightarrow H(\forall y_1 \neg \Diamond o_1(y_1) \wedge \ldots \wedge \forall y_m \neg \Diamond o_m(y_m)) \wedge$$
$$\wedge \forall y_1 \neg \Diamond o_1(y_1) \wedge \ldots \wedge \forall y_m \neg \Diamond o_m(y_m)) \wedge$$
$$\wedge G(\forall \vec{y} \neg \nabla a(\vec{y}))$$

where $a, a_1, \ldots, a_n$ are all the action symbols of the object description signature, $a_i \neq a$, and $o_1, \ldots, o_m$ are all the observation symbols of the object description signature.

Informally, this abbreviation wrt to an action symbol a states that (i) the occurrence of an a-action must precede the occurrence of any other action, (ii) when an a-action occur none observation is or has been enabled, (iii) an a-action occurs only once. So an a-action acts as a "birth" action.

DEFINITION 2.1.7: VALUATION FORMULA

A *valuation formula* for the action symbol a is the following abbreviation

$$[a(\vec{x})]o_1(y_1),\ldots,o_{k1}(y_{k1})$$
$$\rightarrow o_1(\phi_1(y_1,\vec{x})),\ldots,o_{k1}(\phi_{k1}(y_{k1},\vec{x})),o_{k1+1}(\phi_{k1+1}(\vec{x})),\ldots,o_{k2}(\phi_{k2}(\vec{x})) \qquad =_{def}$$

$$\forall \vec{x} \forall y_1 \forall y_{k1} \ldots \forall y_{k2}, \ldots \forall y_{k3},$$
$$(\nabla a(\vec{x}) \wedge \Diamond o_1(y_1) \wedge \ldots \wedge \Diamond o_{k1}(y_{k1}) \wedge \Diamond o_{k2+1}(y_{k2+1}) \wedge \ldots \wedge \Diamond o_{k3}(y_{k3})$$
$$\Rightarrow$$
$$X(\Diamond o_1(\phi_1(y_1,\vec{x})) \wedge \ldots \wedge \Diamond o_{k1}(\phi_{k1}(y_{k1},\vec{x})) \wedge \Diamond o_{k1+1}(\phi_{k1+1}(\vec{x})) \wedge \ldots \wedge \Diamond o_{k2}(\phi_{k2}(\vec{x}))$$

$$\wedge \Diamond o_{k2+1}(y_{k2+1}) \wedge \ldots \wedge \Diamond o_{k3}(y_{k3}))))$$

where $o_1,\ldots,o_{k1},\ldots,o_{k2},\ldots,o_{k3}$ are all the observation symbols of object description signature, $\phi_{kj} \in OP_{s1,\ldots,sn,s}$, if $a \in ACT_{\varepsilon}\vec{x}$ is omitted, k_1 can be 0 and k_2 can be equal to k_3

An observation symbol in OBS that does not occur in a valuation formula is an *invariant* for it. If $k_1=0$ we say that the valuation formula has no context.

Informally a valuation formula
$[a(\vec{x})]o_1(y_1),\ldots,o_{k1}(y_{k1}) \rightarrow o_1(\phi_1(y_1,\vec{x})),\ldots,o_{k1}(\phi_{k1}(y_{k1},\vec{x}),o_{k1+1}(\phi_{k1+1}(\vec{x})),\ldots,o_{k2}(\phi_{k2}(\vec{x}))$
describes the effects of an a-action on the observations. On the left side of → we only represent (by $o_1(y_1),\ldots,o_{k1}(y_{k1})$) the observations that were enabled when the a-action occurs and whose arguments are relevant to state the effect of the a-action[1].

EXAMPLE 2.1.8:
Considering Σ_{book} we have that $G([new(x,y)] \rightarrow avail(true) \wedge name(x) \wedge reg(y)) \in F_{book}$ and $G([tak(x)] \rightarrow avail(false)) \in F_{book}$. The first formula has neither context nor invariants. On the other hand, the observation symbols name and reg are invariants for the second one.

DEFINITION 2.1.9 : OBJECT DESCRIPTION
An object description is a pair $ob=(\Sigma,F)$ where

- Σ–(ACT,OBS) is an object description signature
- $F \subset F_{ob}$ is finite.

The formulae in F are the *axioms* of the object description.

Given an object description $ob=(\Sigma,F)$ we say that X is the *underlying S-indexed family of sets of variables* if X is the least S-indexed family of sets of variables such that all the terms involved in the formulae of F are elements of $T_{DT}(X)$.

EXAMPLE 2.1.10:
The pair $book=(\Sigma_{book}, F)$, where $F \subset F_{book}$ is as follows, is an object description

$$
\begin{array}{lll}
F= & \{G(\divideontimes new(x,y)), & (a1)\\
 & G([new(x,y)] \rightarrow avail(true) \wedge name(x) \wedge reg(y)) & (a2)\\
 & G([tak(x)] \rightarrow avail(false)) & (a3)\\
 & G([ret(x)] \rightarrow avail(true)) & (a4)\\
 & G(\forall x(\Diamond tak(x) \Rightarrow \Diamond avail(true))) & (a5)\\
 & G(\forall x(\Diamond ret(x) \Rightarrow \Diamond avail(false))) & (a6)\}.
\end{array}
$$

Axiom (a1) establishes new-actions as "birth" actions. Axioms (a1), (a2) and (a3) establish the effects of actions over observations: a new-action affects all the observations but tak-actions and ret-actions only affect avail-observations. The other axioms

[1] So, the observations that are not enabled when the a-action occurs but become enabled after its occurrence or are enable but whose arguments are not relevant to state the effect of the a-action are only represented (by $o_{k1+1}(\phi_{k1+1}(\vec{x})),\ldots,o_{k2}(o_{k2}(\vec{x})))$ on the right side of →. The observations that are not changed by the a-action are not represented at all in the abbreviation.

establish other constraints on the object behaviour. For instance, informally, (a5) states that a book can only be taken if it is available.

2.2 Semantics

We assume some fixed model A_{DT} for $SPEC_{DT}$. As usual A_{si} denotes the carrier set of the sort si and for each $op \in OP_{s1,\ldots,sn,\,s}$ $[\![op]\!]_{DT}$ is the interpretation of the operation symbol op. Interpretation structures for object descriptions are based on suitable enriched Kripke structures [SernadasA *et al* 92].

DEFINITION 2.2.1 : INTERPRETATION STRUCTURE FOR *ob*

An *interpretation structure for the object description* $ob=(\Sigma=(ACT,OBS),F)$ is a map $\lambda:\mathbb{N}_0 \rightarrow IB$ where

$$IB=\mathcal{P}(\{\nabla a(v1,\ldots,vn):\ a\in ACT_{s1,\ldots,sn},\ ACT_{s1,\ldots,sn}\in ACT,\ n\geq 0,\ vi\in A_{si}\}\cup$$
$$\{\Diamond a(v1,\ldots,vn):\ a\in ACT_{s1,\ldots,sn},\ ACT_{s1,\ldots,sn}\in ACT,\ n\geq 0,\ vi\in A_{si}\}\cup$$
$$\{\Diamond o(v):\ o\in OBS_s,\ OBS_s\in OBS,\ v\in A_s\}^1$$

such that

- $\lambda(0)=\emptyset$
- if $\nabla a(v1,\ldots,vn)\in\lambda(k)$ then $\Diamond a(v1,\ldots,vn)\in\lambda(k)$, $n\geq 0$
- if $\nabla a(v1,\ldots,vn)$, $\nabla a(u1,\ldots,un)\in\lambda(k)$ then $vi=ui$, $1\leq i\leq n$
- for each $k\in\mathbb{N}_0$ there is some $a\in ACT_{s1,\ldots,sn}$ and $v1,\ldots,vn$, $n\geq 0$ such that $\nabla a(v1,\ldots,vn)\in\lambda(k)$
- if $\Diamond o(v)$, $\Diamond o(u)\in\lambda(k)$ then $v=u$
- if $\Diamond o(v)\in\lambda(k)$ then for each $k'>k$ $\Diamond o(u)\in\lambda(k')$ for some suitable u.

The elements of $\mathbb{N}_0$ are the *worlds* or *states* of the interpretation structure. The natural number 0 is the initial state. Hence, no action has occur yet and so the first condition imposes that $\lambda(0)$ is empty. The second condition states that in any state an action only occurs if it is enabled on it. The third imposes mutual exclusion of actions of the same kind. The fourth reflects the fact that an object evolves exclusively through the occurrence of its actions. The fifth condition states that in each state only one o-observation is enabled and the last condition expresses that we can not "delete" observations: if we have an o-observation then in future we will have always some o-observation.

DEFINITION 2.2.2 : ASSIGNMENT

Given an interpretation structure for the object description ob and the underlying S-indexed family of sets of variables X, an *assignment* to X is a S-indexed family ASG of maps such that $ASG_s: X_s \rightarrow A_s$.

DEFINITION 2.2.3 : INTERPRETATION

Given an interpretation structure λ for the object description ob, the underlying S-indexed family of sets of variables X and an assignment ASG to X, the *interpretation* over λ through ASG of the terms in $T_{DT}(X)$ is defined as follows:

- $[\![x]\!]_{ASG}=ASG_s(x)$ for $x\in X_s$
- $[\![op(t1,\ldots,tn)]\!]_{ASG}=[\![op]\!]_{DT}([\![t1]\!]_{ASG},\ldots,[\![tn]\!]_{ASG})$ for $op\in OP_{s1,\ldots,sn,s}$ and terms t1,...,tn of sorts s1,...,sn, respectively.

[1] Given a set T, $\mathcal{P}(T)$ denotes the set of all subsets of T.

DEFINITION 2.2.4 : SATISFACTION OF FORMULAE

Given an interpretation structure $\lambda:\mathbb{N}_0\rightarrow\mathbb{B}$ for the object description ob, the underlying S-indexed family of sets of variables X, an assignment ASG to X and a state k, the *satisfaction by λ for ASG at k* is inductively defined as follows:

- $\lambda,ASG,k\models\nabla a(t1,\ldots,tn)$, $a\in ACT_{s1,\ldots,sn}$, $n\geq 0$ iff $\nabla a(\llbracket t1\rrbracket_{ASG},\ldots,\llbracket tn\rrbracket_{ASG})\in\lambda(k)$.
- $\lambda,ASG,k\models\diamond a(t1,\ldots,tn)$, $a\in ACT_{s1,\ldots,sn}$, $n\geq 0$ iff $\diamond a(\llbracket t1\rrbracket_{ASG},\ldots,\llbracket tn\rrbracket_{ASG})\in\lambda(k)$
- $\lambda,ASG,k\models\diamond o(t)$, $o\in OBS_s$, iff $\diamond o(\llbracket t\rrbracket_{ASG})\in\lambda(k)$
- $\lambda,ASG,k\models op(t1,\ldots,tn)$, $op\in OP_{s1,\ldots,sn,bool}$ $n\geq 0$ iff $\llbracket op\rrbracket_{DT}(\llbracket t1\rrbracket_{ASG}\ldots,\llbracket tn\rrbracket_{ASG})=1$[1]
- $\lambda,ASG,k\models\neg f$ iff is not the case that $\lambda,ASG,k\models f$
- $\lambda,ASG,k\models f1\wedge f2$ iff $\lambda,ASG,k\models f1$ and $\lambda,ASG,k\models f2$
- $\lambda,ASG,k\models\exists xf$, $x\in X_s$ iff there is an assignment ASG' x-equivalent[2] to ASG such that $\lambda,ASG',k\models f$
- $\lambda,ASG,k\models Ff$ iff there is $k'>k$ such that $\lambda,ASG,k'\models f$
- $\lambda,ASG,k\models Xf$ iff $\lambda,ASG,k+1\models f$
- $\lambda,ASG,k\models Pf$ iff there is $k'<k$ such that $\lambda,ASG,k'\models f$
- $\lambda,ASG,k\models t1=t2$ iff $\llbracket t1\rrbracket_{ASG}=\llbracket t2\rrbracket_{ASG}$.

DEFINITION 2.2.5 : INITIALLY VALID FORMULAE

Given an interpretation structure $\lambda:\mathbb{N}_0\rightarrow\mathbb{B}$ for the object description *ob* and the underlying S-indexed family of sets of variables X

- a formulae f is *initially valid* in λ, denoted by $\lambda\models_0 f$, iff $\lambda,ASG,0\models f$ for every assignment ASG to X
- a formulae f is *initially valid*, denoted by $\models_0 f$, iff $\lambda\models_0 f$ for every interpretation structure λ.

DEFINITION 2.2.6 : MODEL FOR *ob*

The interpretation structure λ for the object description *ob* is a *model* for *ob* iff every axiom of*ob* is initially valid in λ.

So, the axioms of the object description are formulae intended to be valid at the initial world and they reflect the properties that should be verified during the object's life. A model represents a possible life of the object, i. e., one that verifies the properties described by the axioms.

DEFINITION 2.2.8 : INCONSISTENT OBJECT DESCRIPTIONS

An object description *ob* is *inconsistent* iff there is no model for *ob*.

DEFINITION 2.2.7 : *ob*-VALID FORMULAE

Given an object description *ob* a formulae f is *ob-valid* iff for every model λ for *ob* and every assignment ASG to the underlying S-indexed family of sets of variables X, $\lambda,ASG,k\models f$ for all $k\neq 0$.

[1] Assuming $A_{bool}=\{1,0\}$.

[2] An assignment ASG' is x-equivalent to ASG iff $ASG(y)=ASG'(y)$ for every variable $y\neq x$.

An *ob*-valid formula accounts for a property we want verified on every non initial world of a model, i. e., informally, on every state of the object after its creation during an object's possible life.

3. Tableau Systems

In this section we present the tableau system ***OB*** for reasoning about a given object description $ob=(\Sigma,F)$.

We introduce *prefixed formulae*, i. e., each formula of F_{ob} can be prefixed where the prefix, intuitively, indicates a state where we want this formula to be true. Moreover we also introduce as formulae, sequences of such prefixes.

3.1 Syntax and Semantics of Prefixed Formulae

Given a set A, we consider the following sets: $\circ A=\{\circ a\colon a\in A\}$, $A^{i}=\{a^{i}\colon a\in A\}$ and the operation ν such that $\nu(a)=\nu(a^{i})=a$.

DEFINITION 3.1.1 : PREFIXED FORMULAE
Consider the set $\mathbf{W}=\{\mathbf{w}_0,\mathbf{w}_1,\ldots\}$. The set of all *prefixed formulae* associated with the object description $ob=(\Sigma,F)$ is

$F_{obP}=\{p{:}f : p\in Pr,\ f\in F_{ob}\}\cup Sq$

where $Pr=\mathbf{W}\cup\mathbf{W}^{i}$ and $Sq=\{\sigma_1\ldots\sigma_r\in(Pr\cup\circ Pr)^{*} : r>1,\ \sigma_1\notin\circ Pr\}$.

Within this context each element of *Pr* is a *prefix*. An element of $\mathbf{W}^{i}$ is an *initial prefix*. The symbol ∘ is the *direct accessibility relation symbol*. We will see below that these designations are related with the semantics of prefixed formulae. The semantics of a prefix is a state and the semantics of an initial prefix will be an initial state. Elements of *Sq* will represent sequences of states: if, for instance, in $\mathbf{w}_1\circ\mathbf{w}_2\mathbf{w}_3$ the semantics of $\mathbf{w}_1$ is a state $k\in\mathbb{N}_0$ then $\mathbf{w}_2$ will represent the next state, i. e., k+1 and $\mathbf{w}_3$ some state n>k.

The following definitions, related with prefixes, will be necessary when defining the tableau system.

DEFINITION 3.1.2:
Let $p,p'\in Pr$, $\circ q\in\circ Pr$, $\sigma=\sigma_1\ldots\sigma_r\in Sq$, $p{:}f\in F_{obP}$, $\Pi\subset Pr\cup\circ Pr$ e $\Phi\subset F_{obP}$

- $pref(p)=p$, $pref(\circ q)=q$ and $pref(\Pi)=\{pref(p')\colon p'\in\Pi\}$; each $pref(\sigma_i)$ is a *prefix of* σ
- $\nu(t)=\nu(pref(\sigma_1))\ldots\nu(pref(\sigma_r))$
- $comp(\sigma)=\{\sigma_1,\ldots,\sigma_r\}$ is the set of the *components of* σ; $sucessor(\sigma_i)=\sigma_{i+1}$, $1\le i<r$; $\nu(comp(\sigma))=\{\nu(pref(\sigma_i))\colon 1\le i\le r\}$
- if $\nu(p)=\nu(pref(\sigma_i))$ then the set of *future components of p in* σ is $comp_f(\sigma,p)=\{\sigma_j\colon i<j\le r\}$;
- if $\nu(p)=\nu(pref(\sigma_i))$ then the set of *past components of p in* σ is $comp_p(\sigma,p)=\{\sigma_j\colon 1\le j<i\}$;
- p' is *accessible from p in* σ iff $\nu(p')\in pref(comp_f(\nu(\sigma),\nu(p)))$ or $\nu(p)\in pref(comp_p(\nu(\sigma),\nu(p')))$
- p is *used* in Φ iff $p\in pref(\Phi)=\{p'\in Pr\colon p'{:}f\in\Phi$ or $p'\in pref(comp(\sigma)),\ \sigma\in\Phi\}$
- p is *free* in Φ iff $p=\mathbf{w}_k\in\mathbf{W}$, and neither $\mathbf{w}_k$ nor $\mathbf{w}_k{}^{i}$ are used in Φ.

DEFINITION 3.1.3: FUTURE EXTENSIONS OF FORMULAE

Let $\sigma=\sigma_1 \ldots \sigma_r \in Sq$, p,q prefixes, p prefix of σ and q free in $\{\sigma\}$. The *future extension of σ from p through q*, $ext_f(\sigma,p,q)$, is the set of formulae $\sigma' \in Sq$ such that

- $\sigma'=\sigma_1 \ldots \sigma_j \tau$ where $p=pref(\sigma_j)$ for some $1 \leq j \leq r$ and τ is a permutation of elements of $comp_f(\sigma,p) \cup \{q\}$
- $\sigma' \downarrow comp(\sigma)=\sigma$ where $\sigma' \downarrow comp(\sigma)$ is the restriction of σ' to the set $comp(\sigma)$
- in σ', $sucessor(q) \notin \circ Pr$.

We define, in a similar way, the *past extension of σ from p through q*, $ext_p(\sigma,p,q)$.

EXAMPLE 3.1.4:

Consider the formula $\sigma=\omega_2\omega_1 \circ \omega_5\omega_3 \in Sq$. Then, for example:

(i) $comp(\sigma)=\{\omega_1,\omega_2,\omega_3, \circ \omega_5\}$

(ii) $pref(\circ \omega_5)=\omega_5$

(iii) ω_5 is accessible from ω_2 in σ

(iv) $comp_f(\sigma,\omega_2)=\{\omega_1,\omega_3, \circ \omega_5\}$

(v)$comp_p(\sigma,\omega_1)=\{\omega_2\}$

(vi) $\omega_2\omega_1 \circ \omega_5\omega_6\omega_3$is an element of $ext_f(\sigma,\omega_5,\omega_6)$ but $\omega_2\omega_1\omega_6 \circ \omega_5\omega_3$ is not an element of $ext_p(\sigma,\omega_3,\omega_6)$.

Next, we present the semantics of prefixed formulae.

DEFINITION 3.1.5: INTERPRETATION OF A SET OF PREFIXES

Let $\Pi \subset Pr$ be a set of prefixes and $\lambda:\mathbb{N}_0 \rightarrow IB$ an interpretation structure for the object description *ob*. An interpretation I^p of Π on λ is a map $I^p:\Pi \rightarrow \mathbb{N}_0$ such that $I^p(p)=0$, for every initial prefix p and if $v(p')=v(p'')$ then $I^p(p')=I^p(p'')$.

DEFINITION 3.1.6: SATISFIABLE FORMULA

A formula $f \in F_{obP}$ is *satisfiable* iff there is an interpretation structure λ for *ob* such that

- if f=p:f1 then there is an interpretation I^p of {p} on λ and an assignment ASG such that $\lambda,ASG,I^p(p) \vDash f1$
- if $f=\sigma_1 \ldots \sigma_r \in Sq$ then there is an interpretation I^p of $pref(comp(f))$ on λ such that for every $1 \leq i < r$ $I^p(pref(\sigma_i))<I^p(pref(\sigma_{i+1}))$ and if $\sigma_{i+1} \in \circ Pr$, then $I^p(pref(\sigma_{i+1}))=I^p(pref(\sigma_i))+1$.

Notation: $\lambda,ASG,I^p \vDash f$.

DEFINITION 3.1.7: SATISFIABLE AND UNSATISFIABLE SET OF FORMULAE

Let $\Phi \subset F_{obP}$

- Φ is *satisfiable* iff there is an interpretation structure λ, an interpretation I^p of $pref(\Phi)$ on λ and an assignment ASG such that $\lambda,ASG,I^p \vDash f$ for every $f \in \Phi$

  Notation: $\lambda,ASG,I^p \vDash \Phi$
- Φ is *unsatisfiable* iff is not satisfiable.

3.2 Tableaux System *OB*

In this section we present the tableau system ***OB*** for reasoning about the object description $ob=(\Sigma=(ACT,OBS),F=\{a1,\ldots,an\})$. In the tableau system ***OB*** each tableau is a tree such that each node is labelled with a finite subset of F_{obP} and it is built following the ***OB***-rules presented below. A tree t is a *tableau for Φ* iff Φ is the label of the root node of t. If β is a branch and υ a node of β then Φ_β, Φ_υ denote the union of the labels of the nodes in β and the label of υ, respectively.

We will present the ***OB***-rules in the usual way:

$$\begin{array}{l} \Phi\ ,\ f1,\ldots,fn \\ \big|\ \ \text{rule 1} \\ g1 \\ \ldots \\ gk \end{array} \qquad\qquad \begin{array}{lcll} \Phi\ ,\ f1,\ldots,fn & & & \\ \big| & \ldots & \big| & \text{rule 2} \\ g11 & & gm1 & \\ \ldots & & \ldots & \\ g1k_1 & & gmk_m & \end{array}$$

meaning that if $\Phi\cup\{f1,\ldots,fn\}$ is the union of the labels of all nodes of a branch of the tableau then we can create one successor node of the leaf of the branch (rule 1) labelled with $\{g1,\ldots,gk\}$, or m successor nodes of the leaf of the branch (rule 2) with $\{gi1,\ldots,gik_i\}$ labelling node i, $1\leq i\leq m$. The arity of a rule is the number of successor nodes added. The set $\{f1,\ldots,fn\}$ is the set of *principal formulae* of rules 1 and 2. The set $\{g1,\ldots,gk\}$ is the set of *derived formulae* of rule 1 and $\{gi1,\ldots,gik_i\}$ is the ith set of *derived formulae* of rule 2. We say that a rule *can be applied* to a branch β of the tableau iff the set of principal formulae of the rule is a subset of Φ_β. We say that a rule is *appliable to a formula* (*set of formulae*) if it is a principal formula (the set of principal formulae) of the rule.

Next we present the tableau construction rules, ***OB***-rules and the definition of closed branch and closed tableau.

DEFINITION 3.2.1: ***OB***-RULES

Let Φ be a finite subset of F_{obP} , $\sigma\in Sq$ and ϑ_a a valuation formula for an action symbol a. The ***OB***-rules are the following:

First Order Rules

$$\begin{array}{c} \Phi,\ p{:}\neg\neg f \\ \big|\ \neg\neg \\ p{:}f \end{array} \qquad \begin{array}{c} \Phi,\ p{:}f1\wedge f2 \\ \big|\ \wedge \\ p{:}f1 \\ p{:}f2 \end{array} \qquad \begin{array}{ccc} & \Phi,\ p{:}\neg(f1\wedge f2) & \\ \big| & \neg\wedge & \big| \\ p{:}\neg f1 & & p{:}\neg f2 \end{array}$$

$$\begin{array}{c} \Phi,\ p{:}\neg\exists xf \\ \big|\ \neg\exists \\ p{:}f[x\,|\,t] \end{array} \qquad\qquad \begin{array}{c} \Phi,\ p{:}\exists xf \\ \big|\ \exists \\ p{:}f[x\,|\,y] \end{array}$$

Conditions:
t is a term free for x in f

Conditions:
y is a variable not free on $\Phi\cup\{p{:}\exists xf\}$

$$\begin{array}{c} \Phi \\ \big|\ \text{Ref} \\ p{:}t=t \end{array} \qquad \begin{array}{c} \Phi,\ p{:}\ t1=t2 \\ \big|\ \text{Sim} \\ p{:}t2=t1 \end{array} \qquad \begin{array}{c} \Phi,\ p{:}t1=t2,\ p{:}t2=t3 \\ \big|\ \text{Tr} \\ p{:}t1=t3 \end{array}$$

Conditions:
t is any term, p is any prefix

$$\begin{array}{l} \Phi,\ p: t1=r1,\ldots,p: tn=rn \\ \quad \mid\ \text{Sub i} \\ p:op(t1,\ldots,tn)=op(r1,\ldots,rn) \end{array}$$

Conditions:
$op \in OP_{s1,\ldots,sn,s}$,
t1,...,tn are terms of sorts
s1,...,sn,respectively, $s \neq bool$

$$\begin{array}{l} \Phi,\ p: t1=r1,\ldots,p: tn=rn \\ \quad \mid\ \text{Sub ii} \\ p:\xi(t1,\ldots,tn)\Rightarrow\xi(r1,\ldots,rn) \end{array}$$

Conditions:
$\xi \in OP_{s1,\ldots,sn,bool}$ or $\xi \in \Diamond OBS_s$, n=1, or
$\xi \in \Diamond ACT_{s1,\ldots,sn} \cup \nabla ACT_{s1,\ldots,sn}$
t1,...,tn are terms of sorts s1,...,sn,respectively

Temporal Rules

$$\begin{array}{l} \Phi,\ p:Xf \\ \quad \mid\ \text{Xi} \\ p \circ q \\ q:f \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Xf\}$,
$Sq \cap \Phi = \emptyset$

$$\begin{array}{l} \Phi,\ \sigma_1\ldots\sigma_j \circ q\ldots\sigma_r,\ p:Xf \\ \quad \mid\ \text{Xii} \\ q:f \end{array}$$

Conditions:
$v(pref(\sigma_j))=v(p)$

$$\begin{array}{l} \Phi,\ p:\neg Xf \\ \quad \mid\ \neg\text{Xi} \\ p \circ q \\ q:\neg f \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Xf\}$,
$Sq \cap \Phi = \emptyset$

$$\begin{array}{l} \Phi,\ \sigma_1\ldots\sigma_j \circ q\ldots\sigma_r,\ p:\neg Xf \\ \quad \mid\ \neg\text{Xii} \\ q:\neg f \end{array}$$

Conditions:
$v(pref(\sigma_j))=v(p)$

$$\begin{array}{l} \Phi,\ p:(Ff) \\ \quad \mid\ \text{Fi} \\ pq \\ q:f \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Ff\}$,
$Sq \cap \Phi = \emptyset$

$$\begin{array}{ccccccc} & & \Phi,\ \sigma,\ p:(Ff) & & & & \\ \mid & \cdots & \mid & \mid & \cdots & \mid & \text{Fii} \\ p_1:f & & p_{r_p}:f & \sigma^1 & & \sigma^{r_\sigma} & \\ & & & q:f & & q:f & \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Ff,\sigma\}$, v(p) is a prefix of $v(\sigma) \in Sq$
$pref(comp_f(\sigma,p))=\{p_1,\ldots,p_{r_p}\}$ and
$ext_f(\sigma,p,q)=\{\sigma^1,\ldots,\sigma^{r}\sigma\}$

$$\begin{array}{l} \Phi,\ p:(Pf) \\ \quad \mid\ \text{Pi} \\ qp \\ q:f \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Pf\}$,
$Sq \cap \Phi = \emptyset$

$$\begin{array}{ccccccc} & & \Phi,\ \sigma,\ p:(Pf) & & & & \\ \mid & \cdots & \mid & \mid & \cdots & \mid & \text{Pii} \\ p_1:f & & p_{r_p}:f & \sigma^1 & & \sigma^{r_\sigma} & \\ & & & q:f & & q:f & \end{array}$$

Conditions:
q is free on $\Phi \cup \{p:Pf,\}$, v(p) is a prefix of $v(\sigma) \in Sq$
$pref(comp_p(\sigma,p))=\{p_1,\ldots,p_{r_p}\}$ and
$ext_p(\sigma,p,q)=\{\sigma^1,\ldots,\sigma^{r}\sigma\}$

$$\begin{array}{c} \Phi, \sigma, p{:}\neg Ff \\ \big| \;\neg F \\ q{:}\neg f \end{array}$$

Conditions:
q is accessible from p in $\sigma \in Sq$

$$\begin{array}{c} \Phi, \sigma, p{:}\neg Pf \\ \big| \;\neg P \\ q{:}\neg f \end{array}$$

Conditions:
p is accessible from q in $\sigma \in Sq$

Action/Observation Rules

$$\begin{array}{c} \Phi,\ p{:}\Diamond o(t),\ p{:}\Diamond o(r) \\ \big| \;\Diamond \text{obs i} \\ p{:}t{=}r \end{array}$$

Conditions:
$o \in OBS_s$

$$\begin{array}{c} \Phi,\ \sigma,\ p{:}\Diamond o(t) \\ \big| \;\Diamond \text{obs ii} \\ q{:}\exists x \Diamond o(x) \end{array}$$

Conditions:
$o \in OBS_s$
q is accessible from p in σ

$$\begin{array}{c} \Phi,\ p{:}\nabla a(t1,\ldots,tn) \\ \big| \;\nabla \Diamond \\ p{:}\Diamond a(t1,\ldots,tn) \end{array}$$

Conditions:
$a \in ACT_{s1,\ldots,sn}$

$$\begin{array}{c} \Phi, p{:}\nabla a(t1,\ldots,tn),\ p{:}\nabla a(r1,\ldots,rn) \\ \big| \;\nabla \text{act i} \\ p{:}t1{=}r1,\ldots,p{:}tn{=}rn \end{array}$$

Conditions:
$a \in ACT_{s1,\ldots,sn}$

$$\begin{array}{ccc} & \Phi,\ \sigma_1\ldots\sigma_r & \\ \big| & \ldots & \big| \;\nabla \text{act ii} \\ p{:}\exists\vec{y}\,\nabla a_1(\vec{y}) & & p{:}\exists\vec{y}\,\nabla a_n(\vec{y}) \end{array}$$

Conditions:
$a_1,\ldots,a_n$ are all the action symbols of the object description signature,
p is a prefix of $\sigma_1\ldots\sigma_r$, $p \neq v(\sigma_1)$

Sq-Rules

$$\begin{array}{ccc} & \Phi,\ \sigma_1\ldots\sigma_r & \\ \big| & \text{Si} & \big| \\ \sigma_1\ldots\sigma_i \circ \sigma_{i+1}\ldots\sigma_r & & \sigma_1\ldots\sigma_i\, p \circ \sigma_{i+1}\ldots\sigma_r \end{array}$$

Conditions:
p is free on $\Phi\cup\{\sigma_1\ldots\sigma_r\}$ and $\sigma_{i+1} \in Pr$

$$\begin{array}{ccc} & \Phi,\ \sigma_1\ldots\sigma_r & \\ \big| & \text{Sii} & \big| \\ \omega_k{}^{i}\sigma_2\ldots\sigma_r & & p \circ \sigma_1\ldots\sigma_r \end{array}$$

Conditions:
$\sigma_1{=}\omega_k$, p is free in $\Phi\cup\{\sigma_1\ldots\sigma_r\}$

$$\begin{array}{ccc} & \Phi,\ \sigma_1\ldots\sigma_r & \\ \big| & \text{Siii} & \big| \\ \sigma_1\ldots\sigma_i \circ \sigma_{i+1}\ldots\sigma_r & & \sigma_1\ldots\sigma_i \circ p\sigma_{i+1}\ldots\sigma_r \end{array}$$

Conditions:
p is free on $\Phi\cup\{\sigma_1\ldots\sigma_r\}$ and $\sigma_{i+1} \in Pr$

$$\begin{array}{c} \Phi,\ \sigma_1\ldots\sigma_r \\ \big| \;\text{Siv} \\ \sigma_1\ldots\sigma_r \circ p \end{array}$$

Conditions:
p is free in $\Phi\cup\{\sigma_1\ldots\sigma_r\}$

From the rules presented above we can infer rules related with the abbreviations introduced above. We have the usual rules $\vee$, $\neg\vee$, $\Rightarrow$, $\neg\Rightarrow$, $\Leftrightarrow$, $\neg\Leftrightarrow$, $\forall$ and $\neg\forall$. The rules $\neg$Gi, $\neg$Gii, $\neg$Hi, $\neg$Hii, G and H are similar to the rules Fi, Fii, Pi, Pii, $\neg$F and $\neg$P, respectively. We consider also the following rules related with the other abbreviations introduced.

Valuation

$$\Phi\ ,\ p{:}\nabla a(\vec{t}),\ p{:}\,[a(\vec{x})]\rightarrow o_1(\phi_1(\vec{x})),\ldots,o_m(\phi_m(\vec{x}))$$

$$\Big|\ \text{val i}$$

$$p{:}X(\Diamond o_1(\phi_1(\vec{t}))\wedge\ldots\wedge\Diamond o_m(\phi_m(\vec{t})))$$

Conditions:
ϑ_a has no invariants

$$\begin{gathered}\Phi,\ \sigma,\ p'{:}G(\vartheta_{a1}),\\ p{:}\nabla a_1(\overrightarrow{t1}),\\ q{:}\Diamond o_1(r_1),\ldots,q{:}\Diamond o_{k1}(r_{k1}))\\ q{:}\nabla a_2(\overrightarrow{t2}),\\ q{:}\ [a_2(\vec{x})]o_1(y_1),\ldots,o_{k1}(y_{k1})\rightarrow\\ o_1(\phi_1(\vec{x},y_1)),\ldots,o_{k1}(\phi_{k1}(\vec{x},y_{k1})),o_{k1+1}(\phi_{k1+1}(\vec{x})),\ldots,o_{k2}(\phi_{k2}(\vec{x}))\end{gathered}$$

$$\Big|\ \text{val ii}$$

$$q{:}X(\Diamond o_1(\phi_1(\overrightarrow{t2},r_1))\wedge\ldots\wedge\Diamond o_{k1}(\phi_{k1}(\overrightarrow{t2},r_{k1}))\wedge\Diamond o_{k1+1}(\phi_{k1+1}(\overrightarrow{t2}))\wedge\ldots\wedge\Diamond o_{k2}(\phi_{k2}(\overrightarrow{t2})))$$

Conditions:
In σ q is accessible from p and p is accessible from p'
ϑ_{a1} has neither invariants nor context

These two rules are related with the valuation formulae. Rule val i deals with valuation formulae having neither invariants nor context. Rule val ii deals with generic valuation formulae. As we will see below in the examples, typically, $p{:}\nabla a_1(\overrightarrow{t1})$ represents the occurrence of a birth action affecting all the observations as we have also $p'{:}G(\vartheta_{a1})$ and ϑ_{a1} has neither invariants nor context.

$$\begin{gathered}\Phi,\ \sigma,\ p'{:}G(\vartheta_{a1})\\ p{:}\nabla a_1(\overrightarrow{t1}),\\ q{:}\nabla a_2(\overrightarrow{t2}),\ q{:}\vartheta_{a2},\\ q{:}\Diamond o(r)\end{gathered}$$

$$\Big|\ \text{inv}$$

$$q{:}X\Diamond o(r)$$

Conditions:
In σ q is accessible from p e p is accessible from p'
ϑ_{a1} has neither invariants nor context and o is an invariant of ϑ_{a2}

This rule is important to deal with enabled observations after the occurrence of an action when this action does not affect the observation (i.e. the observation symbol is an invariant of the corresponding valuation formula). The comments above about $p{:}\nabla a_1(\overrightarrow{t1})$ are also appliable here.

$$\begin{array}{c} \Phi, \sigma, \ p':G(\vartheta_{a1}), \\ p:\nabla a_1(\overrightarrow{t1}) \\ \Big| \ \ \Diamond \ \text{obs iii} \\ q:\Diamond o(y) \end{array}$$

Conditions:
In σ q is accessible from p e p is accessible from p'
ϑ_{a1} has neither invariants nor context
y is a variable not free in $\Phi\cup\{\sigma, p:\nabla a_1(\overrightarrow{t1}), p':G(\vartheta_{a1})\}$

This rule is also related with enabled observations. Having in mind the above comments about $p:\nabla a_1(\overrightarrow{t1})$, $p':G(\vartheta_{a1})$ and the definition of interpretation structure it turns out that in any state after the occurrence of the a_1-action an o-observatiom must be enabled. The application of this rule can be important before the application of val ii or inv rules in order to introduce enabled observations.

Birth

$$\begin{array}{c} \Phi, \ \sigma, \ p:G\text{※}a_1(\vec{x}), \ q:\nabla a_2(\vec{t}) \\ \Big| \ \ \text{※i} \\ q:P\exists\vec{y}\nabla a_1(\vec{y}) \end{array}$$

Conditions:
In σ q is accessible from p
$a_1 \neq a_2$

$$\begin{array}{c} \Phi, \ \sigma, \ p':G\text{※}a(\vec{x}), \ p:\nabla a(\vec{t}) \\ \Big| \ \ \text{※ii} \\ q:\neg\Diamond o(r) \end{array}$$

Conditions:
In o p is accessible from p'
q=p or in σ p is accessible from q
o is an observation symbol

$$\begin{array}{c} \Phi, \ \sigma, \ p':G\text{※}a(\vec{x}), \ p:\nabla a(\vec{t}) \\ \Big| \ \ \text{※iii} \\ q:\neg\nabla a(\vec{r}) \end{array}$$

Conditions:
In σ q is accessible from p and
p is accessible from p'

$$\begin{array}{c} \Phi, \ \sigma, \ p':G\text{※}a_1(\vec{x}), \ p:\nabla a_1(\vec{t}) \\ \Big| \ \ \text{※iv} \\ q:\neg\nabla a_2(\vec{r}) \end{array}$$

Conditions:
In σ p is accessible from p'
q=p or in σ p is accessible from q
$a_1 \neq a_2$

These rules are related with birth formulae. Rules※i and ※iv are related with the imposed condition that a birth action must precede all other actions, rule ※ii expresses the condition that no observation is enabled before the occurrence of a birth action (or when it occurs) and rule ※iii indicates that a birth action occurs only once.

DEFINITION 3.2.2: CLOSED BRANCHES AND CLOSED TABLEAUX

A branch β in a tableau is *closed* iff one of the following conditions holds

- $\{p:f, q:\neg f\} \subseteq \Phi_\beta$ and $v(p)=v(q)$
- $\{p:\nabla a(v1,\ldots,vn)\} \subseteq \Phi_\beta$ for some $a\in ACT_{s1,\ldots,sn}$ and p is an initial prefix or there is an initial prefix $q\in$ *pref*(Φ_β) such that $v(p)=v(q)$
- $\{p:\Diamond a(v1,\ldots,vn)\} \subseteq \Phi_\beta$ for some $a\in ACT_{s1,\ldots,sn}$ and p is an initial prefix or there is an initial prefix $q\in$ *pref*(Φ_β) such that $v(p)=v(q)$
- $\{p:\Diamond o(v)\} \subseteq \Phi_\beta$ for some $o\in OBS_s$ and p is an initial prefix or there is an initial prefix $q\in$ *pref*(Φ_β) such that $v(p)=v(q)$

- $\{p{:}Pf\} \subseteq \Phi_\beta$ and p is an initial prefix or there is an initial prefix $q \in \mathit{pref}(\Phi_\beta)$ such that $v(p)=v(q)$
- $\{\sigma=\sigma_1...\sigma_r\} \subseteq \Phi_\beta$ and there is σ_i, $1<i\leq r$, such that $\mathit{pref}(\sigma_i)$ is an initial prefix.

A tableau is *closed* iff all branches are closed.

Herein we do not include forms of closing tableaux related to arithmetic, equational and inequational reasoning which were essential when automating proofs. We have recently developed a theorem prover prototype based on the tableau system presented here which includes a module for arithmetic, equational and inequational reasoning following the work of [Nelson and Oppen 79,Nelson and Oppen 80].

3.3 Soundness

The tableau system presented above is sound wrt to unsatisfiable finite subsets of F_{obP} in the sense that if there is a closed tableau for Φ then Φ is unsatisfiable. As consequence, if, for instance, $\Phi=\{p{:}\neg f\}$ we can conclude that f is a valid formulae, i. e., f is true at every world of every interpretation structure for *ob*. If $\Phi=\{p{:}a1,...,p{:}an\}$, being a1,...,an all the axioms of *ob* and p an initial prefix, then *ob* is an inconsistent object description as there is no interpretation structure such that every axiom is initially true on it, i. e., *ob* as no models. If $\Phi=\{p{:}a1,...,p{:}an,p{:}\neg Gf\}$ and p is also an initial prefix then we can conclude that f is *ob*-valid. The following propositions establish these results. The proofs are long but straightforward and for lack of space we do not present them here.

PROPOSITION 3.3.1:
Let β be a branch of a tableau t, r a $\mathcal{OB}$-rule of arity m that can be applied to β and, for $1\leq i\leq m$, $\{gi1,...,gik_i\}$ the ith set of derived formulae of r. If Φ_β is satisfiable then at least one of the sets $\Phi_\beta \cup \{gi1,...,gik_i\}$, $1\leq i\leq m$, is satisfiable.

PROPOSITION 3.3.2:
Let t be a tableau for Φ and β a branch of t
(i) if β is closed then Φ_β is unsatisfiable
(ii) if t is closed then Φ is unsatisfiable.

PROPOSITION 3.3.3:
Consider the object description $ob=(\Sigma=(ACT,OBS),F=\{a1,...,an\})$, t a closed tableau for $\Phi\subset F_{obP}$ and p an initial prefix
(i) if $\Phi=\{p{:}a1,...,p{:}an\}$ then *ob* is an inconsistent object description
(ii) if $\Phi=\{p{:}a1,...,p{:}an,p{:}\neg Gf\}$ then f is *ob*-valid
(iii) if $\Phi=\{p{:}a1,...,p{:}an,p{:}\neg f\}$ then f is *ob*-initially valid
(iv) if $\Phi=\{q{:}\neg f\}$ then f is initially valid if q is an initial prefix and f is valid otherwise.

Next we present some examples of tableaux within the context of the object description *employee* introduced above.

In order to make tableaux more readable, we assign a number to formulae and we indicate the rule considered in the construction of each new node and the numbers of its principal formulae. Each closed branch is marked with the symbol $\times$ and the formulae that close a branch are printed in bold.

EXAMPLE 3.3.4

With the tableau system ***BOOK*** we can reason about the object description $book=(\Sigma_{book},\{(a1),\ldots,(a6)\})$ given above. Consider the following tableau:

$w_1{}^i{:}(a1),\ldots,w_1{}^i{:}(a6)$

$w_1{}^i{:}\neg G(\neg(\exists x \nabla tak(x) \wedge \exists x \nabla ret(x)))$ (1)

$\mid\ \neg G(i)\ (1)$

$w_1{}^i\, w_2$ (2)

$w_2{:}\neg\neg(\exists x \nabla tak(x) \wedge \exists x \nabla ret(x))$ (3)

$\mid\ \neg\neg\ (3)$

$w_2{:}\exists x \nabla tak(x) \wedge \exists x \nabla ret(x)$ (4)

$\mid\ \wedge\ (4)$

$w_2{:}\exists x \nabla tak(x)$ (5)

$w_2{:}\exists x \nabla ret(x)$ (6)

$\mid\ \exists\ (5)$

$w_2{:}\nabla tak(t1)$ (7)

$\mid\ \exists\ (6)$

$w_2{:}\nabla ret(t2)$ (8)

$\mid\ \nabla\diamond\ (7)$

$\mathbf{w_2{:}\diamond tak(t1)}$

$\mid\ \nabla\diamond\ (8)$

$w_2{:}\diamond ret(t2)$

$\mid\ G\ (2)\ (a5)$

$w_2{:}\forall x(\diamond tak(x) \Rightarrow \diamond avail(true))$ (9)

$\mid\ G\ (2)\ (a6)$

$w_2{:}\forall x(\diamond ret(x) \Rightarrow \diamond avail(false))$ (10)

$\mid\ \forall\ (9)$

$w_2{:}\diamond tak(t1) \Rightarrow \diamond avail(true)$ (11)

$\mid\ \forall\ (10)$

$w_2{:}\diamond ret(t2) \Rightarrow \diamond avail(false))$ (12)

$\mid\ \Rightarrow (11)\ \mid$

$\mathbf{w_2{:}\neg\diamond tak(t1)}$ $\times$ $\qquad$ $w_2{:}\diamond avail(true)$ (13)

$\mid\ \Rightarrow (12)\ \mid$

$w_2{:}\diamond avail(false)$ (14) $\qquad$ $\mathbf{w_2{:}\neg\diamond ret(t2)}$ $\times$

$\mid\ \diamond obs\ (13)\ (14)$

$\mathbf{true=false}$

$\times$

Note that the second branch is closed because it includes the formula true=false which is contradictory in the data type theory we adopt. In the theorem prover prototype this branch is considered closed by the module which deals with the arithmetic, equational and inequational reasoning.

By proposition 3.3.3, as we have built a closed tableau for the set of formulae

$$\{\omega_1{}^{i}{:}(a1),\ldots,\omega_1{}^{i}{:}(a6),\omega_1{}^{i}{:}\neg G((\neg(\exists x\nabla tak(x)\wedge\exists x\nabla ret(x)))\}$$

we proved that the formula $\neg(\exists x\nabla tak(x)\wedge\exists x\nabla ret(x)))\}$ is *book*-valid which means that a book can not be taken and returned at the same time.

EXAMPLE 3.3.5:

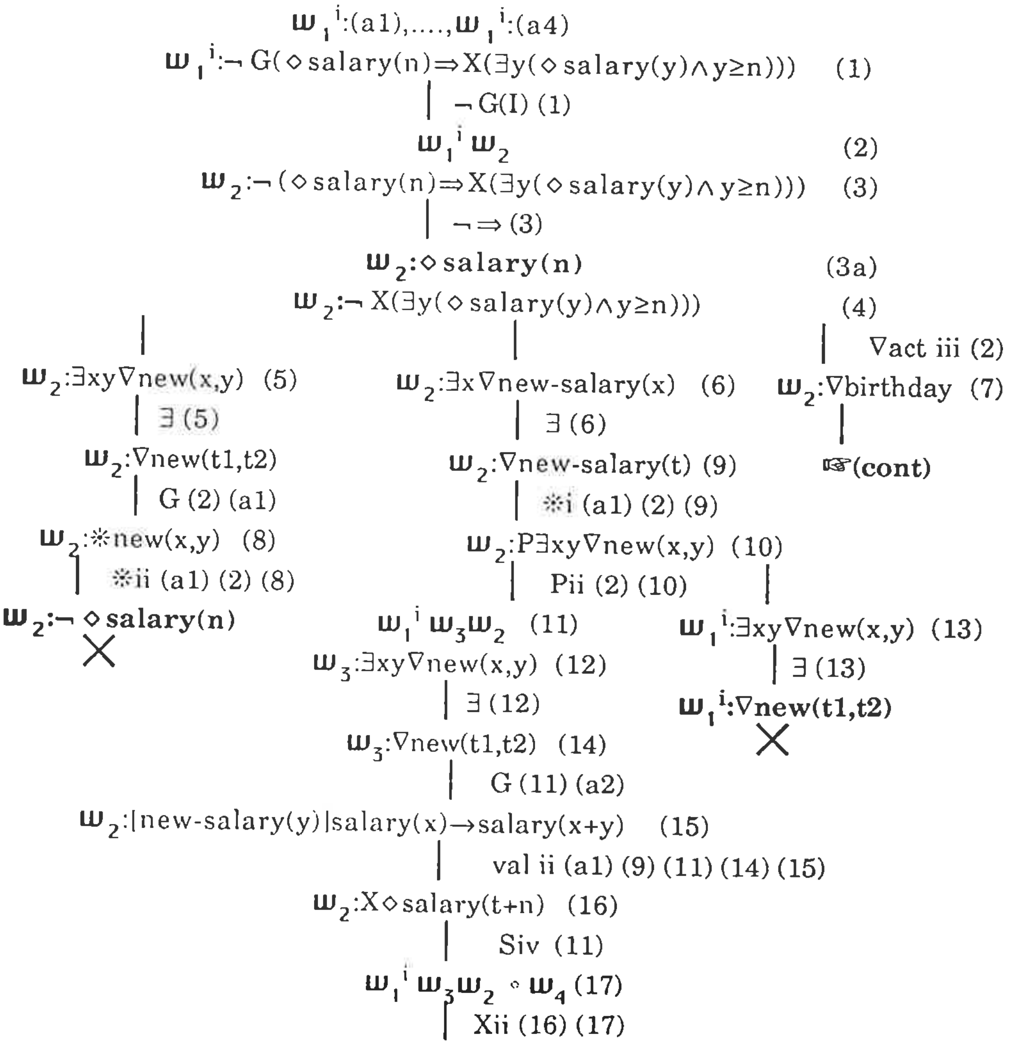
Let us consider a more elaborated example: the object description *employee*=(Σ_{emp},*F*) with Σ_{emp}=(ACT,OBS)

$ACT_{nat\ nat}=\{new\}$
$ACT_{nat}=\{new\text{-}salary,birthday\}$
$OBS_{nat}=\{salary,age\}.$

and

F= $\{G(\text{※}new(x,y)),$ (a1)
$G([new\text{-}salary(y)]salary(x)\rightarrow salary(x+y)),$ (a2)
$G([new(x,y)]\rightarrow salary(x)\wedge age(y)),$ (a3)
$G([birthday]age(x)\rightarrow age(x+1))$ (a4)}

With the tableau system *EMPLOYEE* we can reason about the object description *employee*. Consider the following tableau



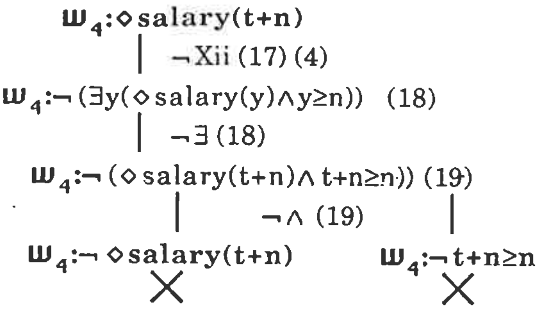
$\omega_1{}^{i}{:}(a1),\ldots,\omega_1{}^{i}{:}(a4)$
$\omega_1{}^{i}{:}\neg G(\diamond salary(n)\Rightarrow X(\exists y(\diamond salary(y)\wedge y\geq n)))$ (1)
| $\neg G(I)$ (1)
$\omega_1{}^{i}\,\omega_2$ (2)
$\omega_2{:}\neg(\diamond salary(n)\Rightarrow X(\exists y(\diamond salary(y)\wedge y\geq n)))$ (3)
| $\neg\Rightarrow$ (3)
$\omega_2{:}\diamond salary(n)$ (3a)
$\omega_2{:}\neg X(\exists y(\diamond salary(y)\wedge y\geq n)))$ (4)

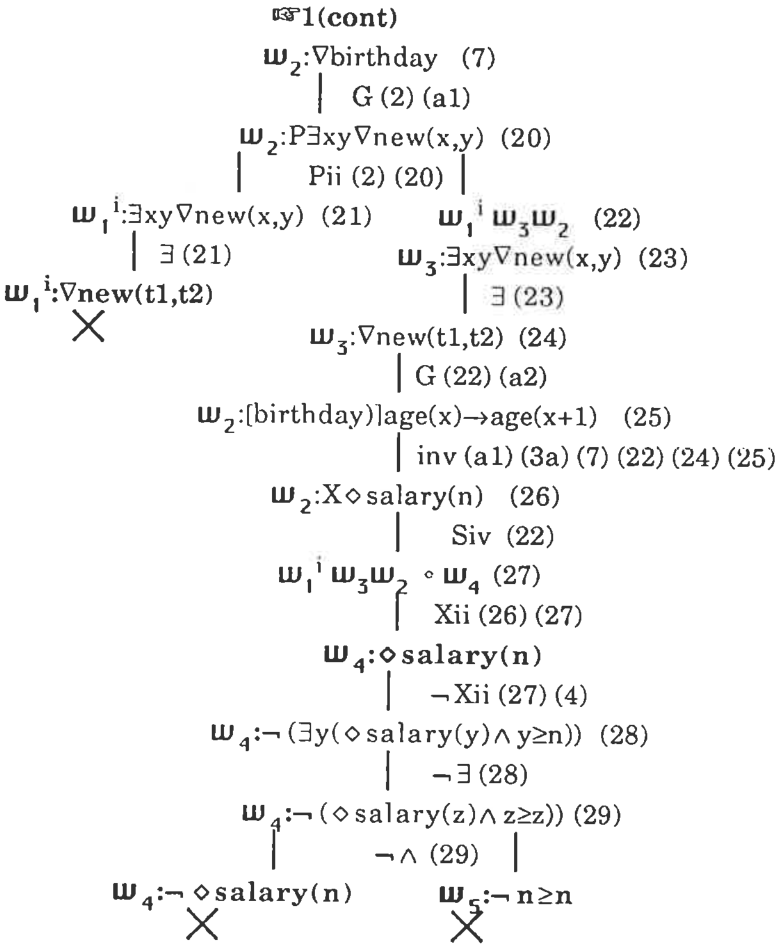
$\omega_2{:}\exists xy\nabla new(x,y)$ (5)
| $\exists$ (5)
$\omega_2{:}\nabla new(t1,t2)$
| G (2) (a1)
$\omega_2{:}\text{※}new(x,y)$ (8)
| ※ii (a1) (2) (8)
$\omega_2{:}\neg\diamond salary(n)$
×

$\omega_2{:}\exists x\nabla new\text{-}salary(x)$ (6)
| $\exists$ (6)
$\omega_2{:}\nabla new\text{-}salary(t)$ (9)
| ※i (a1) (2) (9)
$\omega_2{:}P\exists xy\nabla new(x,y)$ (10)
| Pii (2) (10)
$\omega_1{}^{i}\,\omega_3\omega_2$ (11)
$\omega_3{:}\exists xy\nabla new(x,y)$ (12)
| $\exists$ (12)
$\omega_3{:}\nabla new(t1,t2)$ (14)
| G (11) (a2)
$\omega_2{:}[new\text{-}salary(y)]salary(x)\rightarrow salary(x+y)$ (15)
| val ii (a1) (9) (11) (14) (15)
$\omega_2{:}X\diamond salary(t+n)$ (16)
| Siv (11)
$\omega_1{}^{i}\,\omega_3\omega_2\circ\omega_4$ (17)
| Xii (16) (17)

|
$\omega_1{}^{i}{:}\exists xy\nabla new(x,y)$ (13)
| $\exists$ (13)
$\omega_1{}^{i}{:}\nabla new(t1,t2)$
×

| ∇act iii (2)
$\omega_2{:}\nabla birthday$ (7)
|
☞**(cont)**




☞1(cont)
ω_2:∇birthday (7)
G (2) (a1)
ω_2:P∃xy∇new(x,y) (20)
Pii (2) (20)
ω_1^i:∃xy∇new(x,y) (21)
ω_1^i $\omega_3\omega_2$ (22)
∃ (21)
ω_3:∃xy∇new(x,y) (23)
ω_1^i:∇new(t1,t2)
∃ (23)
ω_3:∇new(t1,t2) (24)
G (22) (a2)
ω_2:[birthday)]age(x)→age(x+1) (25)
inv (a1) (3a) (7) (22) (24) (25)
ω_2:X◇salary(n) (26)
Siv (22)
ω_1^i $\omega_3\omega_2$ ∘ ω_4 (27)
Xii (26) (27)
ω_4:◇salary(n)
¬Xii (27) (4)
ω_4:¬(∃y(◇salary(y)∧y≥n)) (28)
¬∃ (28)
ω_4:¬(◇salary(z)∧z≥z)) (29)
¬∧ (29)
ω_4:¬◇salary(n)
ω_5:¬n≥n

Note that the third branch is closed because it includes the formula $\neg t+n\geq n$ which is contradictory in the natural number theory. In the theorem prover prototype this branch is considered closed by the module which deals with the arithmetic, equational and inequational reasoning. A similar comment applies to the branch with the formula $\neg n\geq n$.

The tableau for

$$\{\omega_1^i:(a1),....,\omega_1^i:(a4),\omega_1^i:\neg G(\diamond salary(n)\Rightarrow X(\exists y(\diamond salary(y)\wedge y\geq n)))\}$$

presented above is closed so, by proposition 3.3.3, the formula $(\Diamond \text{salary}(n) \Rightarrow X(\exists y(\Diamond \text{salary}(y) \wedge y \geq n)))$ is *employee*-valid which means that the description ensures that the salary of every employee must never decrease.

4. Interaction

Consider again the library application. We have already presented the object description *book*. Let us consider now the object description $user=(\Sigma_{user}, \{(u1),\ldots,(u5)\})$ where Σ_{user}=(ACT,OBS) with

$\text{ACT}_{\varepsilon}=\{\text{new}\}$ $\qquad\qquad$ $\text{OBS}_{\text{setbook}}=\{\text{books}\}$

$\text{ACT}_{\text{book}}=\{\text{taks, rets}\}$

and

$G(※\text{new})$ (u1)

$G([\text{new}]\rightarrow \text{books}(\varnothing))$ (u2)

$G([\text{taks}(x)]\text{books}(s)\rightarrow \text{books}(s\cup\{x\}))$ (u3)

$G([\text{rets}(x)]\text{books}(s)\rightarrow \text{books}(s\setminus\{x\}))$ (u4)

$G(\forall x(\nabla \text{taks}(x)\Rightarrow F\nabla \text{rets}(x)))$ (u5).

The above description states that each user can take and return books and keeps record of the books currently borrowed.

Borrowing books from the library can be seen from two different points of view. Consider the book b and the user u. On one hand the book b is taken by the user u, i. e., the action tak(u) occurs within the context of book b. On the other hand user u takes book b, i. e., the action taks(b) occurs within the context of user u. As consequence an interaction is established between book b and user u. This interaction can be established trough the formula $G((\nabla b.\text{tak}(u)\Leftrightarrow \nabla u.\text{taks}(b)))$ called an *interaction formula*. Interaction formulae are built prefixing each action with the correspondent object identifier and establishing an equivalence (or implication) between the occurrence of these actions of the two objects.

Returning books to the library is a similar situation of interaction between books and users that can be established trough the formulae $G((\nabla b.\text{ret}(u)\Leftrightarrow \nabla u.\text{rets}(b)))$.

We can use tableau systems to reason about objects in the presence of interactions, i.e., we can use tableau systems to reason about the conceptual schema itself. These tableau systems are similar to the ones introduced above but we consider also the interaction formulae and in the axioms of the objects involved each action symbol and each observation symbol is prefixed with the correspondent object identifier.

Consider the following tableau where b.(ai), $1\leq i\leq 6$ and u.(uj), $1\leq j\leq 5$, are the axioms of the object description *book* and *user* presented above where each action symbol and each observation symbol involved is prefixed with the correspondent object identifier.

ω_1^i:b.(a1),....,ω_1^i:b.(a6)

ω_1^i:u.(u1),....,ω_1^i:u.(u5)

$\omega_1^i:G((\nabla b.tak(u)\Leftrightarrow\nabla u.taks(b)))$ (i1),

$\omega_1^i:G((\nabla b.ret(u)\Leftrightarrow\nabla u.rets(b)))$ (i2)

$\omega_1^i:\neg G(b.\nabla tak(u)\Rightarrow Fb.\nabla ret(u)))$ (1)

| $\neg G(i)$ (1)

$\omega_1^i\,\omega_2$ (2)

$\omega_2:\neg(b.\nabla tak(u)\Rightarrow Fb.\nabla ret(u)))$ (3)

| $\neg\Rightarrow$ (3)

$\omega_2:b.\nabla tak(u)$

$\omega_2:\neg Fb.\nabla ret(u)$ (4)

| G (i1)

$\omega_2:b.\nabla tak(u)\Leftrightarrow u.\nabla taks(b)$ (5)

| $\Leftrightarrow$ (5) |

$\omega_2:\neg b.\nabla tak(u)$

$\omega_2:\neg u.\nabla taks(b)$

×

$\omega_2:b.\nabla tak(u)$

$\omega_2:u.\nabla taks(b)$

| G (u.a5)

$\omega_2:\forall x(u.\nabla taks(x)\Rightarrow Fu.\nabla rets(x))$ (6)

| $\forall$ (6)

$\omega_2:u.\nabla taks(b)\Rightarrow Fu.\nabla rets(b)$ (7)

| $\Rightarrow$ (7) |

$\omega_2:Fu.\nabla rets(b)$ (8)

| Fii (8)

$\omega_1^i\,\omega_2\omega_3$ (9)

$\omega_3:u.\nabla rets(b)$

| G (i2)

$\omega_3:b.\nabla ret(u)\Leftrightarrow u.\nabla rets(b)$ (10)

| $\Leftrightarrow$ (10) |

$\omega_3:\neg b.\nabla ret(u)$

$\omega_3:\neg u.\nabla rets(b)$

×

$\omega_3:b.\nabla ret(u)$

$\omega_3:u.\nabla rets(b)$

| $\neg F$ (3) (9)

$\omega_3:\neg b.\nabla ret(u)$

×

$\omega_2:\neg u.\nabla taks(b)$

×

This closed tableau assures that, in the presence of the above interactions, in the context of object description *book* if the book b is taken by the user u then it will be returned by the user to the library. Note that it is easy to see that this property would not necessarily be true if we considered only the object description *book* in isolation.

5. Conclusions

In this paper we presented tableau systems for reasoning with objects.Given an object description *ob* and an assertion to be verified we can build a tableau using the rules of the tableaux system ***OB***. We also outlined how we can reason about the conceptual schema itself using the objects as well as interaction mechanisms. Our tableau system is based on the prefixed tableau system for modal logic and incorporates different aspects represented by different collections of rules: the usual rules related to connectives and quantifiers, rules for dealing with the temporal operators, rules related to sequences of states and rules that are specific of objects (related to birth, effects of actions on observations, occurrence of actions and enabling of observations and actions). We have not detailed here the module we used for arithmetic, equational and inequational reasoning in the development of a theorem prover prototype based on the tableau system presented.

Given an object description *ob*, the tableau system ***OB*** associated with it is sound wrt unsatisfiable finite subsets of F_{obP} , i. e., if there a closed tableau for Φ then Φ is an unsatisfiable set. Completness is under current research.

References

[Booch 91]
Booch, *Object-oriented Design*, The Benjamin/Cummings House, 1991.

[Demolombe *et al* 90]
Demolombe, R., Illarramendi, A. and Blanco, J.-M., Semantic Optimization in Data Bases Using Artificial Intelligence Techniques, R. Meersman, Shi, Z. and C. Kung (eds), *The Role of Artificial Intelligence in Databases and Information Systems*, North-Holland, 1990, 519-528

[Emerson 90]
Emerson, E., Temporal and Modal Logic, *Handbook of Theoretical Computer Science*, J. van Leewen (ed), Elsevier Science Publishers, 1990, 997-1072.

[Fitting 83]
Fitting, M., *Proof Methods for Modal and Intuitionistic Logics*, Synthese Library 169, Reidel, 1983.

[Gallaire *et al* 84]
Gallaire, H., Minker, J. and Nicolas, J.-M., Logic Databases: A Deductive Approach, *Computing Surveys*, 16[2], 1984, 153-185.

[Gouveia 92]
Gouveia, P., *Tableaux for Local Reasoning About Objects* (in Portuguese), Master Thesis, Technical University of Lisbon, 1992.

[Gouveia *et al* 93]
Gouveia, P., Sernadas, C., Gomes, J., Apolinário, M-J.,Tableaux for Reasoning about Objects, *Proc. 2nd Workshop on Theorem Proving with Analytic Tableaux and Related Methods*, April 1993, Marseille, France (in print).

[Jungclaus *et al* 91]
Jungclaus, R., Saake, G. and Sernadas, C., Formal Specification of Object Systems, Abramsky, S. and Maibaum, T. (eds), *TAPSOFT'91*.

[Li and SernadasA 91]
Li, R., Sernadas, A., Reasoning About Objects Using Tableau Method, *Journal of Logic and Computation*, 1 (5), North Holland, 1991, 575-611.

[Loucopoulos and Zicari 92]
Loucopoulos, P., Zicari, R. (eds), *Conceptual Modeling Databases and CASE: An Integrated View of Information Systems Development*, John Wiley, 1992.

[Manna and Pnueli 92]
Manna, Z., Pnueli, A., *The Temporal Logic of Active and Concurrent Systems*, Springer Verlag, 1992.

[Nelson and Oppen 79]
Nelson, G., Oppen, D., Simplification by Cooperating Decision Procedures, ACM Transactions on Programming Languages and Systems, 1 (2), October 1979, 245-257.

[Nelson and Oppen 80]
Nelson, G., Oppen, D., Fast Decision Procedures based on Congruence Closure, Journal of the Association for Computing Machinery, 27 (2), April 1980, 356-364.

[Olivé 89]
Olivé, A., *Deductive Conceptual Modeling*, IFIP WG8.1 Meeting, Sesimbra, June, 1989.

[Reiter 84]
Reiter, R., Towards a Logical Reconstruction of Relational Database Theory, M. Brodie, J. Mylopoulos and J. Schmidt (eds), *On Conceptual Modeling*, Springer-Verlag, 1984, 191-233.

[Saake and Lipeck 89]
Saake, G. and Lipeck, U., Using Finite-Linear Temporal Logic for Specifying Database Dynamics, *Proc. CSL'88 2nd Workshop Computer Science Logic*, Borger, E. Kleine Buening, H. and Richter, M., LNCS 385, Springer Verlag, 1989, 288-300.

[SernadasA 80]
Sernadas, A., Temporal Aspects of Logic Procedure Definition, *Information Systems 5*, 1980, 167-187.

[SernadasA *et al* 89]
Sernadas, A., Fiadeiro, J., Sernadas, C., Ehrich, H., The Basic Building Blocks of Information Systems, *Information Systems Concepts: An In-depth Analysis*, Falkenberg, E., Lindgreen, P. (eds), North Holland, 1989, 225-246.

[SernadasA *et al* 91]
Sernadas, A., Ehrich, H., Sernadas, C., What is an Object, After All?, *Object Oriented Databases: Analysis, Design and Construction*, Meersman, R., Kent, W., Khosla, S. (eds), North Holland, 1991, 39-69.

[SernadasA *et al* 92]
Sernadas, A., Sernadas, C., Costa, F., Object Specification Logic, Research Report DMIST/IST, submited, 1992.

[SernadasC *et al* 90]
Sernadas, C., Gouveia, P., Lopes, A., Objects as Structuring Units for Incorporating Dynamics in Deductive Conceptual Modeling, *Proc. International Workshop on the Deductive Approach to Information Systems and Databases*, 1990, 93-110.

[SernadasC *et al* 92]
Sernadas, C., Gouveia, P., Sernadas, A., Refinement: Layered Definition of Conceptual Schemata, Sernadas, C., Gouveia, P. e Sernadas, A., *Information System Concepts*, Falkenberg, E., Rolland, C., El-Sayed, E., (eds), North Holand, 1992, 19-51.

[Wieringa 90]
Wieringa, R., Equational Specification of Dynamic Objects, R. Meersman and B. Kent (eds), *Object-oriented Databases: Analysis. Design and Construction*, North-Holland, 1990.