Università di Pisa

Dipartimento di Informatica

Technical Report : TR-98-01

A Logic-Based Calculus of Fluents

S. Brandano

January 12, 1998

ADDR: Corso Italia 40,56125 Pisa,Italy. TEL: +39-50-887111. FAX: +39-50-887226

A Logic-Based Calculus of Fluents

Sergio Brandano

January 12, 1998

UNIVERSITÀ DEGLI STUDI DI PISA
DIPARTIMENTO DI INFORMATICA

Technical Report: TR-01/98

A Logic-Based Calculus of Fluents

Sergio Brandano[1]

Abstract. This report formally defines the class of all problems on *Reasoning about Actions and Change*, where accurate and complete information about actions, together with strict inertia in continuous time, continuous change and alternative results of possibly concurrent and independent actions are the assumed properties. The intended model set for each member in the class is defined in terms of a model-theoretic trajectory semantics. The case is designated, in the *Features and Fluents* framework, with the $\mathcal{K}$-$\mathbf{RAC}i$ family of reasoning problems. A fix-point characterization of the subclass $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$ is then given in terms of a simulative algebraic semantics, and show which are the difficulties when approaching the full class with that method. A non-simulative algebraic semantics is then presented as an alternative algebraic characterization of the model-theoretic trajectory semantics. Still the characterization is made in terms of complete lattices and continuous operators on those complete lattices. The non-simulative algebraic semantics is shown to be correct and complete with respect to the trajectory-semantics, when applied to reasoning problems in $\mathcal{K}$-$\mathbf{RAC}i$. Finally, adopting the non-simulative algebraic semantics as proof procedure, a logic-based *Calculus of Fluents* is defined, via a meta-theoretic extension of the Horn Clause Logic using the non-ground representation of object level variables. The meta-interpreter consists in the Horn clause representation of the proof procedure. The calculus is shown to be decidable, domain independent and class dependent; its semantics consists in the non-simulative algebraic semantics, and is executable as a logic program by the SLD-resolution rule.

1987 CR Categories: D.3.1, F.3.2, I.2.3, I.2.4
Key words: nonmonotonic temporal reasoning, semantics, logic programming.

January 1998

ADDR: Corso Italia 40 - 56125 Pisa - Italy
TEL:+39-50-887111 - TLX:590291 DIPISAI
E_MAIL:di.unipi.it

[1]Partially supported by the European Community and the University of Pisa with grant under P.I.C. 94-S-3017/11 (ERASMUS Project).

Contents

Chapter 1

Introduction

The search for non-monotonic logics dates back its motivation to [San72], at the genesis of the so called *Formal Artificial Intelligence*. Non-monotonic entailment methods have been proposed in the precise attempt to formalize common sense reasoning when reasoning about actions and change. The established research methodology in the field, until the early 90's, consisted in constructing problems on reasoning about actions and change of a certain peculiar interest, in identifying their general domain knowledge and problem specification, and in developing an intuitively plausible formal language to represent and process that knowledge. Such methodology payed its intrinsic weakness on the ease to refute those logics via counterexamples; the absence of a solid theoretical structure was the most evident ingenuity of the artificial intelligence, or rather its not achieved maturity. Recently, Erik Sandewall [San94a] proposed an alternative and *systematic* research methodology, based on a taxonomy of epistemological and ontological assumptions a reasoning problem could require. Logics for reasoning about actions and change, precisely Shoham's preferential logics [Sho89], may be gathered in congruence classes modulus the particular assumptions they satisfy, in determining their *range of applicability*. For a given logic, to the purpose of the classification, a soundness result is required with respect to the specific underlying semantics that formally defines the target class.

1.1 Contributions

In this thesis we first consider the problem to define an underlying semantics for all the logics for reasoning about actions and change which deal with explicit continuous time, within a certain well precise family of epistemological and ontological assumptions. A previous attempt in this respect is due to Erik Sandewall [San89a, San89b], at the genesis of his Features and Fluents project. The known $\mathcal{K}$-**IA** class of reasoning problems, that is the broadest class in the project's report [San94a], defines the knowledge representation method and the intended model set for each reasoning problem where accurate and complete information about actions, together with *strict inertia in integer* time and alternative results of actions are the assumed properties. In this thesis we consider the advanced case of accurate and complete information about actions, together with *strict inertia in continuous time*, *continuous change* and alternative results of possibly concurrent and independent actions, and we designate the case with the $\mathcal{K}$-**RAC**i class of reasoning problems. In the genuine style of Sandewall's simulative underlying semantics for $\mathcal{K}$-**IA** , the set of all the intended answers for each reasoning problem in $\mathcal{K}$-**RAC**i is built simulating infinite game developments between a trajectory-semantics ego and a trajectory-semantics world. The new underlying semantics subsumes the underlying semantics for $\mathcal{K}$-**IA** , and allows trajectory-semantics worlds with explicit continuous time and a combination of encapsulated and/or non encapsulated actions with discrete and/or continuous change, depending on the need.

We then give an interesting fix-point characterization of the intended model set for scenario descriptions in $\mathcal{K}sp$-**RA**d**C**i , and show which are the difficulties when approaching the full $\mathcal{K}$-**RAC**i class with that method.

As will be clear, underlying and fix-point semantics are model-theoretic semantics. According to their simulative method, the meaning of scenario descriptions is given without get straight to implicit details of any proof procedure of practical use; their methods just provide a detailed description (a specification) for all underlying techniques by explaining "what" should be intended and computed. Naturally the desirable complement of every model-theoretic semantic, even underlying, is a proof-theoretic semantic. In our case, the novel procedural intended interpretation should give the proof mechanism by explaining which is the computational mechanism underlying scenario descriptions interpretation, where scenario descriptions come now together with the query. Concerning preferential logics, while the model-theoretic underlying semantics represent the authoritative reference for their formal semantics, the proof-theoretic underlying semantics may represent an authoritative reference for their design and implementation. In fact, in addition to the current model-theoretic investigations of the range of applicability for various entailment methods, it is also important to develop methods which can serve as a basis for the design and implementation of these logics. In this thesis we develop a novel approach in this respect, based on the algebraic theory of lattices.

We first define a *non-simulative* algebraic semantics as an algebraic characterization of the intended meaning of chronicles in $\mathcal{K}$-**RAC**i where the computational mechanism underlying scenario descriptions interpretation is specified without simulating game developments between a trajectory-semantics ego and a trajectory-semantics world. When reasoning on problems in $\mathcal{K}$-**RAC**i , conclusions from the non-simulative algebraic semantics are shown correct and complete with respect to the intended conclusions from the simulative underlying semantics.

Then a calculus for computing feature values at different points in time is designed and implemented as case study. The Horn Clause Logic, that is pure Positive Logic Programming, is shown to be a Shoham's monotonic preferential logic. The logic fails to include much of the intended models, and an operation of recovery of the model set is then superimposed on its fix-point semantics by means of meta-interpretation. Roughly, the method involves the construction of an *algorithm* consisting of two parts, a *logic part* and a *control part*; the logic part is built by compilation of a scenario description Υ in $\mathcal{K}$-**RAC**i to a set Γ of definite Horn clauses, while the control part consists in the Horn clause representation of the non-simulative algebraic semantics. The logic part, which is declaratively read as the specification of a problem domain, is the object of reasoning for the control part, which is a meta-interpreter. The resulting formalism is executable as a logic program by the SLD-resolution: if Δ is the control part, and Γ is the logic part, then $\Delta \cup \Gamma$ can be combined with a query concerning the value of a feature at a specific point in time, and the execution of $\Delta \cup \Gamma$ as a conventional logic program, together with the query, provides the intended answer. The non-simulative algebraic semantics is, de facto, the formal semantics of the resulting calculus. The Horn Clause Logic, augmented with the *control part*, results to be a Shoham's non-monotonic preferential logic, that performs backward *and* forward temporal reasoning in explicit continuous time. Such logic is class dependent (its range of applicability is $\mathcal{K}$-$\mathbf{RAC}i$), and domain independent. The property of domain independence makes the calculus capable of reasoning on incoming and partially described situations, an evolving schedule, and evolving laws for describing the effects of those actions the agent is allowed to perform.

Finally, we compare the Fluent Calculus with the $\mathcal{A}$ language of Gelfond and Lifschitz, the Abductive Logic Programming with integrity constraints, Doherty's PMON logic, and the Event Calculus of Kowalski. The comparisons, according to the Features and Fluents systematic approach, are made by means of set inclusion with respect to the range of applicability of the compared calculi, and show the advantages and the greater expressive and problem-solving power of the Fluent Calculus.

1.2 Summary

The thesis work is organized as follows.

Chapter 2 is devoted to some preliminaries on Sandewall's Features and Fluents framework, the simulative underlying semantics for $\mathcal{K}$-$\mathbf{IA}$ reasoning problems, and Shoham's general notion of preferential logics.

In chapter 3 we define the $\mathcal{K}$-$\mathbf{RAC}i$ class of reasoning problems by extending the simulative underlying semantics for $\mathcal{K}$-$\mathbf{IA}$ to the case of continuous time and continuous change.

In chapter 4 we define the fix-point characterization of the ego-world game, together with its soundness and completeness result for the class $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$.

In chapter 5 we present a *non-simulative approach* to continuous change via an algebraic semantics, together with its soundness and completeness results with respect to the underlying semantics for $\mathcal{K}$-$\mathbf{RAC}i$.

Chapter 6 applies the algebraic semantics to some classical problems on reasoning about actions and change.

In chapter 7 the relation of Features and Fluents with Logic Programming is investigated. The Horn Clause Logic is shown to be a Shoham's monotonic preferential logic. A calculus is then defined as the Horn Clause Logic augmented with a meta-interpreter implementing the algebraic semantics of chapter 5. The calculus is then assessed to the full $\mathcal{K}$-$\mathbf{RAC}i$ class of reasoning problems, and compared with some of the most representative approaches in the present literature.

Chapter 2

Preliminaries

We present the prime concepts and results of Sandewall's systematic approach, together with the formal definition of the $\mathcal{K}$-**IA** class of reasoning problems. The former is the general approach we adopted for this thesis, while the class will be generalized in chapter 3. We also remind Shoham's general notion of preferential logics, that is the general notion on those logics the whole framework is concerned with. The knowledgeable reader may safely skip this chapter.

2.1 Sandewall's systematic approach

The *Features and Fluents Framework* [San94a] was originally proposed by Erik Sandewall as the reference framework for a new systematic approach to the representation of knowledge about dynamical systems. The new systematic methodology was justified by the difficulties and limitations of the *example-based methodology*. The example-based methodology has been summarized by Davis [Dav90] as follows:

> *The basic approach used here, as in much of the research on automating common-sense reasoning, is to take a number of examples of common-sense inference in a common-sense domain, generally deductive inference; identify the general domain knowledge and the particular problem specification used in the inference; develop a formal language in which this knowledge can be expressed; and define the primitives of the language as carefully and precisely as possible.*

Then Sandewall [San93] arouse rigorous criticism on it:

> *There has been much research in recent years on methods for reasoning about actions and change, and on finding solutions to the so-called "frame problems". New variants of non-monotonic logics for common-sense reasoning have been proposed, only to be quickly refuted by counterexamples. According to the standard research methodology in the area, the evidence in favor of a proposed logic should consist of intuitive plausibility arguments and a small number of scenario examples for which the logic is proven (or claimed) to give the intended conclusions and no others. Clearly there is a need for*

more systematic results, where a proposed logic is verified for a whole class of reasoning problems and not only for single examples.

Sandewall characterized the systematic methodology with the following five major aspects: (1) the use of an *underlying semantics* which provides a general characterization of inhabited dynamical systems; (2) the use of an underlying semantics for a formal definition of what are the *intended models* for a given scenario description; (3) the use of an *ontological taxonomy*, which is strictly defined in terms of the underlying semantics, and which characterizes classes of dynamical systems with specific restrictions on them; (4) explicit statements of *epistemological assumptions*; (5) formally proven *assessments* of the range of applicability of non-monotonic entailment criteria, based on the strict definition of intended models. The systematic approach is given in *model-theoretic* terms.

The following is the underlying semantics that formally defines the broadest class in [San94a], which is properly the $\mathcal{K}$-**IA** class of reasoning problems.

A *system*, or *world*, consists in a collection of *objects* which have individual *properties*, and where there are *relationships* between the participating objects. Such properties and relations are on the whole referred to as *features*, and they are assumed to be observables. Two levels of description for a system are assumed: the *material level* and the *image level*. The material level consists in the description of the physical system, or can be even identified with it, and uses continuous time and continuous-valued state variables that change according to the laws of (classical) physics. The image level characterizes the system's present state as the abstract representation of the material level, it models the way the world is perceived by the agent and is the basis for reasoning about the system. The image level uses discrete-valued state variables that change according to a chosen assumption of inertia and peculiar laws for actions. A *time-point* is formally defined to be a non negative number expressing a point in time, its domain is denoted as $\mathcal{T}$ and is identified with the set of all natural numbers (zero is assumed to belong the set). Examples of features are "the gun is *loaded*", "the turkey is *alive*", or "the block b_i is *on* the block b_j". Objects bringing a particular property or involved in a relation, like $loaded(gun\_1)$, or $on(block\_i, block\_j)$ and so on, are arguments for features; features without arguments are referred to as *global features*. The domain of all feature names is denoted as $\mathcal{F}$ ("*loaded*", "*alive*", etc.), the domain of all objects is denoted as $\mathcal{O}$ ("*gun_j*","*block_j*", etc.), while the domain for all feature values is denoted as $\mathcal{V}$, and is here the set $\{true, false\}$ of truth values. Finally, assuming *global features*, an *observation* is a whatever tuple $\langle t, f, v\rangle \in \mathcal{T} \times \mathcal{F} \times \mathcal{V}$.

Definition 2.1.1 (history) *A* complete history *is a mapping $H: \mathcal{T} \times \mathcal{F} \to \mathcal{V}$ which for each feature f and time-point t assigns an appropriate value for that feature. A* finite history *is an (complete) history over $\mathcal{T} = [0, t]$, for a chosen t.*

Definition 2.1.2 (fluent) *Given is a history H. If a specific feature f is chosen, then the resulting function of time $H(f): \mathcal{T} \to \mathcal{V}$ is a fluent. A* fluent *is a property, or relation, traced over time.*

Definition 2.1.3 (state) *Given is a history H. If a specific time-point t is chosen, then the resulting function from features to corresponding values $H(t): \mathcal{F} \to \mathcal{V}$ is a* state. *The set of all possible states of a world is denoted by $\mathcal{S}$.*

Definition 2.1.4 (strict inertia) *A state persists unchanged, until an action or event occurs that causes a change, therefore the transition to a new state. Features of objects in the world they does not change their value unless under the direct and known effect of an action.*

According to Newton's dynamics, an *observer* is needed. It can be located in an inertial or non-inertial reference frame, with consequences in its comprehension of phenomena. The former case has been assumed in dealing with *inhabited dynamical systems*, that is systems that contain one or more *agents* which can make observations and influence the system's state at later times by performing actions or activities. Precisely, agents are physical entities consisting of two parts, an *ego* and a *vehicle*. The *vehicle* is a system for perception and actuation, while the *ego* is the level of knowledge and rational decision making of the agent.

To make observations means to perceive the surrounding world via a "perception function", defined as follows.

Definition 2.1.5 (perception) *Let be $\mathcal{S}_M$ a material-level state domain, and $\mathcal{S}$ an image-level state domain. A* perception function *is a mapping $Perc$ from finite histories for $\mathcal{S}_M$ to finite histories for $\mathcal{S}$ which satisfies the following assumptions:* (1) *the argument and value are histories over the same interval;* (2) *if H is a history over $[0,t]$, $s < t$, and $H_{[0,s]}$ is the restriction of H to the interval $[0,s]$, then $Perc(H)_{[0,s]} = Perc(H_{[0,s]})$.*

To perform an action means to make a transformation from state to state. An action is expressed by logical formulae and characterized in terms of the underlying semantics (see section 2.1). It is started by a decision of the ego (that is the ego indicates to the vehicle that it shall start performing the action), and it is performed and terminated by the vehicle in the material system. Actions may have a duration over more than one time point. The duration and the states of the world during the execution of the action can be specified. An *activity* is similar to an action except that it can potentially go on for indefinite time, and its termination as well as its start is decided by the ego. An *event* is a whatever phenomenon that happens in a world. If no agents are involved in the event, then the event is called a *material event* and proceeds in accordance with "natural" laws (ex. the rain falls). We represent with $\mathcal{E}$ the set of all symbols for events.

The following is the *underlying language* for occurrences. An *object expression* is either a member of $\mathcal{O}$, an object constant symbol, or an object variable. An *elementary feature expression* is a feature symbol f_k^0, or $f_k^i(w_1, \ldots, w_i)$ where each argument is an object expression. A *fluent formula* has the form $f \hat{=} \chi$ where f is a feature expression and χ is a subset of its domain. A *time-point expression* is a member of $\mathcal{T}$, a time-point constant symbol, a time-point variable, or an expression formed from time-point expressions using any of the functions in the time structure. An *elementary occurrence expression* is an expression of the form $A(w_1, \ldots, w_k)$ where A is an occurrence symbol, and each w_i is either an object expression or an item. An *occurrence expression* is defined recursively to be either an elementary occurrence expression, or a *sequential expression* of the form "$\epsilon_1; \epsilon_2$", or a *conditional expression* of the form "`if` c `then` ϵ_1 `else` ϵ_2", or a *loop expression* of the form "`do` ϵ `while` c", where ϵ_1, ϵ_2, and ϵ are occurrence expressions, and c is a condition. An *occurrence formula* is a formula of the form $[s,t]A$, where A is an occurrence expression, and s and t are time-point expressions specifying respectively the starting time and the termination time of the occurrence. A *rigid occurrence formula* is an occurrence formula where s and t are time-points and $s < t$. An *occurrence definition* is a mapping from occurrence formulae to their *expansions* which are logical formulae; such mapping is written with $\Rrightarrow$ to signal the translation, for example $[s,t]A \Rrightarrow [t]f$. A *reassignment expression* is an expression of the form $f_i := \chi_i$, where f_i is an elementary feature expression, χ_i is a non empty subset of the domain for f_i, and the operator $:=$ represents value change of a fluent at a time-point or within a temporal interval. A *reassignment formula* is a formula of the form $[s,t]f_1 := \chi_1 \wedge \ldots \wedge f_k := \chi_k$, where s and t are time-point expressions, each $f_i := \chi_i$ is a reassignment expression, and all f_i are different. A reassignment formula is an abbreviation for $([s,t]f_1 := \chi_1) \wedge \ldots \wedge ([s,t]f_k := \chi_k)$. A *conditional reassignment formula* over s and t is a formula of the form $[s]\phi \Rightarrow [s,t]\alpha$, where ϕ is a fluent formula, and $[s,t]\alpha$ is a reassignment formula. Finally, in simple standard cases, *action laws* (occurrence definitions) will map action statements (occurrence formulas) to conjunctions of conditional reassignment formulae with mutually exclusive ϕ. The range of syntax that action statements can expand into, depends on the ontological family is considered. A *schedule SCD* is a possibly parameterized description of a process, expressed as a set of logical formulae. A *process* is intended to be a structure consisting of several occurrences, sequential or concurrent ones.

The ego and the world are then formally defined in terms of "developments".

Definition 2.1.6 (finite development) *A* finite development *(over images in integer time and discrete state variables) is a tuple $\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle$ where: (1) $\mathcal{B} \subseteq \mathcal{T}$ is a finite subset of $\mathcal{T}$, whose members will be called* break-points*. The largest member of $\mathcal{B}$ will be called* now *and will be written as $n_{\mathcal{B}}$ or (when the choice of $\mathcal{B}$ is clear from the context) as n. The subset of $\mathcal{T}$ with all values between 0 and $n_{\mathcal{B}}$ inclusive will be written $\mathcal{T}_{\mathcal{B}}$. (2) M, the* valuation*, is a mapping which assigns values in $\mathcal{T}$ to* some *or* all *of the* temporal constant *symbols $\mathbf{t}_i$, and values in $\mathcal{O}$ to all the* object constant *symbols $\mathbf{o}_i$. The remaining temporal constants are left undefined. (3) H is a* finite history *over $\mathcal{T}_{\mathcal{B}}$. (4) $\mathcal{A}$, the* past action set*, is a set of occurrences $\langle s,t,E \rangle$ where $s,t \in \mathcal{B}$, $s < t$, and $t < n_{\mathcal{B}}$. (5) $\mathcal{C}$, the* current action set*, is a set of tuples $\langle s,E \rangle$ where s is a member of $\mathcal{B}$, and E is a member of $\mathcal{E}$. We represent with $\mathcal{J}$ the domain of all finite developments.*

Definition 2.1.7 (complete development) *A* complete development *(over images) is defined as a similar tuple $\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle$ with the only difference that H is a mapping from $\mathcal{T}$ to $\mathcal{S}$ and that $\mathcal{B}$ may be either a finite or an infinite subset of $\mathcal{T}$. We represent with $\mathcal{J}^\star$ the domain of all complete developments.*

Definition 2.1.8 (infinite development) *An* infinite development *is defined as a complete development over the time domain $\mathcal{T} = [0,\infty)$.*

Definition 2.1.9 (correct revision) *Let J and J' be two finite developments. J' is said to be a* correct revision *of J iff the following conditions hold. It is required that $\mathcal{B} \subseteq \mathcal{B}'$, $M \subseteq M'$, $\mathcal{A} \subseteq \mathcal{A}'$, and that the restriction of H' to the* *time interval* $[0, n_{\mathcal{B}}]$ *shall equal* H. *Every* $b \in \mathcal{B}' - \mathcal{B}$ *must satisfy* $b > n_{\mathcal{B}}$. *For every* $\langle s, t, E \rangle \in (\mathcal{A}' - \mathcal{A})$, *it must be the case that* $t = n_{\mathcal{B}'}$, *and that* $\langle s, E \rangle$ *is a member of* $\mathcal{C}$ *but not of* $\mathcal{C}'$. *For every* $\langle s, E \rangle \in (\mathcal{C} - \mathcal{C}')$, *it must be the case that* $s = n_{\mathcal{B}}$ *or* $\langle s, n_{\mathcal{B}'}, E \rangle \in \mathcal{A}'$. *Also, for every* $\langle s, E \rangle \in (\mathcal{C}' - \mathcal{C})$, *it must be the case that* $s = n_{\mathcal{B}}$.

Definition 2.1.10 (world) *A* world *is defined as a transformation* $W\colon \mathcal{J} \to \mathcal{J} \cup \mathcal{J}^{\star}$, *i.e. a subset of* $\mathcal{J} \times (\mathcal{J} \cup \mathcal{J}^{\star})$ *such that* $\forall J\ \exists J'.W(J, J')$, *and if* $W(J, J')$ *then* $\mathcal{C}' \subseteq \mathcal{C}$, J' *is a correct revision of* J, *and if* n *is the largest member of* $\mathcal{B}$, *either of the following must hold: (a) there is some* $n' > n$ *such that* $\mathcal{B}' = \mathcal{B} \cup \{n'\}$; *(b)* $\mathcal{B}' = \mathcal{B}$ *and the restriction of* H' *to* $[0, n]$ *equals* H.

Definition 2.1.11 (ego) *An* ego *is defined as a mapping* $K\colon \mathcal{J} \to \mathcal{J}$ *where* $K(J)$ *is a correct revision of* J, *and if* $K(J) = J'$, *then* $\mathcal{B} = \mathcal{B}'$, *where* $J = \langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle$ *and* $J' = \langle \mathcal{B}', M', H', \mathcal{A}', \mathcal{C}' \rangle$.

Common-sense reasoning is understood as a game between an ego and a world. Given a world and at least one ego, ego and world they *interact* by playing a board game. The board consists in a finite development, which captures how the game progress over time. The game starts at time zero with the world in an initial state. The ego and the world then take turns, constructing a development which eventually ranges to infinity. During the game the ego and the world, as two players, alternate and modify a development partially.

The world can do the following on its turn: (a) add one member n' to $\mathcal{B}$, where $n' > max\{n \mid n \in \mathcal{B}\}$, and constructs a new finite development as a correct revision up to n'; (b) leave $\mathcal{B}$ unchanged, and extend the history so that it becomes complete.

The ego can do two things on its turn: (a) start an action or activity by adding a tuple $\langle s, E \rangle$ to the current action-set $\mathcal{C}$, where $s = n_{\mathcal{B}}$ and $E \in \mathcal{E}$; (b) end an activity (an action cannot be terminated by the ego), by removing $\langle s, E \rangle$ from $\mathcal{C}$ and adding $\langle s, t, E \rangle$ to $\mathcal{A}$, where $t = n_{\mathcal{B}}$. The ego can not advance time.

If the ego initiates several actions at the same time τ, then the world choses one trajectory non-deterministically.

Definition 2.1.12 (finite game) *Let an ego* K, *a world* W, *an initial state* σ_0, *and an initial valuation* M_0 *be given. A* finite game *for* K, W, σ_0, *and* M_0 *is a finite sequence of developments* $J_0, J_0', J_1, J_1', \ldots, J_k$ *where* $J_k \in \mathcal{J}^{\star}$ *and* $J_0 = \langle \{0\}, M_0, \{0 \mapsto \sigma_0\}, \varnothing, \varnothing \rangle$, $J_i' = K(J_i)$ *for* $0 \le i < k$, *and* $W(J_i', J_{i+1})$ *for* $0 \le i < k$. *If* $J_k = \langle \mathcal{B}_k, M_k, H_k, \mathcal{A}_k, \mathcal{C}_k \rangle$ *and* $M_k \subseteq M$ *where* M *is a complete valuation, then* $\langle \mathcal{B}_k, M, H_k, \mathcal{A}_k, \mathcal{C}_k \rangle$ *is a* resulting development *from the finite game.*

Definition 2.1.13 (infinite game) *An* infinite game *for* K, W, σ_0 *and* M_0 *(chosen as in the definition of finite game) is an infinite sequence of developments* $J_0, J_0', J_1, J_1', \ldots$ *where each* J_i *and* J_i' *is a finite development, and* $J_0 = \langle \{0\}, M_0, \{0 \mapsto \sigma_0\}, \varnothing, \varnothing \rangle$, $J_i' = K(J_i)$ *for* $i \ge 0$, *and* $W(J_i', J_{i+1})$ *for*

$i \geq 0$. *If* $J_i = \langle \mathcal{B}_i, M_i, H_i, \mathcal{A}_i, \mathcal{C}_i \rangle$ *for every* $i \in \mathbb{N}$, *then a* resulting development *from the game is any development of the form*

$$\langle \lim_i \mathcal{B}_i, M, \lim_i H_i, \lim_i \mathcal{A}_i, \lim_i \mathcal{C}_i \rangle$$

where for the purpose of taking the limit each H_i *is considered as a set of maplets of the form* $(t \mapsto r)$, M *is a complete valuation, and* $\lim_i M_i \subseteq M$.

Definition 2.1.14 (scenario) *Let* W *be an IDS world. A* scenario *for* W *is a set of games where* W *was the world player.*

2.1.1 The underlying semantics

Definition 2.1.15 (influence) *Given an action* E *and a state* σ, $Infl(E, \sigma)$ *represents the set of those features which may be affected if the action* E *is initiated in the state* σ. *The influence is a function* $Infl: \mathcal{E} \times \mathcal{S} \rightarrow 2^{\mathcal{F}}$.

It is clear that $\mathcal{F} \supset Infl(E, \sigma)$, varying E in $\mathcal{E}$ and σ in $\mathcal{S}$.

Definition 2.1.16 (occlusion) *Let* f *be a feature symbol in* $\mathcal{F}$. *For all* $[s,t] \subset \mathcal{T}$ *and* $A \in \mathcal{E}$ *such that* $[s,t]A$ *holds, if* $f \in Infl(A, H(s))$ *then* f *is* occluded *in* $(s,t]$.

Underlying semantics that adopt the occlusion concept are called "encapsulated semantics", since feature values, during time intervals in which influencing actions are performed, are *prevented from been seen*. The reassignment expression $[t]f := \chi$ can be expanded into $[t](f \hat{=} \chi \wedge \varkappa f)$, and $[s,t]f := \chi$ into $(s,t]\varkappa f \wedge [t]f \hat{=} \chi$, where $\varkappa f$ means "occluded f".

Definition 2.1.17 (trajectory set) *Given an action* E *and a state* σ, *the set* $Trajs(E, \sigma)$ *represents the set of possible trajectories of* E *initiated in* σ, *and it is expressed as a set of finite nonempty sequences of partial states each of which assigns values to those features appearing in* $Infl(E, \sigma)$, *but not to other features.* $Trajs$ *is a function from* $\mathcal{E} \times \mathcal{S}$ *to set of trajectories over* $Infl(E, \sigma)$.

Underlying semantics that adopt the notion of trajectory are called "trajectory semantics". Unlike the encapsulated semantics, trajectory semantics they allow features to be observed, even if influencing actions are performed.

Definition 2.1.18 *Let* W *be an IDS world. The* set theoretic description *of the world* W *is the corresponding tuple* $\langle Infl, Trajs \rangle$.

To the trajectory level description of a world as a pair $\langle Infl, Trajs \rangle$, is associated a corresponding logical formula called the *Full Trajectory Normal Form* (FTNF) for the action laws. The description of the world $\langle Infl, Trajs \rangle$ consists in a propositional table that specifies the effects of each action A by indicating, for each choice of state σ, the value of $Infl(A, \sigma)$ and the single member of $Trajs(A, \sigma)$. *The FTNF for an action law is the direct translation of the information in the table.* The choice of the ontological family for the world will impose syntactical as well as semantical constraints for such laws. We represent with LAW a set of action laws in FTNF.

In order to define the concept of trajectory-semantics world, the following auxiliary notation has been adopted for extending a finite history with a trajectory. Let H be a finite history over $[0, t]$, and let $h = \langle \sigma'_1, \sigma'_2, \ldots, \sigma'_k \rangle$ be a trajectory. Then $H \rhd h$ is the finite history H' over $[0, t+k]$ which is defined by $H'(s) = H(s)$ if $s \leq t$ and $H'(s) = H(t) \oplus \sigma'_i$ if $s = t + i > t$. The operator $\oplus : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ is such that $\sigma \oplus \sigma'$ is the state $\sigma^\star$ where $\sigma^\star(f) = \sigma'(f)$ when $\sigma'(f)$ is defined, and $\sigma^\star(f) = \sigma(f)$ otherwise. In particular $H \rhd \langle \varnothing \rangle$ extends H by one time-step from t to $t+1$ such that $H(t+1) = H(t)$.

Definition 2.1.19 (trajectory-semantics world) *A* trajectory-semantics world *is an IDS-world which corresponds to some world description. An IDS-world W is the* corresponding world *of a world description $\langle Infl, Trajs \rangle$ if and only if it satisfies the following conditions. If $J = \langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle$, $0 \in \mathcal{B}$, $n_{\mathcal{B}} = max(\mathcal{B})$, and H is defined over $[0, n_{\mathcal{B}}]$, then $W(J, J')$ iff $J' = \langle \mathcal{B} \cup \{n'\}, M, H', \mathcal{A}', \varnothing \rangle$ where either of the following holds: (a) for some action designator E and for some trajectory $h = \langle \sigma'_1, \sigma'_2, \ldots, \sigma'_k \rangle$, $h \in Trajs(E, H(n_{\mathcal{B}}))$, it holds that $\langle n_{\mathcal{B}}, E \rangle \in \mathcal{C}$, $n' = n_{\mathcal{B}} + k$, $H' = H \rhd h$, and $\mathcal{A}' = \mathcal{A} \cup \{\langle n_{\mathcal{B}}, n', E \rangle\}$; (b) $\mathcal{C} = \varnothing$, $n' = n_{\mathcal{B}} + 1$, $H' = H \rhd \langle \varnothing \rangle$, and $\mathcal{A}' = \mathcal{A}$.*

Definition 2.1.20 (trajectory-semantics ego) *A* trajectory-semantics ego *is an IDS-ego K such that if $K(\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle) = \langle \mathcal{B}', M', H', \mathcal{A}', \mathcal{C}' \rangle$ hence $\mathcal{A} = \mathcal{A}'$, $\mathcal{C} \subseteq \mathcal{C}'$, and $\mathcal{C}' - \mathcal{C}$ has at most one member.*

A set of games between a trajectory-semantics world and a trajectory-semantics ego may be obtained using a *scenario description.* Such description (also called *chronicle*) consists in a description of the world itself and of how its state has changed over time. A scenario description then defines unambiguously an IDS-world, in providing the ego's available information. The formal definition of scenario description, and the *taxonomy of scenario descriptions* itself, is obtained by making explicit *ontological assumptions* about the actions and the world where the actions occur, and *epistemological assumptions* about the reasoner's knowledge, the actions and the world itself.

- The $\mathcal{N}$ epistemological assumption represents the absence of any epistemological assumptions and, in particular, represents no assumption of inertia.

- The $\mathcal{K}$ epistemological assumption allows to describe worlds with accurate and complete information about actions. Precisely: (1) *elementary actions* are allowed, i. e. a schedule specifies each action in terms of action symbol, arguments, starting time and ending time; (2) there are no other actions besides those mentioned in the schedule; (3) action laws must explicitly specify all the features they may influence (an empty set of postconditions is not allowed), together with a possibly empty set of preconditions for their applicability; (4) for every combination of action designator and starting state, there is a definition for how to perform the action from that state (actions are always feasible), that is $Trajs(E, \sigma) \neq \varnothing$ for every action E and state σ. An action is successfully executed if its preconditions are satisfied. A non successfully executed action it has no effect (because of the inertia assumption), that is $Infl(E, \sigma) = \varnothing$ and $Trajs(E, \sigma) = \{\langle \varnothing \rangle\}$.

Moreover, epistemological sub-characteristics may be specified, and combined, in order to restrict the extent of the class. For example **s** restricts to the case of complete knowledge about the initial state, **p** further restricts **s** to the case of no information at any state later than the initial one (pure prediction problem), while the absence of **g** relaxes $\mathcal{K}$**g** to the case of "non-ground" scenarios i.e. those scenarios where the interpretation of temporal and object constants may not be initially given and the mapping M is then needed.

Other epistemological assumptions may be addressed, such as the $\mathcal{Q}$, $\mathcal{P}$, $\mathcal{U}$, $\mathcal{M}$ and $\mathcal{I}$ assumptions (see [San94a]).

- The **IAD** ontological family is defined as the set of all trajectory-semantics worlds where: (1) strict inertia in integer time applies, (2) alternative results of actions are allowed, (3) structural ramification is allowed, and (4) $Trajs(E, \sigma) \neq \emptyset$ for every action E and state σ.

- Let σ be a state, let $\mathcal{O}$ be a set of objects, let $\mathcal{F}/\mathcal{O} \subseteq \mathcal{F}$ be the set of those features where all arguments are members of $\mathcal{O}$, then let $\sigma/\mathcal{O} \subseteq \sigma$ be the partial state which is obtained from σ by restricting its arguments to $\mathcal{F}/\mathcal{O}$. Let $args(E)$ be the set of objects which occur as arguments in E. A world described by the pair $\langle Infl, Trajs \rangle$ is said to have *strict locality* iff for all $E \in \mathcal{E}$ and $\sigma' \in \mathcal{S}$, holds $\sigma/args(E) = \sigma'/args(E) \Rightarrow Infl(E, \sigma) = Infl(E, \sigma') \subseteq \mathcal{F}/args(E) \wedge Trajs(E, \sigma) = Trajs(E, \sigma')$. A world W is in **IA** iff it is in **IAD** and has strict locality.

- A world W is in **I** iff W is in **IA** and in any world description $\langle Infl, Trajs \rangle$ of W all members of $Trajs(E, \sigma)$ have the same last element, $Trajs(E, \sigma) = Trajs(E, \sigma')$, and $Infl(E, \sigma) = Infl(E, \sigma')$ for arbitrary E, σ and σ'.

  I means *strict inertia in integer time.* Ontological sub-characteristics may be specified, and combined, in order to further restrict the extent of the class. For example **s** restricts **I** to the case of single-step actions, **e** restricts to the case of encapsulated actions, **f** further restricts actions in **Ie** to have a specific duration, **b** restricts to binary features, and **u** imposes the actions to effect just one feature.

- **A** means *alternative results of actions.* **Ad** is a restriction on **A** for which the final effect of an action is completely determined by the state of the world at the time when the action started (deterministic change).

Above epistemological and ontological designators are just examples of possible codes for assumptions; the formal definition of their corresponding assumptions, as shown, is made in terms of the underlying semantics (influence, occlusion or trajectory).

The epistemological and ontological assumptions are used as the basis for the classification in the taxonomy. By combining their designators (like $\mathcal{K}sp$-**IA**d) it is possible to identify precisely a *class of reasoning problems* to be reasoned about. Moreover, such combinations are used to show the range of applicability of different logics. Classes of scenarios are characterized by this coding scheme, that makes explicit the epistemological and ontological assumptions of the class.

Definition 2.1.21 (scenario description) *A* scenario description *is defined as a tuple* $\langle \mathcal{Z}, \Upsilon_1, \Upsilon_2, \ldots, \Upsilon_k \rangle$ *where* $\mathcal{Z}$ *is an epistemological characteristics that identifies the epistemological assumptions made about the scenario. The elements* Υ_i *are sets of formulae or other formal objects; the number and type are specific for each choice of* $\mathcal{Z}$.

Definition 2.1.22 (scenario description in $\mathcal{N}$) *A* scenario description *in* $\mathcal{N}$ *is a tuple* $\langle \mathcal{N}, \mathcal{O}, OBS \rangle$ *where* $\mathcal{O}$ *is an object domain, and* OBS *is a set of observations.*

Definition 2.1.23 (scenario description in $\mathcal{K}$) *A* scenario description *in* $\mathcal{K}$ *is a tuple* $\langle \mathcal{K}, \mathcal{O}, Diw, SCD, OBS \rangle$ *where* $\mathcal{O}$ *is an object domain,* Diw *is a description* $\langle \psi, LAW \rangle$ *of an IDS world where* ψ *is an ontological designator and* LAW *is a set of action laws,* SCD *is a schedule specifying a number of action occurrences and their temporal order, and* OBS *is a set of observations of the values of features at some point in time.*

A *chronicle* is a scenario description where the epistemological designator is one of $\mathcal{K}$, $\mathcal{Q}$ or $\mathcal{P}$, or another epistemological designator. A chronicle is not defined as a single set of logic formulae, but as a tuple whose structure depends on the ontological family, and whose elements are sets of formulae or other syntactic entities.

The desired task so far was to conclude as much as possible about the values of features at different points in time. Such set of conclusions are referred to as *chronicle completion.* Of course, our interest is in intended completions, i.e. those completions containing all and only desirable conclusions constrained by a chosen assumption of inertia. Chronicle completion for a scenario description Υ is then the operation of determining the *intended model set* for Υ.

Definition 2.1.24 (complete development set) *For a given scenario description* Υ *and time domain* $\mathcal{T}$, $Mod(\Upsilon)$ *denotes the* set of complete developments *which are correctly described by* Υ.

Definition 2.1.25 (intended model set) *For a given scenario description* Υ *belonging a class* C*, an* intended model *is an intended interpretation* $\langle M, H \rangle$ *such that the development* $\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle \in Mod(\Upsilon)$. *The* set of intended models *for the given scenario description is defined as being the set* $\Sigma(\Upsilon)$ *of those interpretations* $\langle M, H \rangle$ *obtained by extracting the* M *and the* H *components from developments in* $Mod(\Upsilon)$.

$$\Sigma_C(\Upsilon) = \{\langle M, H \rangle \mid \langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle \in Mod(\Upsilon)\}$$

Sometimes, misusing the notation, $Mod(\Upsilon)$ stands for $\Sigma_C(\Upsilon)$.

Definition 2.1.26 (assessment) *By an* assessment *of a non-monotonic logic is meant a statement that for a certain class of reasoning problems within the taxonomy, the logic is guaranteed to always give the correct set of conclusions: the set of intended conclusions is equal to the set of conclusions obtained through the logic.*

Definition 2.1.27 (range of applicability) *The* range of applicability *for a given logic identifies for which family of scenario descriptions in the taxonomy does the logic obtain the correct conclusions.*

2.1.2 The $\mathcal{K}$-IA class

The broadest class in [San94a] is the $\mathcal{K}$-**IA** class of reasoning problems.

Definition 2.1.28 ($\mathcal{K}$-IA chronicle) *A $\mathcal{K}$-**IA** chronicle is a five-tuple*

$$\langle \mathcal{K}, \mathcal{O}, \langle \mathbf{IA}, LAW \rangle, SCD, OBS \rangle$$

*where $\mathcal{K}$ is the assumed epistemology, $\mathcal{O}$ is a finite set of objects, SCD is a set of schedule statements, and OBS is a set of observation statements. The tuple $\langle \mathbf{IA}, LAW \rangle$, where **IA** is the assumed ontology and LAW is a set of action laws in Full Trajectory Normal Form, is the corresponding logical description of the world description $\langle Infl, Trajs \rangle$, for some $\mathcal{K}$-**IA** world.*

Definition 2.1.29 (complete development set for $\mathcal{K}$-IA chronicles) *Let be $\mathcal{T}$ the half line $[0, \infty)$. If a chronicle $\Upsilon = \langle \mathcal{K}, \mathcal{O}, \langle \mathbf{IA}, LAW \rangle, SCD, OBS \rangle$ in $\mathcal{K}$-**IA** is given, then the set of complete developments over $\mathcal{T}$ and $\mathcal{O}$ is obtained as follows: (1) construct the unique trajectory-semantics world W that is precisely specified by $\langle \mathbf{IA}, LAW \rangle$, then select an arbitrary trajectory-semantics ego and an arbitrary initial state; (2) generate all possible developments which can be obtained in games between them; (3) add an arbitrary M component to each development; (4) restrict the set of developments to those where all formulas in $SCD \cup OBS$ are satisfied, and where there is a one-to-one correspondence between actions in the development and action statements in SCD. This subset is $Mod(\Upsilon)$.* $\square$

An action statement $\langle s, t, E \rangle$ is said to be satisfied by an interpretation $\langle M, H \rangle$ if and only if there is some game, resulting in H, in which the action E was invoked by an ego at time $M[s]$, and the world then selected a trajectory of length $M[t] - M[s]$. A set of action statements is said to be satisfied by an interpretation $\langle M, H \rangle$ if and only if there is some game, resulting in H, such that all the given action statements and no others are satisfied for $\langle M, H \rangle$ in the game.

Definition 2.1.30 (intended model set for $\mathcal{K}$-IA chronicles) *Let be Υ a chronicle in $\mathcal{K}$-**IA** . The* set of intended models *for the given chronicle is defined as*

$$\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}(\Upsilon) = \{\langle M, H \rangle \mid \langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C} \rangle \in Mod(\Upsilon)\}$$

Definition 2.1.31 (classical model set for $\mathcal{K}$-IA chronicles) *The* classical model set $[\![\alpha]\!]$ *for a given first-order formula α is, as usual, the set of all interpretations satisfying α, that is the set of Tarskian models for α as provided by classical first-order logic. The* classical model set *for a chronicle Υ in $\mathcal{K}$-**IA** , written $[\![\Upsilon]\!]$, is defined as $[\![LAW[SCD] \cup OBS]\!]$, where $LAW[SCD]$ is intended to be the expansion of all formulae in SCD by using their corresponding definitions in LAW.*

Proposition 2.1.1 *For all chronicles Υ in $\mathcal{K}$-**IA** , $\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}(\Upsilon) \subseteq [\![\Upsilon]\!]$.*
Proof. The set $[\![\Upsilon]\!]$ allows models which do not satisfy inertia and models where there are additional actions besides those specified in SCD.

A final note. A scenario description is called *ground* (properly, it belongs to a class with the **g** epistemological sub-characteristics) if and only if observations and schedule's elements are completely instantiated, that is all temporal and object constant symbols are respectively numbers in $\mathcal{T}$, and objects in $\mathcal{O}$. A scenario description is *non ground* iff at least one of its temporal or object constant symbols is non-ground, i.e. it is not interpreted. This interpretation is demanded to the mapping M. Non-groundness may result in scenarios where the schedule does not completely specify the temporal order of the actions, then different action orders may result in entirely different histories. In dealing with ground scenario descriptions, the M component cease its relevance. We recall that a valuation is required to assign values to all temporal constant symbol $\mathbf{t}_i$ and object constant symbols $\mathbf{o}_i$. Thus, such valuation M will be irrelevant in case of ground scenarios. If $\langle M, H\rangle$ is an intended model for a ground scenario in $\mathcal{K}$-**IA** , then any $\langle M', H\rangle$ for arbitrary M' is an intended model as well.

2.1.3 Semantic methods for model reduction

Since the intended model set for a chronicle does not generally agree with its classical model set (proposition 2.1.1), preferential entailment methods [Sho89, San94a, SS94a] are the proper machinery to reconcile them. The intended model set can be expressed in terms of the classical model set, by neglecting less desirable conclusions via the application of a *preference relation* $\ll$ on classical models. A *selection function* ζ, defined directly in terms of a preference relation $\ll$, will map any set $[\![\Upsilon]\!]$ of classical models for the given premises to a subset $\zeta([\![\Upsilon]\!])$ of selected models, so that the set of selected models is $\zeta([\![\Upsilon]\!]) = Min(\ll, [\![\Upsilon]\!])$, where

$$Min(\ll, [\![\Upsilon]\!]) = \{I \in [\![\Upsilon]\!] \mid \neg\exists\, I' \,.\, I' \in [\![\Upsilon]\!] \wedge I' \ll I\}$$

We remind that $[\![\Upsilon]\!] = [\![LAW[SCD] \cup OBS]\!]$. The selection function obtains those members of $[\![\Upsilon]\!]$ which are minimal according to $\ll$, so the maximally preferred models are the selected ones. A selection function ζ is *correct* (def. 2.1.26) for a chronicle Υ if and only if $\zeta([\![\Upsilon]\!]) = \Sigma\,(\Upsilon)$. The preferential entailments (non-monotonic logics) that are to be assessed and compared, are all characterized in terms of a preference relation on models: for all interpretation I, $I \models_\zeta \gamma \quad \Leftrightarrow \quad \zeta(I) \models \gamma$. A number of notable preference relations and related entailment methods has been defined with that technique. The Prototypical Global Minimization (McCarthy, 1984) has been assessed for chronicles in $\mathcal{K}r$-**Is**$\mathbf{A}z$, the Original Chronological Minimization (Kautz, 1986) has been assessed for chronicles in $\mathcal{K}sp$-**Is**$\mathbf{A}d$, and the Prototypical Chronological Minimization has been assessed for chronicles in $\mathcal{K}p$-**IA**ex . Then the following negative result:

Proposition 2.1.2 *There does not exist any preference order $\ll$ such that*

$$Min(\ll, [\![\Upsilon]\!]) = \Sigma_{\mathcal{K}\text{-}\mathbf{IA}}\,(\Upsilon)$$

Proof. Consider a world where there is a single propositional feature p and a single action A whose effect is like tossing a coin, i.e. the value of p after the action is non-deterministically true or false, regardless of its value when the action starts. Let Υ be the chronicle where both SCD and OBS are empty, and let Υ' be the chronicle where OBS is empty and SCD consists of the single formula $[4,5]A$. Then $[\![\Upsilon]\!] = [\![\Upsilon']\!]$ but $\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}\,(\Upsilon) \neq \Sigma_{\mathcal{K}\text{-}\mathbf{IA}}\,(\Upsilon')$.

The conceptual basis of the above entailment methods is the application of the preference relation over the whole set of premises. The application of different selection functions to the classical model sets of different set of premises is allowed by means of the *filtering* technique [San89b]. The set of premises Υ is separated in the sets OBS and SCD. The set of selected models is chosen by (filter) preferential entailment as

$$Min(\ll, [\![LAW[SCD]]\!]) \cap [\![OBS]\!]$$

so that the minimization is imposed "before" the observations.

Proposition 2.1.3

$$Min(\ll, [\![LAW[SCD]]\!]) \cap [\![OBS]\!] \subseteq Min(\ll, [\![LAW[SCD] \cup OBS]\!])$$

The Prototypical Chronological Minimization with Filtering has been assessed for chronicles in $\mathcal{K}$-$\mathbf{IA}ex$, the Global Minimization of Occlusion with No-change Premises has been assessed for chronicles in $\mathcal{K}r$-$\mathbf{I}s\mathbf{A}$, the Chronological Minimization of Occlusion and Change has been assessed for chronicles in $\mathcal{K}$-$\mathbf{IA}e$, the Chronological Assignment and Minimization of Occlusion and Change has been assessed for the whole $\mathcal{K}$-$\mathbf{IA}$ class, and the Pointwise Minimization of Occlusion with No-change Premises has been assessed for the whole $\mathcal{K}$-$\mathbf{IA}$ class as well.

2.2 Shoham's preferential logics

The general notion of *preferential logic* was originally proposed by J. Shoham in [Sho89]. *Preferential logics are the result of associating with a standard logic (i.e. any logic with the usual compositional, model-theoretic semantics) a preference relation on models.*

Definition 2.2.1 (preferential logic) *Let $\mathcal{L}$ be a standard logic and A a sentence in it, and let $\sqsubset$ be a strict partial order on interpretations for $\mathcal{L}$. Intuitively $M_1 \sqsubset M_2$ will mean that the interpretation M_2 is* preferred *over the interpretation M_1. $\mathcal{L}$ and $\sqsubset$ define a new logic $\mathcal{L}_\sqsubset$. We will call such logics* preference logics.

The syntax of $\mathcal{L}_\sqsubset$ is identical to that of $\mathcal{L}$. The semantics of $\mathcal{L}_\sqsubset$ are as follows.

Misusing the terminology, we will call anything which goes to the left of the $\models$ an *interpretation.*

Definition 2.2.2 *An interpretation M* preferentially satisfies *A (written $M \models_\sqsubset A$) if $M \models A$, and if there is no other interpretation M' such that $M \sqsubset M'$ and $M' \models A$. In this case we say that M is a* preferred model *of A.*

If $M \models_\sqsubset A$ then also $M \models A$.

Definition 2.2.3 (preferential satisfiability) *A is* preferentially satisfiable *if there exists an M such that $M \models_\sqsubset A$.*

All preferential satisfiable formulas are also satisfiable.

The converse is not necessarily true.

Definition 2.2.4 (preferential validity) *A is* preferentially valid *(written $\models_{\sqsubset} A$) if and only if $\neg A$ is not preferentially satisfiable.*

Lemma 2.2.1 *A is preferentially valid if for any M, either $M \models A$, or there is an M' such that $M \sqsubset M'$ and $M' \not\models A$.*

All valid formulas are also preferentially valid. The converse is not true.

Definition 2.2.5 *A partial order $\sqsubset$ on models is said to be* bounded *if there does not exist an infinite sequence of models $M_1, M_2, M_3, \ldots$ such that $M_1 \sqsubset M_2 \sqsubset M_3 \sqsubset \ldots$.*

Theorem 2.2.1 *Let $\mathcal{L}$ be a preferential logic, $\sqsubset$ a bounded partial order on its models, and A a formula in $\mathcal{L}$ (and therefore also in $\mathcal{L}_{\sqsubset}$). Then A is satisfiable in $\mathcal{L}$ if and only if A is preferentially satisfiable in $\mathcal{L}_{\sqsubset}$.*

Proof. (see [Sho89])

Corollary 2.2.1 *Let $\mathcal{L}$ be a preferential logic, $\sqsubset$ a bounded partial order on its models, and A a formula in $\mathcal{L}$ (and therefore also in $\mathcal{L}_{\sqsubset}$). Then A is valid in $\mathcal{L}$ if and only if A is preferentially valid in $\mathcal{L}_{\sqsubset}$.*

$\mathcal{L}$ can be viewed as the special case $\mathcal{L}_{\phi}$, where ϕ is the empty relation (that is a bounded relation).

Definition 2.2.6 (preferential entailment) *A* preferentially entails *B (written $A \models_{\sqsubset} B$) if for any M, if $M \models_{\sqsubset} A$ then $M \models B$, or, equivalently, if the models (preferred and otherwise) of B are a superset of the preferred models of A.*

Definition 2.2.7 (monotonic preferential logic) *A preferential logic $\mathcal{L}_{\sqsubset}$ is* monotonic *if for all $A, B, C \in \mathcal{L}$, if $A \models_{\sqsubset} C$ then also $A \wedge B \models_{\sqsubset} C$.*

The above definition is equivalent to saying that a logic is monotonic if all preferred models of $A \wedge B$ are also preferred models of A. $\mathcal{L}_{\phi}$ is monotonic.

Lemma 2.2.2 *Let $\mathcal{L}_{\sqsubset}$ be a preferential logic, and A a sentence in it. Then A is preferentially satisfiable if and only if there does not exists a sentence B in $\mathcal{L}_{\sqsubset}$ such that both $A \models_{\sqsubset} B$ and $A \models_{\sqsubset} \neg B$.*

Lemma 2.2.3 *Let $\mathcal{L}_{\sqsubset}$ be a preferential logic, A and B two sentences in it, and M a model. Then, if $M \models_{\sqsubset} B$ and $M \models_{\sqsubset} A$, then $A \models_{\sqsubset} A \wedge B$.*

Theorem 2.2.2 (weak deduction) *Let $\mathcal{L}$ be a preferential logic, and A, B and C three sentences in it. Then, if $A \wedge B \models_{\sqsubset} C$, then also $A \models_{\sqsubset} B \supset C$.*

Proof. .

1. $A \wedge B \models_{\sqsubset} C$ (assumption).
2. $A \not\models_{\sqsubset} B \rightarrow C$ (assumption).
3. There exists an M_0 such that $M_0 \models_{\sqsubset} A$, but $M_0 \not\models B \rightarrow C$ (from 2).

4. $M_0 \models_{\sqsubset} A$, $M_0 \models B$, and $M_0 \not\models C$ (from 3).
5. $M_0 \not\models_{\sqsubset} A \wedge B$ (from 1).
6. If $M_0 \models_{\sqsubset} A$ and $M_0 \models B$ then $M_0 \models_{\sqsubset} A \wedge B$ (Lemma 2.2.3).
7. Contradiction (from 4,5 and 6).

Theorem 2.2.3 *Let $\mathcal{L}$ be a preferential logic. Then the following two statements are equivalent:*

1. *for all $A, B, C \in \mathcal{L}_{\sqsubset}$, if $A \models_{\sqsubset} B \supset C$ then also $A \wedge B \models_{\sqsubset} C$.*
2. *$\mathcal{L}_{\sqsubset}$ is monotonic.*

PROOF. The $(2 \Rightarrow 1)$ direction is trivial. For the $(1 \Rightarrow 2)$ direction, assume that $A \models_{\sqsubset} C$. Therefore by definition $A \models_{\sqsubset} B \rightarrow C$ for any B. Then by the hypothesis $A \wedge B \models_{\sqsubset} C$.

Approaches to non-monotonic reasoning that make use of a whatever notion of minimum model, like circumscription, are subsumed by the notion of *preferential entailment*.

Chapter 3

Underlying Semantics

In analyzing the reality using models, even intended ones, we should never forget the existing gap among the reality and models: the solution of a problem always is, as a matter of fact, the solution of the representation we built of the real problem. The model will always be a very limited description of the reality, however it must represent with reasonable accuracy those aspects of interest for the solution of the decisional problem we are dealing with. Naturally a *necessarily discrete* description of the reality imposes some restrictions in this sense. We wish to extend the image-level description of the world to the case of continuous or discrete time, continuous or discrete-valued state variables, and continuous or discrete change, depending on the need. In pursuing this aim, in this chapter we define the knowledge representation method and the intended model set for each reasoning problem where accurate and complete information about actions, together with strict inertia in continuous (or discrete) time, continuous (or discrete) change and alternative results of possibly concurrent and independent actions are the assumed properties. The case is designated, in the taxonomy of Sandewall, with the $\mathcal{K}$-**RAC***i* class of reasoning problems.

3.1 Some philosophical problems

The western philosophy dates back to Aristotle the notion of time as measurable order of the movement. According to Aristotle (Aristotle - *Fisica*, IV, 11, 219 b 1), time is not change in itself, but *the measure of change according to a before and an after*. Since time evolves because of change, the absence of change determines the end of the world. Aristotle assumed the world eternal. The emphasis on the irreversibility of change, due to the second principle of thermodynamics, was later made by Reichenbach [Rei56], for which time has a growth direction. Sandewall's notion of time for $\mathcal{K}$-**IA** worlds fully respects both the above requirements, in fact: (1) when the current action set is empty, that is when no action is initiated by the ego, then the world initiates, executes and terminates a null action which trajectory is the null trajectory $\langle \varnothing \rangle$ of length 1, so that the current *now* is increased by one time-point; (2) when the current action set is not empty, that is when an action is initiated by the ego, then the world increases the current *now* by k time-points, where k is the length of the invoked action. Executed actions are the image level justification of flowing time, and determine, de facto, the way the world beats time. Concerning the direction of time, actions may also have reversible effects on feature values, in fact the only one operation the world applies on the *now* is the sum operation.

In pursuing the aim of extending Sandewall's notion of time to the continuum case, we meet two classical problems. The first problem is that no action can be physically initiated and terminated such that its duration is infinitesimal and its repeated execution is the image level justification to continuous time. The world humans perceive is, in fact, fully discrete, and time and continuity are abstractions of the human mind. The problem with beating the continuous time *formally* consists in the well known non-existence of the successor function for real numbers, in fact the image level world can just advance time by increasing the current *now* by an infinitesimal quantity ε, and lose an infinite amount of time-points between the *now* $n_{\mathcal{B}}$ and $n' = n_{\mathcal{B}} + \varepsilon$. We quickly solve as follows. Per definition, the material level consists in the description of the physical system, or can be even identified with it. In the former case, since the material level is the description of an inertial system, the notion of time at the image level can be reasonably identified with the notion of time in classical physics, which is a well known mathematical structure of linear time. In the latter case the material level has no time, since no real world object can be identified with it, and no continuity, since no physical experience can be made in this sense; then the notion of time at the image level will be a convenient mathematical structure, which will not necessarily be the one adopted in classical physics. We also show that Sandewall's definitions for $\mathcal{K}$-**IA** can be re-obtained as a particular instance of the former case.

Tightly connected with the absence of a successor function for real numbers is the famous second evidence against motion of Zenone from Elea (Italy, 489 B.C.), which occur as follows: the trajectory of a running action can never describe the continuous change during the scheduled time interval. We solve this problem, within the Features and Fluents approach, by defining the concept of "partial fluent". Again, Sandewall's definition of trajectory semantics world for $\mathcal{K}$-**IA** will be shown to be a particular instance of the proposed solution.

3.2 Continuous time

From a human perspective, since humans may generally perceive the world in different manners, the time itself is susceptible of different understandings. The time in fact *flows in a natural fashion* according to our peculiar common sense, and logical approaches to temporal reasoning are plenty of (at all natural) time structures. Let then assume the existence of a certain machinery capable to describe the flux of time, let say a "master clock", and get rid of that by giving freedom for its definition. Formally, the master clock is a strictly monotonic rising function $clock \colon \to \mathcal{T}$, without arguments, with internal state. The internal state, that depends on the choice of the temporal domain $\mathcal{T}$, determines the *clock* function non-ambiguously, since the *clock* function just reports the answers it generates. An answer of *clock* is called a "now", written ν (greek nu). We simply assume per default a strictly increasing evolution of the *now*, and rely to the master clock the task to beat time.

With respect to the image level description of the world, the function symbol *clock* may be associated to an *internal* or an *external* function. An internal clock function is, per definition, a computable function; the world knows this definition, hence it knows exactly how to make the computation. An external clock function has no internal formal definition, it may also be a non computable function, it is simply invoked by the world and its actual value is obtained for magic. An example of internal clock may be the successor function of Peano's arithmetics, recursively defined as $clock = 0$, $clock = clock + 1$, where 0 is the origo of time-points and 1 is the length of the null trajectory. An external clock function may be, for example, the function which reads the time-signal of an atomic clock; in that case the resulting underlying semantics is *hybrid*, since at least this extra logical component is involved.

We now fix an order relation and operations on members of the time-point domain. The *basic time structure* is $\langle \mathcal{T}; \{+^2, clock^0\}, \{=^2, \sqsubseteq^2\}\rangle$, defined with the obvious first order language $\mathcal{L}_\Sigma$ for time-points, built on top of the signature $\Sigma = \{\{+^2, clock^0\}, \{=^2, \sqsubseteq^2\}\}$.

- (Axioms for the sum function) for all pair of time-points s and t in $\mathcal{T}$ is defined a third time-point in $\mathcal{T}$, written $s + t$, in such a way

S1 :	$\forall\, r, s, t \in \mathcal{T}\ .\ (r + s) + t = r + (s + t)$	associativity
S2 :	$\forall\, s, t \in \mathcal{T}\ .\ s + t = t + s$	commutativity
S3 :	$\forall\, t \in \mathcal{T} . \exists\, t_0 \in \mathcal{T}\ :\ t + t_0 = t$	neuter element
S4' :	$\forall\, s, t \in \mathcal{T} : s \sqsubseteq t . \exists!\, k \in \mathcal{T}\ :\ t + k = s$	

- (Axiom for the *clock* function)

  C1 : For all pairs of subsequent calls of *clock*, ν and ν', is $\nu \sqsubset \nu'$.

- (Axioms for the order predicate) For all pair of time-points s and t is it possible to say whether s precedes t ($s \sqsubseteq t$) or t precedes s ($t \sqsubseteq s$) in such a way

O1 :	$\forall\, t \in \mathcal{T}\ .\ t \sqsubseteq t$	reflexivity
O2 :	$\forall\, s, t \in \mathcal{T}\ .\ s \sqsubseteq t \wedge t \sqsubseteq s \Rightarrow s = t$	anti-symmetry
O3 :	$\forall\, r, s, t \in \mathcal{T}\ .\ r \sqsubseteq s \wedge s \sqsubseteq t \Rightarrow r \sqsubseteq t$	transitivity
O4 :	$\forall\, s, t \in \mathcal{T}\ .\ s \sqsubseteq t \vee t \sqsubseteq s$	totality (optional)
O5 :	$\forall\, t \in \mathcal{T}\ .\ t_0 \sqsubseteq t$	
O6 :	$\forall\, r, s, t \in \mathcal{T}\ .\ r \sqsubseteq s \rightarrow r + t \sqsubseteq s + t$	

The equality is obviously symmetric, that is $\forall\, s, t \in \mathcal{T}\ .\ (s = t) \rightarrow (t = s)$. The time-point t_0 is the *origo* of time-points, which does not necessarily correspond to the number 0. Because of the axiom *O5*, if $t_0 = 0$ we shall exclude the classical fourth axiom *S4* for the sum, for which $\forall\, t \in \mathcal{T}\ .\ \exists\, t' \in \mathcal{T} : t + t' = t_0$, that states the existence of the inverse element; the subtraction function would not be definable, since its definition would be $s - t = s + (-t)$ with $-t$ inverse element of t. For our purposes it suffices axiom *S4'* instead of *S4*. The relation $s \sqsubseteq t$ is exactly equivalent to $t \sqsupseteq s$. The symbol $s \sqsubset t$ is an abbreviation for $s \sqsubseteq t \wedge s \neq t$. Likewise for the symbol $s \sqsupset t$.

Let $\mathcal{T}$ be a time-point domain. $\mathcal{T}$ is a *discrete domain* if the axioms *S1*, …, *S4'* for the sum operation, and *O1*, …, *O6* for the order relation, are defined for its basic structure. The time-point domain $\mathcal{T}$ is a *continuous domain* when the following well known axiom is further defined for its structure:

> *O7* : Let be A and B non empty subsets of $\mathcal{T}$ such that $a \sqsubseteq b$ for all $a \in A$ and $b \in B$. Then exists $\xi \in \mathcal{T}$ such that $a \sqsubseteq \xi \sqsubseteq b$ for all $a \in A$ and $b \in B$. (Axiom of completeness)

Clearly additional axioms must be introduced for characterizing the temporal structure we like to deal with. In the sequel we assume time is linearly ordered, that is we assume *O4*.

Definition 3.2.1 (world) *Let be $\mathcal{P}$ the set of "past time-points" with respect to a given ν, defined as $\mathcal{P} = [t_0, \nu] \subseteq \mathcal{T}$. Let $\mathcal{J}$ be the domain of finite developments, and $\mathcal{J}^\star$ the domain of complete developments. A* world *is defined as a transformation $W \colon \mathcal{J} \to \mathcal{J} \cup \mathcal{J}^\star$, i.e. a subset of $\mathcal{J} \times (\mathcal{J} \cup \mathcal{J}^\star)$ such that $\forall J \in \mathcal{J} . \exists J' \in \mathcal{J} : W(J, J')$, and if $W(J, J')$ then $J' = \langle \mathcal{P}', M', H', \mathcal{A}', \mathcal{C}' \rangle$ is a correct revision of $J = \langle \mathcal{P}, M, H, \mathcal{A}, \mathcal{C} \rangle$ such that $\mathcal{P}' = \mathcal{P} \cup (\nu, \nu']$, where ν is the largest member of $\mathcal{P}$, the restriction of H' to $[t_0, \nu]$ equals H, and $\mathcal{C}' \subseteq \mathcal{C}$.* □

The definition 3.2.1 is equivalent to Sandewall's definition of (discrete) world if we assume $\mathcal{T} = \mathbb{N}$, $t_0 = 0$ and the function *clock* as the successor function of Peano's arithmetics.

3.3 Continuous change

The world has a set theoretic description $\langle Infl, Trajs \rangle$, given in terms of influenced features and corresponding trajectories. We remind that for a given action E and state σ_s, $Infl(E, \sigma_s)$ represents the set of those features which may be affected if the action E is initiated in the state σ_s, while $Trajs(E, \sigma_s)$ represents the set of possible trajectories of the action E if the action E is initiated in σ_s, and consists in a set of finite nonempty sequences of partial states each of which assigns values to those features appearing in $Infl(E, \sigma_s)$ but not to other features. Each single trajectory defines an action alternative. The set theoretic description consists in a propositional table where the description of each trajectory is given in a detailed case by case: if σ_s is the state of the world at time s, starting from $s{+}1$ a trajectory in $Trajs(E, \sigma_s)$ gives a finite nonempty sequence $\langle \sigma_{s+1}, \ldots, \sigma_t \rangle$ of tuples in $\mathcal{F} \times \mathcal{V}$, up to a certain time-point t, $t \neq s{+}1$, at which the trajectory ends. For example:

Starting state σ	$Infl(Fire, \sigma)$	$Trajs(Fire, \sigma)$
$\{a : true, l : true\}$	$\{a, l\}$	$\langle \{a : true, l : false\}, \{a : false, l : false\} \rangle$
$\{a : true, l : false\}$	$\varnothing$	$\langle \varnothing \rangle$
$\{a : false, l : true\}$	$\{l\}$	$\langle \{l : false\} \rangle$
$\{a : false, l : false\}$	$\varnothing$	$\langle \varnothing \rangle$

where $\langle \varnothing \rangle$ is the null trajectory. According to the first statement, the firing action takes place in two time units; at the fist time-step the gun becomes unloaded and the turkey is still alive, while at the second time-step the gun remains unloaded and the turkey dies.

If we assume the value domain for features is a continuous domain, that is we assume axiom *O7*, we get a continuum of possible starting states, one tuple $\langle Infl(E,\sigma), Trajs(E,\sigma)\rangle$ for each specific starting state σ and action E, and a continuum of ordered partial states for each trajectory in $Trajs(E,\sigma)$. We solve as follows. We assume to deal with a finite number of influenced features for each action, we assume that for all f_i and f_j in $Infl(E,\sigma)$ is $f_i \neq f_j$, we assume a finite number of action alternatives, and a finite number of starting states σ such that $Infl(E,\sigma) \neq \varnothing$. Concerning trajectories, we proceed as follows. A finite number of ordered partial states defines a discrete trajectory, that can be also explicitly written when reasonably small. Clearly a continuum of ordered partial states cannot be explicitly written, unless with a characteristic property that *implicitly* defines them by selection over a common domain. Because of the nonexistence of a "next partial state", the order relation between partial states must necessarily be the order relation between time-points, where each partial state consists in a set of tuples $\langle t, f, v\rangle$ in $\mathcal{T} \times \mathcal{F} \times \mathcal{V}$.

Let be $\mathcal{H}$ the set of all *observations* $\langle t, f, v\rangle$, varying t in $\mathcal{T}$, f in $\mathcal{F}$ and v in $\mathcal{V}$. The order relation $\sqsubseteq$ applies as follows on members of $\mathcal{H}$: $\langle t_1, f_1, v_1\rangle \sqsubseteq \langle t_2, f_2, v_2\rangle$ if and only if $t_1 \sqsubseteq t_2$, for all $\langle t_1, f_1, v_1\rangle$ and $\langle t_2, f_2, v_2\rangle$ in $\mathcal{H}$. Since we assumed axiom *O4*, then $(\mathcal{H}; \sqsubseteq)$ is a totally ordered set.

Definition 3.3.1 (partial fluent) *A* partial fluent *is a function* $\varphi\colon \mathcal{T}^3 \rightarrow \mathcal{V}$ *which definitional domain is* $Dom(\varphi) = \{\langle x, y, z\rangle \in \mathcal{T}^3 : x \sqsubseteq y \sqsubseteq z\}$. *We define length of* $\varphi(s, \tau, t)$, *where* $\varphi(s, \tau, t)$ *is the invocation of a partial fluent, the unique member* l *of* $\mathcal{T}$ *such that* $s + l = t$.

Partial fluents are *single-feature trajectory descriptors.* Partial fluents may be *internal* and static, *internal* and dynamic, or *external* of two different species.

An internal static partial fluent is a computable function which definition is known and does not change passing time. For example, an internal static partial fluent φ can be a mathematical function $\varphi : \mathbb{R}^3 \rightarrow \mathbb{R}$, or a function $\varphi : \mathbb{N}^3 \rightarrow \{true, false\}$ as well. For a given feature f_j such that several actions are concurrently running and influencing f_j, instead of separately consider all involved partial fluents we can just consider their combined function φ_j, like in Newtonian physics. An internal static partial fluents allows also to simulate lack of knowledge when complete knowledge about the change is not required, or the change is not exactly known; its definition may be as follows:

$$\varphi_j(s, \tau, t) = \begin{cases} w & \text{if } \tau = s \\ unknown & \text{if } \tau \in (s, t) \\ v & \text{if } \tau = t \end{cases}$$

where w is the expected value of a precondition involving the feature f_j, *unknown* represents $\varkappa f_j$ (occluded f_j), and v is the terminal value of the post-condition. Clearly the tuple $\langle t, f, unknown\rangle$ is an abbreviation for $\bigvee_i \langle t, f, v_i\rangle$, varying i over all possible tuples $\langle t, f, v_i\rangle$ in $\mathcal{H}$ (assuming a continuous value domain for features, i is necessarily a continuous index), and express the *don't know* non-determinism.

An internal dynamic partial fluent is a computable function which definition is known and is allowed to change, passing time, according to internal criteria. Its definition may be updated, for instance, during a learning process.

An <u>*external partial fluent of first specie*</u> *is a computable function which definition is known and is allowed to change passing time according to external criteria.* An example of external partial fluent of first specie is the following. Suppose the function $\varphi_i : \mathbb{R}^3 \to \mathbb{R}^3$ represents the i-th airway within a certain airspace. The invocation $\varphi_i(s, \tau, t)$ represents the trajectory of an airplane, and is generated by an air-traffic scheduler, which is supposed to be "external" to the airplane co-pilot system, so that for all τ in $[s, t]$ no other airplane is using the i-th airway. The co-pilot system "knows" φ_i at time τ, and may drive the airplane accordingly.

An <u>*external partial fluent of second specie*</u> *is a partial function which definition is not known, its values are not predictable, and the function itself may also be a non computable function.* A simple example of external partial fluent of second species is the not-predictable trajectory of another agent of the same dynamical system.

Under the $\mathcal{K}$ epistemological assumption, external partial fluents of the second specie are not allowed, in fact their definition is not known and the information they provide it is not predictable, so that prediction problems could not be solved.

Definition 3.3.2 (trajectory) *Let be E an action and σ a state of the world at time s such that $Infl(E, \sigma) \neq \varnothing$. Let be n the number of alternatives of the action E initiated in σ and let be $\varphi_{1j}, \ldots, \varphi_{nj}$ the partial fluents that are associated to each feature f_j in $Infl(E, \sigma)$, where for all φ_{ih} and φ_{ik} associated to $f_h, f_k \in Infl(E, \sigma)$, $i = 1, \ldots, n$, is $s \notin Dom(\varphi_{ih})$, $s \notin Dom(\varphi_{ik})$ and $length(\varphi_{ih}) = length(\varphi_{ik})$. The* <u>*i-th trajectory*</u> *of E initiated in σ is the totally ordered set $(T_i(E, \sigma); \sqsubseteq)$, where $T_i(E, \sigma)$ is defined as follows:*

$$T_i(E, \sigma) = \{\langle \tau, f_j, \varphi_{ij}(s, \tau, t)\rangle \in \mathcal{H} : \tau \in Dom(\varphi_{ij}) \text{ and } f_j \in Infl(E, \sigma)\}$$

Given a trajectory $T_i(E, \sigma)$ the set $h(\tau) = \{\langle t, f, v\rangle \in T_i(E, \sigma) : t = \tau\}$ will represent the <u>*partial state*</u> *of the world at time τ according to $T_i(E, \sigma)$. If $s = Min\{\tau : \langle \tau, f, v\rangle \in T_i(E, \sigma)\}$ and $t = Max\{\tau : \langle \tau, f, v\rangle \in T_i(E, \sigma)\}$, we define* <u>*length*</u> *of a trajectory $T_i(E, \sigma)$ the unique member l of $\mathcal{T}$ such that $s + l = t$. A null trajectory is an empty trajectory.* □

Definition 3.3.3 (trajectory set) *Let be E an action and σ a starting state. The set $Trajs(E, \sigma)$ is the set of all possible trajectories of E initiated in σ.*

3.3.1 Action laws

From the set theoretic description of a world as a pair $\langle Infl, Trajs\rangle$ we shall now construct a corresponding logical formula, called the *Full Trajectory Normal Form* (FTNF) for the action laws.

Definition 3.3.4 (trajectory formula) *A* trajectory formula *for a given feature f_j in $\mathcal{F}$ is defined as a trajectory specification of the form*

$$\forall \tau \in [s, t] \subset \mathcal{T} \,.\, [\tau] f_j \doteq \varphi_j(\tau)$$

where the descriptor φ_j is a partial fluent *defined over $D \subseteq [s, t] \subset \mathcal{T}$, $s \neq t$.*

Definition 3.3.5 *Let be M a mapping and $\theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\}$ a valuation. The meaning of a trajectory formula $\forall\, \tau \in [s,t] \subset \mathcal{T}\,.\,[\tau]f_j \doteq \varphi_j(s,\tau,t)$ is the lattice $(S_j(\theta); \sqsubseteq)$, where the domain $S_j(\theta)$ is defined as follows:*

$$S_j(\theta) = \{\langle \tau, f_j, v\rangle \in \mathcal{H} : v = \varphi_j(s\theta, \tau, t\theta) \text{ and } \tau \in Dom(\varphi_j)\}$$

In the sequel, S will represent a trajectory formula and $S(\theta)$, for a given valuation θ, will represent its meaning.

Definition 3.3.6 (action law) *An action law in Full Trajectory Normal Form with no alternatives is a mapping of an action statement to a conjunction of trajectory formulas and is written in the form*

$$[s,t]A \Rrightarrow \textstyle\bigwedge_{j=1}^{m} S_j$$

where $A \in \mathcal{E}$, the conjunction $\bigwedge_{j=1}^{m} S_j$ represents a trajectory set, and each S_j is the corresponding trajectory formula for the feature f_j. The Full Trajectory Normal Form for actions with alternative results is the mapping

$$[s,t]A \Rrightarrow \textstyle\bigvee_{i=1}^{n}(\bigwedge_{j=1}^{m} S_{ij})$$

where the action occurrence $[s,t]A$ is expanded into a formula in Full Disjunctive Normal Form, *that is into a disjunction of conjunctions of trajectory formulas S_{ij}, each of which corresponds to the feature f_j in the alternative i.* □

Conventions. Within the same action definition alternatives involve the same set of features and partial fluents have the same length, which defines the length of the action itself. Each action type has influence over those features occurring in the action definition, while all other feature values remain unchanged, and there is no change during those periods when there is no action.

We now show the above FTNF is an equivalent and compact version of the FTNF for the discrete case. Let Pre be a conjunction of statements like $[s]f_n = v_n$, and let $Post$ be a conjunction of statements like $[s,t]f_m := v_m$; the former represents the initial expected values for those features involved in a precondition, while the latter describes the trajectory for those features involved in a postcondition.

Basic case:	$[s,t]A \Rrightarrow [s,t]f := v$
Random effect:	$[s,t]A \Rrightarrow [s,t]f := v \vee [s,t]f := \neg v$
Simple conditional:	$[s,t]A \Rrightarrow Pre \Rightarrow Post$
Disjunctive preconditions:	$[s,t]A \Rrightarrow (Pre_1 \vee \ldots \vee Pre_k) \Rightarrow Post$
Disjunctive postconditions:	$[s,t]A \Rrightarrow Pre \Rightarrow (Post_1 \vee \ldots \vee Post_k)$
Generic alternative:	$[s,t]A \Rrightarrow C_1 \vee \ldots \vee C_k$

The basic case is trivial. The generic alternative is a disjunction of C_i statements, each of which is like the body $Pre \Rightarrow Post$ of a simple conditional, where Pre may also be empty, while an empty set of postconditions is not allowed. For each Pre_i in C_i and Pre_j in C_j, although with possibly different values, they involve the same set of features. The random effect is that particular instance of the generic alternative for which two or more distinct values are non-deterministically assigned to the same feature. In order to not compromise scenario description's consistency, the disjunction is intended to be exclusive. The conditional with disjunctive preconditions is an alternative writing for a generic alternative where each C_i has the same postcondition $Post$. The conditional with disjunctive postconditions is an alternative writing for a generic alternative where each C_i has the same precondition Pre. The simple conditional, by exploiting partial fluents expressiveness, may be rewritten as follows: (1) write a partial fluent φ_n for each element $[s]f_n = v_n$ in Pre; (2) write a partial fluent ψ_m for each element $[s,t]f_m := v_m$ in Post; (3) for each partial fluent ψ_m such that there exists a φ_n for the same feature, *complete* ψ_m with the definition of φ_n, then eliminate φ_n; (4) make a conjunction of all obtained partial fluents. For example, the famous action law $[s,t]Fire \Rrightarrow [s]l \Rightarrow ([s,t]l := false \wedge [s,t]a := false)$ is equivalent to $[s,t]Fire \Rrightarrow \forall\, \tau \in [s,t] \,.\, [\tau]a \doteq to\_die(s,\tau,t) \wedge [\tau]l \doteq to\_fire(s,\tau,t)$ where the partial fluents to_die and to_fire are defined as follows:

$$to\_die(s,\tau,t) \quad = \begin{cases} unknown & \text{if } \tau \in (s,t) \\ false & \text{if } \tau = t \end{cases}$$

$$to\_fire(s,\tau,t) \quad = \begin{cases} true & \text{if } \tau = s \\ unknown & \text{if } \tau \in (s,t) \\ false & \text{if } \tau = t \end{cases}$$

The case of alternative results of actions, which assumption is represented with the **A** ontological assumption, covers the following cases: random effect, simple conditional (that corresponds to the **A**d ontological assumption), conditional with disjunctive postconditions, and the generic alternative. Then the **IA** ontological family covers everything of the above list, except the case of disjunctive preconditions, since it does not allow alternative results of actions, according to the conventional direction of time.

3.3.2 Trajectory-semantics world

We finally arrived to the clou definition of trajectory-semantics world, which is of prime relevance in the process of generating the intended model set. Let be H a finite history over $[t_0, t]$, let be E an action, σ a state of the world at time t, and let be $T_i(E,\sigma)$ a trajectory such that $l = length(T_i(E,\sigma)) \neq 0$. Then $H \rhd T_i(E,\sigma)$ is the finite history H' over $[t_0, t+l]$ defined as follows:

$$H'(s) = \begin{cases} H(s) & \text{if } s \sqsubseteq t \\ H(t) \oplus h(s) & \text{if } t \sqsubset s \sqsubseteq t+l \end{cases}$$

where $h(s)$ is the partial state of the world at time s, according to $T_i(E,\sigma)$.

Definition 3.3.7 (trajectory-semantics world) *A* trajectory-semantics world *is an IDS-world which corresponds to some world description. A world W is the* corresponding world *of a world description $\langle Infl, Trajs\rangle$ if and only if it satisfies the following conditions. If $J = \langle \mathcal{P}, M, H, \mathcal{A}, \mathcal{C}\rangle$, ν is the largest member of $\mathcal{P}$ and H is defined over $[t_0,\nu]$, then $W(J,J')$ if and only if $J' = \langle \mathcal{P} \cup (\nu, clock], M, H', \mathcal{A}', \varnothing\rangle$, and either of the followings hold:*

- *exists an unique* $E \in \mathcal{E}$ *such that* $\mathcal{C} = \{\langle \nu, E \rangle\}$, $H' = H \rhd T_i(E, \sigma)$ *with* $T_i(E, \sigma) \in Trajs(E, H(\nu))$, *and* $\mathcal{A}'$ *becomes* $\mathcal{A} \cup \{\langle \nu, \mu, E \rangle\}$ *as soon as clock is equal to* μ, *where* $\mu = \nu + length(T_i(E, \sigma))$;
- $\mathcal{C} = \varnothing$, $H' = H \rhd \varnothing$, *and* $\mathcal{A}' = \mathcal{A}$. $\square$

The definition 3.3.7 is equivalent to definition 2.1.19 if we assume $\mathcal{T} = \mathbb{N}$ and the master clock is defined as follows: *clock* is initially zero, then

$$clock = \begin{cases} clock + 1 & \text{if } \ \mathcal{C} = \varnothing \\ clock + length(T) & \text{if } \ \mathcal{C} = \{\langle \nu, E \rangle\} \text{ and } T \in Trajs(E, H(\nu)) \end{cases}$$

where $\mathcal{C}$ is the current action set at $\nu = clock$.

3.4 Ego-World interaction

The *game* between a trajectory-semantics ego and a trajectory-semantics world is still made in terms of finite developments. The game, assuming sequential actions, is defined as follows:

- At time ν the ego communicates its decision to initiate a certain action or activity E, by adding the tuple $\langle \nu, E \rangle$ to $\mathcal{C}$ (we assume synchronous communication between the ego and the world). Then the move of the world consists in modifying the finite development as follows:
  - At time ν, select a trajectory T_i from $Trajs(E, H(\nu))$.
  - From ν to $\nu' = \nu + l$, $l = length(T_i(E, H(\nu)))$, the world's state changes according to the chosen T_i, that is

    $$H(\rho, f_j) = \varphi_{ij}(\nu, \rho, l) \quad \forall \, \rho \in Dom(\varphi_{ij}(\nu, \rho, l))$$
  - At time ν', add the tuple $\langle \nu, \nu', E \rangle$ to $\mathcal{A}$, and reset $\mathcal{C}$ to the empty set (this communicates a "terminated" message to the ego).

  The ego can also terminate E earlier than ν', let say at the present time $\nu'' = clock$, by removing $\langle \nu, E \rangle$ from $\mathcal{C}$ and adding $\langle \nu, \nu'', E \rangle$ to $\mathcal{A}$. In that case the world persists from ν'' until another initialization message is received from the ego.
- The ego has not decided to initiate an action at time ν, so that the current action set $\mathcal{C}$ is empty. In this case the world's state simply persists in its values at ν and at subsequent new timepoints, until an initialization message is received from the ego.

Definition 3.4.1 (scenario description) *A* scenario description *in* $\mathcal{K}$-**RA** *is defined as a five-tuple*

$$\langle \mathcal{K}, \mathcal{O}, \langle \mathbf{RA}, LAW \rangle, SCD, OBS \rangle$$

where $\mathcal{K}$ *is the assumed epistemology,* $\mathcal{O}$ *is a finite set of objects,* SCD *is a set of action occurrences, and* OBS *is a set of observation statements. The tuple* $\langle \mathbf{RA}, LAW \rangle$*, where* **RA** *is the assumed ontology and* LAW *is a set of action laws in Full Trajectory Normal Form with internal static partial fluents, is the corresponding logical description of the set theoretic description* $\langle Infl, Trajs \rangle$ *of the world, for some* $\mathcal{K}$-**RA** *world* $\square$.

3.5 Intended models

The definitions of complete development set, intended model set and classical model set for a scenario description in $\mathcal{K}$-**RA** are substantially identical to the original ones for $\mathcal{K}$-**IA** , although with right and proper different understanding.

Definition 3.5.1 (complete development set) *Let Υ be a scenario description in $\mathcal{K}$-**RA** . The set of* complete developments *of a game is obtained as follows: (1) construct the unique trajectory-semantics world W that is precisely specified by $\langle \mathbf{RA}, LAW \rangle$, then select an arbitrary trajectory-semantics ego and an arbitrary initial state; (2) generate all possible developments which can be obtained in games between them; (3) add an arbitrary M component to each development; (4) restrict the set of developments to those where all formulas in $SCD \cup OBS$ are satisfied, and where there is a one-to-one correspondence between actions in the development and action statements in SCD. This subset is $Mod(\Upsilon)$.*

Definition 3.5.2 (intended model set) *The* intended model set *for a scenario description Υ in $\mathcal{K}$-**RA** is defined as follows:*

$$\Sigma_{\mathcal{K}\text{-}\mathbf{RA}}(\Upsilon) = \{\langle M, H \rangle \mid \langle \mathcal{P}, M, H, \mathcal{A}, \mathcal{C} \rangle \in Mod(\Upsilon)\}$$

Definition 3.5.3 (classical model set) *The* classical model set *for a scenario description Υ in $\mathcal{K}$-**RA** is defined as follows:*

$$[\![\Upsilon]\!] = [\![LAW[SCD] \cup OBS]\!]$$

where $LAW[SCD]$ is intended to be the expansion of all formulae in SCD using their corresponding definitions in LAW.

The following propositions hold.

Proposition 3.5.1 $\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}(\Upsilon) \subseteq \Sigma_{\mathcal{K}\text{-}\mathbf{RA}}(\Upsilon)$.
Proof. The model set $\Sigma_{\mathcal{K}\text{-}\mathbf{RA}}(\Upsilon)$ equals $\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}(\Upsilon)$ in case of discrete worlds, while $\Sigma_{\mathcal{K}\text{-}\mathbf{IA}}(\Upsilon)$ is empty in case of continuous or almost-continuous worlds.

Proposition 3.5.2 $\Sigma_{\mathcal{K}\text{-}\mathbf{RA}}(\Upsilon) \subseteq [\![\Upsilon]\!]$ *for all Υ in $\mathcal{K}$-**RA** .*
Proof. The set $[\![\Upsilon]\!]$ allows models which do not satisfy inertia and models where there are additional actions besides those specified in SCD.

3.6 Concurrency of actions

Definition 3.6.1 (concurrent actions) *Two or more actions they are said to be* concurrent *if and only if time intervals in which they are respectively performed they overlap in at least one time point.*

Definition 3.6.2 (shared feature) *A feature is* shared *among a group of actions in a certain time interval, if and only if those actions can read or write (have influence over) the feature value during the time interval in at least one of their possible alternatives.*

Definition 3.6.3 *Two or more actions they are said to* influence *each other if and only if they share at least one feature, otherwise they are said to be* independent.

The case of full concurrency is designated, in the Features and Fluents framework, with the **C** ontological assumption, while the simpler case of concurrency for independent actions is obtained by specializing **C** with the **i** ontological subcharacteristics.

3.6.1 Independence case

No structural extensions are required for the knowledge representation and the reasoning for worlds with concurrent independent actions. When playing the game, if the ego initiates several independent actions at the same time τ, then the world simply chooses one trajectory function non-deterministically. Therefore scenario descriptions in $\mathcal{K}$-**RAC***i* are substantially identical to scenario descriptions in $\mathcal{K}$-**RA** , except for action occurrences which are now allowed to overlap in at least one timepoint. The intended and classical model set are defined in the usual fashion, and the relation between $\mathcal{K}$-**RAC***i* models and $\mathcal{K}$-**RA** models is described by the following theorem.

Theorem 3.6.1 *Given are* $\Upsilon_1, \ldots, \Upsilon_n$ *scenario descriptions, each of which is of the form* $\Upsilon_i = \langle \mathcal{K}, \mathcal{O}, \langle \mathbf{RA}, LAW_i \rangle, SCD_i, OBS \rangle$. *Assume also that* $\cup_{i=1}^{n} SCD_i$ *is a schedule appropriate for the* **RAC***i ontological family. Then:*

$$\Sigma_{\mathcal{K}\text{-}\mathbf{RAC}i} \left(\bigcup_{i=1}^{n} \Upsilon_i \right) = \bigcup_{i=1}^{n} \Sigma_{\mathcal{K}\text{-}\mathbf{RA}} \left(\Upsilon_i \right)$$

Proof. Independent concurrent actions are compositional, in fact their joint effect is the sum of their individual effects. $\Box$

Due to the physical principle of *independence of simultaneous actions*, the case of concurrency in classical physics falls within $\mathcal{K}$-**RAC***i* . According to the well known principle, if more than one action (power) applies on the same physical object, then each action applies on the object independently from the others, while the overall applied power consists in the combination (the vectorial sum) of all the involved powers. In order to include such case, the ego-world game shall be extended as follows: if several concurrent trajectory functions are associated to the same feature, then only one trajectory function will be considered for that feature as the composed trajectory function of the given trajectory functions, and its domain will be the closed and limited interval in time which minimum element is the given τ and the maximum element μ is the maximum over all involved domains. If $\circ$ represents the vectorial sum and φ^i represents the i-th vector of the i-th power, the composed trajectory formula will look as follows:

$$[s,t]A^1 \circ A^2 \circ \ldots \circ A^k \Rrightarrow \forall \tau \in [s,t] \, . \, [\tau] f \doteq \varphi^1 \circ \varphi^2 \circ \ldots \circ \varphi^k(s, \tau, t)$$

When such composed function is a constant function c, then persistence arise for the influenced feature if and only if exists a small positive $\varepsilon \in \mathcal{T}$ such that $H(\tau - \epsilon) = c(\xi)$, for all ξ in $[\tau, \mu]$. The intended and classical model set are defined in the usual fashion.

3.7 Examples

The following is the most famous example of chronicle in $\mathcal{K}sp$-$\mathbf{I}b\mathbf{A}d$.

Scenario 3.7.1 (Yale Shooting Scenario [HM86]) *The turkey Fred is initially* alive *and the gun is not* loaded*. Successively, a* load*, a* wait *and a* shoot *event occur. The common sense conclusion is the turkey dies after the shooting.*

Using the *F&F* language as in [San94a], its representation is as follows:

$$
\begin{array}{ll}
\text{OBS:} & [t_0]a\wedge\neg l \\
 & \mathbf{t}_1 < \mathrm{s}_2 \\
\text{SCD:} & [\mathrm{s}_1,\mathbf{t}_1]Load \\
 & [\mathrm{s}_2,\mathbf{t}_2]Fire \\
\text{LAW:} & [s,t]Load \Rrightarrow [s,t]l := true \\
 & [s,t]Fire \Rrightarrow [s]l \Rightarrow [s,t](l\wedge a) := false
\end{array}
$$

where $M(\mathbf{s}_1), M(\mathbf{t}_1), M(\mathbf{s}_2), M(\mathbf{t}_2) \in \mathbb{N}$, "$<$" is the common strict order in $\mathbb{N}$, and features are truth valued. As an example of chronicle in $\mathcal{K}sp$-$\mathbf{RA}d$, consider the variant of the above scenario where the additional requirement consists in representing the law for the turkey's agony, which shall be regarded as a detailed script for our favorite turkey-actor:

$$
\begin{array}{ll}
\text{OBS:} & [t_0]a \hat{=} 1.00 \\
 & [t_0]l \hat{=} 0.00 \\
 & [\Box] \sqsubset (\mathbf{t}_1,\mathrm{s}_2) \hat{=} true \\
\text{SCD:} & [\mathrm{s}_1,\mathbf{t}_1]Load \\
 & [\mathrm{s}_2,\mathbf{t}_2]Fire \\
\text{LAW:} & [s,t]Load \Rrightarrow \forall\tau\in[s,t].[\tau]l \doteq to\_load(s,\tau,t) \\
 & [s,t]Fire \Rrightarrow \forall\tau\in[s,t].[\tau]l \doteq to\_fire(s,\tau,t)\wedge[\tau]a \doteq to\_die(s,\tau,t)
\end{array}
$$

where $M(\mathbf{s}_1), M(\mathbf{t}_1), M(\mathbf{s}_2), M(\mathbf{t}_2) \in \mathbb{R}^+$, "$\sqsubset$" is the order relation of the given temporal structure, and partial fluents to_load, to_fire and to_die are defined as follows:

$$
to\_load(s,\tau,t) = \begin{cases} unknown & \text{if } \tau\in(s,t) \\ 1.0 & \text{if } \tau = t \end{cases}
$$

$$
to\_fire(s,\tau,t) = \begin{cases} \frac{-2\tau}{t-s} + \frac{t+s}{t-s} & \text{if } \tau\in\left[s,\frac{s+t}{2}\right] \\ 0.00 & \text{if } \tau\in\left(\frac{s+t}{2},t\right] \end{cases}
$$

$$
to\_die(s,\tau,t) = \begin{cases} 1.0 & \text{if } \tau\in\left(s,\frac{s+t}{2}\right] \\ \frac{1}{t-s} * \sqrt{(t-s)^2-(\tau-s)^2} & \text{if } \tau\in\left(\frac{s+t}{2},t\right] \end{cases}
$$

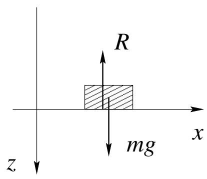
The partial fluent to_load: $(s,t] \subset \mathbb{R}^+ \to [0,1] \subset \mathbb{R}^+$, which describes the trajectory for the encapsulated *Load* action, express the condition for the gun to have a not precisely specified value for the feature *loaded* during the whole interval (s,t), while the gun is surely loaded at $\tau = t$.

The partial fluent to_fire: $[s,t] \subset \mathbb{R}^+ \to [0,1] \subset \mathbb{R}^+$, that is a continuous function, express the condition for the gun to be loaded at s, unloading during $\left(s,\frac{s+t}{2}\right)$, then unloaded during the whole interval $\left[\frac{s+t}{2},t\right]$.

The partial fluent $to\_die \colon (s,t] \subset \mathbb{R}^+ \rightarrow [0,1] \subset \mathbb{R}^+$, that is a continuous function, express the condition for the turkey to be alive during $\left(s, \frac{s+t}{2}\right]$, agonizing during $\left(\frac{s+t}{2}, t\right)$, and dead at $\tau = t$. $\square$

We are confident that many situations from classical physics can be represented as scenario descriptions, and considered as being part of the agent's knowledge. The following is a simple example of inertia with influencing actions.

Scenario 3.7.2 (inertia) *An object of mass m lays on a horizontal plane with no friction. At every point in time, two distinct powers and no others, act on the object: the weight mg and the reaction R of the plane. Due to the principle of action and reaction, the reaction R of the plane has equal magnitude and opposite sign with respect to the action performed by the object on the plane itself, that is $R = -mg$. The common sense conclusion is the object persists in its state of rest.*



The problem domain has the following representation.

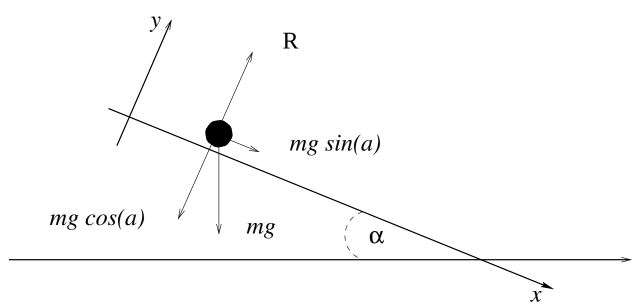
OBS:	$\varnothing$
SCD:	$[s_1, t_1]W(obj)$
	$[s_1, t_1]R(obj)$
LAW:	$[s,t]W(object) \Rrightarrow \forall\tau \in [s,t].[\tau]power(object) \doteq \varphi_1(s,\tau,t)$
	$[s,t]R(object) \Rrightarrow \forall\tau \in [s,t].[\tau]power(object) \doteq \varphi_2(s,\tau,t)$

where $\varphi_1(s,\tau,t) = mg$ and $\varphi_2(s,\tau,t) = -\varphi_1(s,\tau,t)$, for all τ in $[s,t]$. If $\circ$ is the symbol for the vectorial sum, we obtain

$$H(\tau, power(obj)) = \varphi_1(s_1,\tau,t_1) \circ \varphi_2(s_1,\tau,t_1) = 0.0 \qquad \forall\, \tau \in [s_1, t_1]$$

A well known variant of the above scenario consists in inclining the plane ...

Scenario 3.7.3 (Galileo's inclined plane) *Given is an inclined plane with no friction, and let be α the angle it makes with respect to the horizontal plane. We lean an object of mass m on the inclined plane, then we drop the object. The common sense conclusion is the object starts moving along the inclined plane.*

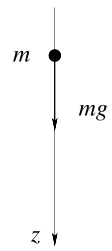


The problem domain has exactly the same representation of the above scenario, except for OBS, where the initial heigth h of the object and its initial distance $l \neq h$ from the bottom are now represented, and for φ_1, which definition is a generalization of the above one: $\varphi_1(s, \tau, t) = mg\ sin(\alpha) \circ mg\ cos(\alpha)$ for all τ in $[s, t]$, where $sin(\alpha) = \frac{h}{l}$. Clearly is

$$H(\tau, power(obj)) = \varphi_1(s_1, \tau, t_1) \circ \varphi_2(s_1, \tau, t_1) = mg\ sin(\alpha) \qquad \forall\, \tau \in [s_1, t_1]$$

If $0 < \alpha < \frac{\pi}{2}$ then is $mgsin(\alpha) > 0$ and the object starts moving. If $\alpha = 0$ then is $mgsin(\alpha) = 0$, and we re-obtain the case of inertia with influencing actions. $\square$

Another well known variant consists in dropping the object from a certain hight h. The common sense conclusion is that the object will fall, moving of rectilinear uniformly-accelerated motion, until an horizontal plane is reached.



The representation of the problem domain consits in that variant of the inclined plane case where initially is $h = l$, that is $\alpha = \frac{\pi}{2}$, where the inclined plane is the vertical plane. Clearly no reaction R applies on the object, so that

$$H(\tau, power(obj)) = \varphi_1(s_1, \tau, t_1) = mg \qquad \forall\, \tau \in [s_1, t_1]$$

where t_1 is the time-point at which the object reaches the horizontal plane.

The following is the famous second evidence against motion of Zenone, which is properly an example of rectilinear and uniform motion.

Scenario 3.7.4 (Achilles and the tortoise) *The tortoise defied the swift-footed Achilles to a single contest. They'll compete in running, which according to the tortoise would have seen her victorious, on condition of a small initial advantage (one step) which Achilles whould have had allowed.*

The problem domain has the following representation:

OBS:	$[0.0]position(Achilles) \hat{=} 0.0$
	$[0.0]position(Tortoise) \hat{=} 1.0$
	$[\square] \sqsubset (\mathrm{s}_2, \mathrm{s}_1) \hat{=} true$
SCD:	$[\mathrm{s}_1, t_1]Run(Achilles)$
	$[\mathrm{s}_2, t_2]Run(Tortoise)$
LAW:	$[s, t]Run(athlete) \Rrightarrow \forall \tau \in [s, t].[\tau]position(athlete) \dot{=} \varphi(s, \tau, t)$

where $\mathbf{s}_1$ and $\mathbf{s}_2$ are temporal constants, while s, t, t_1 and t_2 are variables. The paradox of Zenone assumes rectilinear motion with null speed variation, therefore the partial fluent φ is $\varphi(s, \tau, t) = V(athlete) * \tau$ for all τ in $[s, t]$, and represents the covered space by a certain running athlete. The velocity

$V(athlete)$ is a certain constant velocity V_A for Achilles, and a certain constant velocity V_T for the tortoise, where $V_A > V_T > 0$. The description of the problem domain falls within $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$.

There exists a point in time and space where Achilles reaches the tortoise ?

Let be A and T the points in space that represent Achilles and the tortoise. If d is the initial distance between A and T, then the covered spaces S_T and S_A are described by the law for the uniform motion: $S_A(\tau) = V_A * \tau$ and $S_T(\tau) = d + V_T * \tau$. Let denote with τ_1 the point in time at which A reaches the new position $S_T(\tau_1)$, that is $S_A(\tau_2) = S_T(\tau_1)$, et cetera. We can define the succession (τ_n) as follows: $S_A(\tau_1) = d$ and $S_A(\tau_{n+1}) = S_T(\tau_n)$, so that:

$$S_A(\tau_{n+1}) = V_A * \tau_{n+1} = d + V_T * \tau_n = S_T(\tau_n)$$

Since $V_A * \tau_1 = S_A(\tau_1) = d$, then is $\tau_1 = \frac{d}{V_A}$ and $\tau_{n+1} = \frac{d+V_T*\tau_n}{V_A}$. Note that $\tau_{n+1} - \tau_n = \left(\frac{V_T}{V_A}\right)^n * \tau_1$. Zenone asserts Achilles will never reach the tortoise, that is the sum of all intervals $(\tau_{n+1} - \tau_n)$ is infinite. On the contrary is: $\tau_1 + \sum_{k=1}^{+\infty}(\tau_{k+1} - \tau_k) = \tau_1 + \sum_{k=1}^{+\infty}\left(\frac{V_T}{V_A}\right)^k * \tau_1 = \tau_1 * \left(1 + \sum_{k=1}^{+\infty}\left(\frac{V_T}{V_A}\right)^k\right) = \tau_1 * \sum_{k=0}^{+\infty}\left(\frac{V_T}{V_A}\right)^k \overset{*}{=} \tau_1 * \frac{1}{1-\frac{V_T}{V_A}} = \frac{\tau_1 * V_A}{V_A - V_T} = \frac{d}{V_A - V_T} = \tau^*$, in fact (*) is $\frac{V_T}{V_A} < 1$. Therefore the swift-footed Achilles reaches the tortoise at τ^*, in fact is $S_A(\tau^*) = S_T(\tau^*)$, that is $[\tau^*]position(Achilles) = [\tau^*]position(Tortoise)$. Clearly the ego-world game includes such development, which is the mostly preferred one. □

The reader will surely not blame the author for the mathematical solution. The existence of $\tau^\star$, via common sense reasoning, it is hard to find. Over more than two thousand years of philosophical disquisitions, in fact, the paradox of Zenone still remains an unsolved problem within pure reasoning.

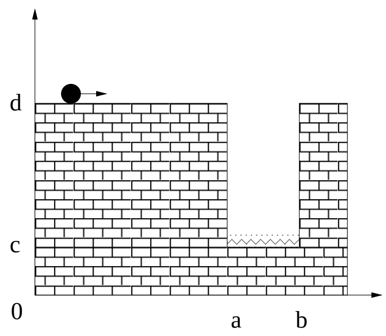
The following problem can be solved likewise the "Achilles and the tortoise" problem. Given is an air-space, and two colliding air-ways in it, let say φ_1 and φ_2. Suppose the airplane a_1 is using the air-way φ_1, a_2 is using φ_2, and both move of rectilinear and uniform motion. Suppose also that a_1 and a_2 have an emergency space distance of respectively d_1 and d_2 space units, so that if a_i is at distance d_i from another airplane, then a_i receives a warning message and performs a safety action in order to avoide the collision. Given the initial position of the respective airplanes at time t_0, it is very simple to write an action law such that the property $alert(a_i)$ becomes true if the following condition holds:

$$H(\nu)position(a_i) - H(\tau)position(a_i) = d_i$$

where ν is the present timepoint and τ is the point in time where the collision will otherwise take place.

We conclude with the following example, which is a variant of the similar example appeared in [San89b]. Its solutions, at this point, is straight forward and given for granted.

Scenario 3.7.5 (A ball falling into shaft) *A ball is moving of rectilinear and uniform motion towards a shaft. The shaft has sand on the bottom. When the ball reaches the point a, it falls into shaft with rectilinear uniformly-accelerated motion, without bounching against the walls of the pit. The bottom of the shaft perfectly absorbs the impact. The common sense conclusion is the ball stops moving when it reaches the bottom.*



Chapter 4

Fix-point Semantics

In this chapter we introduce an algebraic characterization of that subset of the intended model set for scenario descriptions in $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$ where strict inertia in continuous time and discrete deterministic change are allowed. We assume the reader familiar with fix-point theory.

4.1 The language

Let $\mathcal{T}$ be the time-point domain, $\mathcal{F}$ the set of all feature symbols, $\mathcal{V}$ the domain of all feature values, and $\mathcal{E}$ be the set of all action symbols. Let $(\mathcal{H};\sqsubseteq)$ be the lattice which elements, members of $\mathcal{H} = \mathcal{T}\times\mathcal{F}\times\mathcal{V}$, are called *observations*, and the order relation $\sqsubseteq$ is applied as follows: $\langle t_1, f_1, v_1\rangle \sqsubseteq \langle t_2, f_2, v_2\rangle$ iff $t_1 \sqsubseteq t_2$ and $f_1 = f_2$, for all $\langle t_1, f_1, v_1\rangle$ and $\langle t_2, f_2, v_2\rangle$ in $\mathcal{H}$. The tuple $\langle t, f, unknown\rangle$ is an abbreviation for $\langle t, f, \bigvee_i v_i\rangle$ varying i over all the possible tuples $\langle t, f, v_i\rangle$ in $\mathcal{H}$. Let $(\mathcal{G};\sqsubseteq)$ be the lattice which elements, members of $\mathcal{G} = \mathcal{T}\times\mathcal{T}\times\mathcal{E}$, are called *rigid occurrences of actions*, and the order relation $\sqsubseteq$ is applied as follows: $\langle s_1, t_1, A_1\rangle \sqsubseteq \langle s_2, t_2, A_2\rangle$ iff $s_1 \sqsubseteq s_2$, for all $\langle s_1, t_1, A_1\rangle$ and $\langle s_2, t_2, A_2\rangle$ in $\mathcal{G}$. Let now be Υ a scenario description in $\mathcal{K}$-$\mathbf{RAC}i$. The OBS part of Υ is a sub-lattice of the lattice $(B_\Upsilon;\sqsubseteq)$, which elements are members of $B_\Upsilon = \{\langle t, f, v\rangle \in \mathcal{H}: f \in F_\Upsilon\}$, where F_Υ is the set of all features occurring in Υ. Each tuple in OBS specifies a known observation. The SCD part of Υ, whether ground because of the mapping M, is a sub-lattice of the lattice $(\mathcal{G};\sqsubseteq)$. Each tuple in SCD specifies the starting time, the ending time and the action symbol of an action scheduled for execution. The function $\Rrightarrow: \mathcal{G} \to 2^{\mathcal{H}}$ maps each schedule's occurrence in a set of nonempty lattices of observations. The function $\Rrightarrow$ is parametric on the action type, and the LAW part of Υ consists in the definition of $\Rrightarrow$ as a set of action laws in Full Trajectory Normal Form, one law for each action type. The Full Trajectory Normal Form for action laws, we remind, is a mapping $[s,t]A \Rrightarrow \bigvee_{i=1}^{n}\bigwedge_{j=1}^{m} S_{ij}$, for which the action occurrence $[s,t]A$ is *expanded* into a formula in Full Disjunctive Normal Form, that is into a disjunction of conjunctions of trajectory formulas S_{ij}, each of which corresponds to the feature f_j in the alternative i.

We now consider scenario descriptions as being recursive definitions for problem domain specification. Observations in OBS and occurrences of actions in

SCD may be regarded as true propositions, while action laws may be regarded as being *rules* which, from a number of observations and occurrences we already showed being true propositions, permit to infer others true propositions. In the specific case of action laws, inferred propositions can just be observations. The denotation for a scenario description, like for every recursive definition, is defined to be the least fix-point of a transformation associated with the scenario description itself.

4.2 The semantics

Let Υ be a scenario description in the subclass $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$. An interpretation for Υ is a whatever subset for B_Υ. The set 2^{B_Υ}, which is the set of all interpretations for Υ, is a complete lattice under the partial order of set inclusion. The top element of this lattice is B_Υ and the bottom element is the empty set. For all $I \in 2^{B_\Upsilon}$, the transformation $T_\Upsilon: 2^{B_\Upsilon} \to 2^{B_\Upsilon}$, is defined as follows:

$$\begin{aligned} T_\Upsilon(I) = \{ & \langle \tau, f, v\rangle \in B_\Upsilon : \\ & \langle \tau, f, v\rangle \in OBS \text{ and } \tau = 0, \text{ or exists } \langle \_, \_, A\rangle \in SCD \text{ such that} \\ & \langle \tau, f, v\rangle \in Consequents(\tau, A) \text{ and } Antecedents(\tau, A) \subseteq I\} \end{aligned}$$

where the functions *Antecedents* and *Consequents* are defined as follows:

$$Antecedents: \mathcal{T} \times \mathcal{E} \to 2^{B_\Upsilon}$$

$$\begin{aligned} Antecedents(\tau, A) = \{ & \langle s\theta, f_j, \varphi_j(s\theta, s\theta, t\theta)\rangle \in B_\Upsilon \; : \\ & \text{exists } \langle \mathbf{s}, \mathbf{t}, A\rangle \in SCD, \\ & \text{exists } [s,t]A \Rrightarrow \textstyle\bigwedge_{j=1}^{m} S_j \in LAW \text{ and} \\ & \text{exists a valuation } \theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\} \\ & \text{such that } s\theta \sqsubseteq \tau \sqsubseteq t\theta \text{ and } \langle s\theta, f_j, \varphi_j(s\theta, s\theta, t\theta)\rangle \in S_j(\theta)\} \end{aligned}$$

$$Consequents: \mathcal{T} \times \mathcal{E} \to 2^{B_\Upsilon}$$

$$\begin{aligned} Consequents(\tau, A) = \{ & \langle \tau, f_j, \varphi_j(s\theta, \tau, t\theta)\rangle \in B_\Upsilon \; : \\ & \text{exists } \langle \mathbf{s}, \mathbf{t}, A\rangle \in SCD, \\ & \text{exists } [s,t]A \Rrightarrow \textstyle\bigwedge_{j=1}^{m} S_j \in LAW \text{ and} \\ & \text{exists a valuation } \theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\} \\ & \text{such that } s\theta \sqsubset \tau \sqsubseteq t\theta \text{ and } \langle \tau, f_j, \varphi_j(s\theta, \tau, t\theta)\rangle \in S_j(\theta)\} \end{aligned}$$

An interpretation I is a model for Υ if and only if $T_\Upsilon(I) \subseteq I$. In fact I is a model for Υ if and only if for all $\langle \mathbf{s}, \mathbf{t}, A\rangle$ in SCD if $Antecedents(\tau, A) \subseteq I$ then is $Consequents(\tau, A) \subseteq I$, if and only if $T_\Upsilon(I) \subseteq I$.

Proposition 4.2.1 *The operator T_Υ is a monotonic mapping.*
Proof. We have to prove that for all $I_1, I_2 \in 2^{B_\Upsilon}$, if $I_1 \subseteq I_2$ then is $T_\Upsilon(I_1) \subseteq T_\Upsilon(I_2)$. If $I_1 \subseteq I_2$, then is $I_2 = I_1 \cup \{\langle t_1, f_1, v_1\rangle, \ldots, \langle t_n, f_n, v_n\rangle\}$. Per definition of the operator T_Υ, since $I_1 \subseteq I_2$, then is $T_\Upsilon(I_2) = T_\Upsilon(I_1) \cup I_3$, where $I_3 \in 2^{B_\Upsilon}$ is the set of all consequences of those actions which preconditions are satisfied in I_2 but not in I_1 or, if no such actions exist, I_3 is simply the empty set. Since Υ is in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$, then its actions are not allowed to have an empty set of postconditions, so that I_3 it may never be an empty set for successfully executed actions. Then $T_\Upsilon(I_1) \subseteq T_\Upsilon(I_2)$ hold. $\square$

Observe the non-monotonic reasoning is concerned with feature values, not with temporal tokens of knowledge (observations). The truth value, or whatever in $\mathcal{V}$, of a certain property f, is assigned at time t; passing time, f may be assigned to many different values. This of course holds at the *meta-level*, while at T_Υ the syntactical object we are dealing with is just a tuple in B_Υ. So, while at the *meta-level* non-monotonic reasoning takes place, at the *symbol-level* the knowledge is monotonic.

Since T_Υ is a monotonic mapping over a complete lattice, we can also define its ordinal powers as follows:

$$\begin{aligned}
&T_\Upsilon \uparrow 0 = \{\langle 0, f, v\rangle \in B_\Upsilon : \langle 0, f, v\rangle \in OBS\}\\
&T_\Upsilon \uparrow \tau = \{\langle \tau, f, v\rangle \in B_\Upsilon : \text{exists } \langle _, _, A\rangle \in SCD \text{ such that}\\
&\quad \langle \tau, f, v\rangle \in Consequents(\tau, A) \text{ and } Antecedents(\tau, A) \subseteq T_\Upsilon \uparrow (\tau - 1)\}
\end{aligned}$$

Still because T_Υ is a monotonic mapping over a complete lattice, we can apply the Knaster-Tarski fix-point theorem [Tar55], for which the operator T_Υ admits a least fix-point that is equal to the greatest lower bound if its pre-fix-points:

$$lfp(T_\Upsilon) = glb\{I : T_\Upsilon(I) = I\} = glb\{I : T_\Upsilon(I) \subseteq I\}$$

Of course the fix-point theorem just guarantees the existence of $lfp(T_\Upsilon)$. For another well known result on fix-point theory, we can also obtain a constructive characterization of it in terms of the ordinal powers of T_Υ:

$$lfp(T_\Upsilon) \supseteq T_\Upsilon \uparrow \omega$$

Proposition 4.2.2 *The operator T_Υ is a continuous mapping.*
PROOF. We have to prove that $T_\Upsilon(lub(X)) = lub(T_\Upsilon(X))$ for every directed subset X of 2^{B_Υ}. Let X be a directed subset of 2^{B_Υ}. For all $\langle \tau, f, v\rangle \in B_\Upsilon$, $\langle \tau, f, v\rangle \in T_\Upsilon(lub(X))$ if and only if $\langle \tau, f, v\rangle \in Consequents(\tau, A)$ and $Antecedents(\tau, A) \subseteq lub(X)$ for some executed action A. This hold if and only if $Antecedents(\tau, A) \subseteq I$ for some I in X, that is iff $\langle \tau, f, v\rangle$ belongs to $T_\Upsilon(I)$ for some I in X, that is iff $\langle \tau, f, v\rangle$ belongs to $lub(T_\Upsilon(X))$.

Since T_Υ is a continuous mapping over a complete lattice, then we can apply the result of Kleene [Kle52], for which the closure ordinal τ of T_Υ is $\tau \leq \omega$, that is:

$$lfp(T_\Upsilon) = T_\Upsilon \uparrow \omega$$

The fix-point semantics for Υ is then defined as the least fix-point of the T_Υ operator. Given a problem domain Υ and a problem (query) $\langle t, f, v\rangle \in \mathcal{H}$, the solution (denotation) for $\langle t, f, v\rangle$ consists in the truth value corresponding to the characteristic function for $lfp(T_\Upsilon)$ when applied on $\langle t, f, v\rangle$ itself.

Finally, we can more reasonably assume the agent to have a finite lifetime, which is a common-sense assumption, so that the ego-world game simply results in a finite scenario description. Under this assumption we can apply another known result of the fix-point theory: since Υ is finite, then T_Υ is a monotonic operator over the finite and complete lattice $(B_\Upsilon; \sqsubseteq)$, so that T_Υ just needs a finite number of iterations in order to reach its least fix-point, more precisely there exists an ordinal τ such that $T_\Upsilon \uparrow (\tau + 1) = T_\Upsilon \uparrow \tau$. If a scenario description Υ is equivalent to the scenario $\Upsilon(\mu)$, where μ is the unplug-day of the agent or simply the maximum time-point occurring in Υ, then is $T_\Upsilon \uparrow \omega = T_{\Upsilon(\mu)} \uparrow \omega = T_{\Upsilon(\mu)} \uparrow \mu$. So, for finite scenarios, the T_Υ operator may also be understood as an operative method to the construction of the intended model set, although of no practical use. An implementation of this method is reported in appendix A.

4.3 Fix-point and Underlying Semantics

We shall now show that for scenario descriptions in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$, underlying and fix-point semantics coincide, that is the intended model set obtained by simulating ego-world games for $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$ worlds, is equal to the least fix-point of the T_Υ operator when applied on the games description Υ.

To keep in short with the notation, we write $\langle t, f, v\rangle \in \Sigma(\Upsilon)$ instead of "exists an interpretation $\langle M, H\rangle$ such that the development $\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C}\rangle$ is in $Mod(\Upsilon)$ and $H(t, f) = v$".

Theorem 4.3.1 (soundness and completeness) *For all scenario description Υ in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$ is*

$$\langle t, f, v\rangle \in \Sigma_{\mathcal{K}sp\text{-}\mathbf{IA}d\mathbf{C}i}(\Upsilon) \ \Leftrightarrow\ \langle t, f, v\rangle \in T_\Upsilon \uparrow \omega$$

where $\langle t, f, v\rangle \in \mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$.

Proof. The game starts at time $\tau = 0$. Because of the s epistemological sub-characteristics, the initial state of the world is expressed by means of observations $\langle 0, f, v\rangle$ in OBS, so that $T_\Upsilon \uparrow 0 = \{\langle t, f, v\rangle \in \mathcal{B}_\Upsilon : \langle t, f, v\rangle \in OBS$ and $t = 0\}$. The world persists, until the ego communicates its intention to perform an action. Since Υ is the game description, then no tuples may occur in SCD which starting time is τ, then $T_\Upsilon \uparrow 0 = \ldots = T_\Upsilon \uparrow \tau$. The ego, suddenly, adds the tuple $\langle \tau, E\rangle$ to the current action set $\mathcal{C}$, where τ is the current time. Then the world executes the action and terminates it at τ' by removing the tuple $\langle \tau, E\rangle$ from $\mathcal{C}$ and adding the tuple $\langle \tau, \tau', E\rangle$ to the past action set $\mathcal{A}$. The ego may also decide to terminate E earlier, let say at $\tau'' \in (\tau, \tau')$, so that it may autonomously remove the tuple $\langle \tau, E\rangle$ from $\mathcal{C}$ and add $\langle \tau, \tau'', E\rangle$ to $\mathcal{A}$. Naturally $T_\Upsilon \uparrow \tau$ retrieves a tuple $\langle \tau, \tau', E\rangle$ (or $\langle \tau, \tau'', E\rangle$) in SCD, evaluates whether the antecedents for E are satisfied and, in case, the knowledge is raised accordingly, otherwise the action is executed without any effect and the knowledge is not increased (we already proved T_Υ is rising monotonic). Because of the p epistemological sub-characteristics no observations occur in OBS at later time-points than the origo, and no alternative results of actions are allowed because of the d ontological restriction on A.

The ego-world game ranges to infinity, where the intended model set is defined. The T_Υ operator, because of the Kleene theorem, reaches the greatest lower bound of its pre-fix-points by means of its least fix-point, that is $lfp(T_\Upsilon) = glb\{I : T_\Upsilon(I) = I\} = glb\{I : T_\Upsilon(I) \subseteq I\} = T_\Upsilon \uparrow \omega$. □

Corollary 4.3.1 *For all Υ in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$ is $T_\Upsilon \uparrow \omega \subseteq [\![\Upsilon]\!]$.*
Proof. $T_\Upsilon \uparrow \omega = \Sigma_{\mathcal{K}sp\text{-}\mathbf{IA}d\mathbf{C}i}(\Upsilon) \subseteq \Sigma_{\mathcal{K}\text{-}\mathbf{RAC}i}(\Upsilon) \subseteq [\![\Upsilon]\!]$. □

The theorem 4.3.1 is the fix-point characterization of the intended model set for worlds in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$, where accurate and complete information about possibly concurrent and independent actions, complete knowledge about the initial state of the world and no information at any later state than the initial one, together with strict inertia in integer time and discrete deterministic change, are the assumed properties. The T_Υ operator gives a *deterministic simulation* of the ego-world game.

Theorem 4.3.2 *For all scenario description Υ in $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$ is*

$$\langle t, f, v\rangle \in \Sigma_{\mathcal{K}sp\text{-}\mathbf{RA}d\mathbf{C}i}\ (\Upsilon) \Leftarrow \langle t, f, v\rangle \in T_\Upsilon \uparrow \omega$$

where $\langle t, f, v\rangle \in \mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$.

Proof. The SCD part of Υ is a discrete lattice, in fact, per definition, every action has a non-empty length. Then we can define a bijective correspondence between time-points in SCD and natural numbers and apply theorem 4.3.1, which thesis now holds for that subset of the intended model set for worlds in $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$ where strict inertia in continuous time and discrete deterministic change are allowed. $\Box$

4.4 Discussion

There are a number of problems with the above fix-point characterization when approaching the full $\mathcal{K}$-$\mathbf{RAC}i$ class. A first problem is that often the initial state of the world is not known and observations at later states than the initial one may occur, that is why $\mathcal{K}sp$ and $\mathcal{K}$ are not the same epistemological family. Another problem is that actions may also have alternative results, that is why $\mathbf{A}d$ and $\mathbf{A}$ are not the same ontological family. The underlying semantics in fact is defined in terms of the ego-world game, which is a non-deterministic game just because of the initial state and the alternative results of actions. One may argue that since the world *has* that initial state, and the game is defined on top of it, then Υ should have a representation of it; we realize of course there are infinite possible initial states, and the underlying semantics is defined on all of them, not just on the mostly convenient one. Furthermore, observations at later states than the initial one have backward temporal propagation, as well as scheduled actions, and contribute to the model reduction process. We can probably generalize theorem 4.3.1 in this sense by introducing abductive temporal reasoning; the actual T_Υ operator in fact, can just make forward temporal reasoning using Υ, while both backward and forward reasoning are needed in order to handle full $\mathcal{K}$ using Υ. We also remind that $\mathcal{K}$-$\mathbf{IA}$ scenario descriptions may have more than one single model, while fix-point semantics are usually single model semantics. The last one problem is whether such T_Υ can be used for continuous worlds. In continuous time in fact, while the lattice $(B_\Upsilon; \sqsubseteq)$ does not fail to be a complete lattice, the *ordinal* powers of T_Υ are not anymore equal to the intended model set, since we used ordinals as time-points and the upward inductive process as the master clock. We also like to capture the continuous change, and no such generalization can be made on T_Υ using ordinals as time-points.

Chapter 5

Algebraic Semantics

In this chapter we define a non-simulative algebraic semantics for temporal reasoning, and we prove its correctness and completeness with respect to the underlying semantics that formally defines the $\mathcal{K}$-**RAC**i class of reasoning problems. The semantics is provably decidable.

5.1 The language

The language is substantially identical to the language of section 4.1, where scenario descriptions are regarded as being recursive definitions for problem domain specification.

5.2 The semantics

Let Υ be a scenario description in $\mathcal{K}$-**RAC**i (a problem domain) and $\langle t, f, v\rangle$ a tuple in $\mathcal{H}$ (a query). The set $2^{\mathcal{H}}$ is the set of all interpretations for Υ.

The decisional problem we shall solve consists in assigning a proper truth value to the tuple $\langle t, f, v\rangle$ using the information Υ. Equivalently, the problem consists in answering the question whether Υ, augmented with $\langle t, f, v\rangle$ in its OBS part, is a consistent set. Since more than one intended model is possible, the ground query $\langle t, f, v\rangle$ may be generalized as $\langle t, f, ?\rangle$, that is

“ *Which are the intended values of the feature f at time-point t ?* ”

and the decisional problem that is associated with, consists in identifying that single set V of values in $\mathcal{V}$ such that for all $v \in V$ the tuple $\langle t, f, v\rangle$ is *true* in at least one intended model. The subsequent task will be to conclude everything about the values of features at different points in time. Because of the proposition 3.5.2, the model set for Υ is a subset of its classical model set, so that the semantics we are going to define is a model preference semantics. The method we propose involves the explicit definition of the $History$ function in terms of a number of auxiliary functions which purpose is to expand the schedule and perform a certain model preference.

The following is the bottom-up scheme underlying the overall construction.

(1) For every $\langle t, f, v\rangle$ in OBS, $\langle t, f, v\rangle$ is true in every model. (2) For every $\langle t, f, v\rangle$ that may be generated as the effect of at least one alternative of an action law without antecedents, $\langle t, f, v\rangle$ is true in at least one model. (3) For every $\langle t, f, v\rangle$ that may be generated as the effect of at least one alternative of an action law for which every antecedent is satisfied, $\langle t, f, v\rangle$ is true in at least one model. (4) For every $\langle t, f, v\rangle$ that may be generated as the abductive effect of at least one alternative of an action law for which every consequent is satisfied, $\langle t, f, v\rangle$ is true in at least one model. (5) For every $\langle t, f, v\rangle$ for which none of the above hold, $\langle t, f, v\rangle$ is true in every model.

Requirements at points 3 and 4 are then weakened by allowing not every antecedent (consequent) satisfied, but a non empty subset of them, while others shall correspond to unknown feature values.

Differently from the fix-point semantics, the new semantics adopts a *non-simulative* approach, that is it does not simulates game developments between a trajectory-semantics ego and a trajectory-semantics world. In order to avoid the simulation, the focus is on the set $\mathcal{I}$ of all *instantiated action alternatives*, where $\mathcal{I} = \mathcal{T} \times \mathcal{T} \times 2^{\mathcal{H}}$, instead of the set $\mathcal{H}$ of all possible observations.

Definition 5.2.1 (instantiated action alternative) *Let be A an action scheduled for execution, that is $[\mathbf{s}, \mathbf{t}]A$ occur in SCD for some temporal expressions $\mathbf{s}$ and $\mathbf{t}$. Let be $[s,t]A \Rrightarrow \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} S_{ij}$ the corresponding action law in LAW, and let be θ a valuation such that $\theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\}$. We define as* instantiated alternative *of the action A, for some alternative i, the following tuple*

$$\langle s\theta, t\theta, \bigcup_{j=1}^{m} S_{ij}(\theta)\rangle \in \mathcal{T} \times \mathcal{T} \times 2^{\mathcal{H}}$$

Since the temporal structure is an ordered structure, as defined at page 22, we exploit this important property instead of beating time with the master clock. The order relation $\sqsubseteq$ associated with the chosen temporal structure, applies as follows: $\langle s_1, t_1, S_1\rangle \sqsubseteq \langle s_2, t_2, S_2\rangle$ iff $s_1 \sqsubseteq s_2$, for all $\langle s_1, t_1, S_1\rangle$ and $\langle s_2, t_2, S_2\rangle$ in $\mathcal{I}$. The order relation $\sqsubseteq$ is a partial order on $2^{\mathcal{I}}$, in fact for all I_1 and I_2 in $2^{\mathcal{I}}$ is $I_1 \sqsubseteq I_2$ iff for all occurrence $i_1 \in I_1$ and $i_2 \in I_2$ is $i_1 \sqsubseteq i_2$. Also, $2^{\mathcal{I}}$ is a complete lattice under $\sqsubseteq$, in fact the least upper bound ($\sqcup$, lub) of a collection of subsets of $\mathcal{I}$ is their minimum element and the greatest lower bound ($\sqcap$, glb) is their maximum element. Furthermore, the function $Min\colon (2^{\mathcal{I}}; \sqsubseteq) \to (2^{\mathcal{I}}; \sqsubseteq)$ is a monotone (order-preserving) function since $I_1 \sqsubseteq I_2 \Rightarrow Min(I_1) \sqsubseteq Min(I_2)$ for all $I_1, I_2 \in 2^{\mathcal{I}}$. The Min function is also complete, since $Min(\sqcup I) = \sqcup(Min(I)) = Min(I) = \sqcup(I)$ for every directed subset I of $2^{\mathcal{I}}$. The similar property holds for $Max\colon (2^{\mathcal{I}}; \sqsubseteq) \to (2^{\mathcal{I}}; \sqsubseteq)$.

Definition 5.2.2 (influence) *Given a time-point τ and an instantiated action alternative $\langle s, t, S\rangle$, where $s \sqsubset \tau \sqsubseteq t$, the set $Influence(\tau, \langle s, t, S\rangle)$ represents the set of those features which are affected by S at time-point t according to the given scenario description.*

$$Influence : \mathcal{T} \times \mathcal{I} \to 2^{\mathcal{F}}$$
$$Influence(\tau, \langle s, t, S\rangle) = \{f \in \mathcal{F} : s \sqsubset \tau \sqsubseteq t,\ \langle t, f, \_\rangle \in S \text{ and } Sat\_Pre(\langle s, t, S\rangle)\}$$

Definition 5.2.3 (current action set) *The* current action set $\mathcal{C}(\tau, f)$ *for the feature* f *at time-point* τ *is defined as follows.*

$$\begin{aligned}
&\mathcal{C}: \mathcal{T} \times \mathcal{F} \to 2^{\mathcal{I}} \\
&\mathcal{C}(\tau, f) = \{\langle s\theta, t\theta, S_i(\theta)\rangle \in \mathcal{I} : \\
&\quad \textit{exists } \langle \mathbf{s}, \mathbf{t}, A\rangle \in SCD, \\
&\quad \textit{exists } [s,t]A \Rrightarrow \textstyle\bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} S_{ij} \in LAW, \\
&\quad \textit{exists a valuation } \theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\} \textit{ and an index } i \\
&\quad \textit{such that } f \in Influence(\tau, \langle s\theta, t\theta, S_i(\theta)\rangle)\}
\end{aligned}$$

Definition 5.2.4 (past action set) *The* past action set $\mathcal{P}(\tau, f)$ *for the feature* f *at time-point* τ *is defined as follows:*

$\mathcal{P}: \mathcal{T} \times \mathcal{F} \to 2^{\mathcal{I}}$
$\mathcal{P}(\tau, f) = Max(L(\tau, f); \sqsubseteq)$ *union the set of all tuples* $\langle s_1, t_1, S_1\rangle$ *such that* $\langle\langle s_1, t_1, S_1\rangle, \langle s_2, t_2, S_2\rangle\rangle \in L(\tau, f) \times L(\tau, f)$, $t_1 \sqsubseteq s_2$, *and for all* $\langle t_1, f, v\rangle \in S_1$ *is* $\langle s_2, f, v\rangle \notin S_2$.

The set of all lower causal chains $(L(\tau, f); \sqsubseteq)$ *is defined as follows:*

$$\begin{aligned}
&L: \mathcal{T} \times \mathcal{F} \to 2^{\mathcal{I}} \\
&L(\tau, f) = \{\langle s\theta, t\theta, S_i(\theta)\rangle \in \mathcal{I} : \\
&\quad \textit{exists } \langle \mathbf{s}, \mathbf{t}, A\rangle \in SCD, \\
&\quad \textit{exists } [s,t]A \Rrightarrow \textstyle\bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} S_{ij} \in LAW, \\
&\quad \textit{exists a valuation } \theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\} \textit{ and an index } i \\
&\quad \textit{such that } t\theta \sqsubset \tau \textit{ and } f \in Influence(t\theta, \langle s\theta, t\theta, S_i(\theta)\rangle)\}
\end{aligned}$$

Definition 5.2.5 (future action set) *The* future action set $\mathcal{F}(\tau, f)$ *for the feature* f *at time-point* τ *is defined as follows:*

$\mathcal{F} : \mathcal{T} \times \mathcal{F} \to 2^{\mathcal{I}}$
$\mathcal{F}(\tau, f) = Min(U(\tau, f); \sqsubseteq)$ *union the set of all tuples* $\langle s_2, t_2, S_2\rangle$ *such that* $\langle\langle s_1, t_1, S_1\rangle, \langle s_2, t_2, S_2\rangle\rangle \in U(\tau, f) \times U(\tau, f)$, $t_1 \sqsubseteq s_2$, *and for all* $\langle s_2, f, v\rangle \in S_2$ *is* $\langle t_1, f, v\rangle \notin S_1$.

The set of all upper causal chains $(U(\tau, f); \sqsubseteq)$ *is defined as follows:*

$$\begin{aligned}
&U : \mathcal{T} \times \mathcal{F} \to 2^{\mathcal{I}} \\
&U(\tau, f) = \{\langle s\theta, t\theta, S_i(\theta)\rangle \in \mathcal{I} : \\
&\quad \textit{exists } \langle \mathbf{s}, \mathbf{t}, A\rangle \in SCD, \\
&\quad \textit{exists } [s,t]A \Rrightarrow \textstyle\bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} S_{ij} \in LAW, \\
&\quad \textit{exists a valuation } \theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\} \textit{ and an index } i \\
&\quad \textit{such that } \tau \sqsubseteq s\theta,\ \langle s\theta, f, _\rangle \in S_i(\theta) \textit{ and } Sat_Post(\langle s\theta, t\theta, S_i(\theta)\rangle)\}
\end{aligned}$$

Now (1) we analyze the contribution of feature values at τ and at past time-points with respect to τ, and we call it the *greatest lower observations* for f at τ; (2) we analyze the contribution of feature values at future time-points with respect to τ, and we call it the *least upper observations* for f at τ; (3) we determine the solution to the query $\langle \tau, f, ?\rangle$ as the *consistent union* of the g.l.o. with the l.u.o., which is properly the non-simulative $History$. A number of complete lattices has been associated to Υ, sub-lattices of $(2^{\mathcal{I}}; \sqsubseteq)$, so that temporal priorities will determine the candidate answers g.l.o. and l.u.o. .

Definition 5.2.6 (greatest lower observations) *The set of all* greatest lower observations *for the feature f at time-point τ is defined as follows:*

$$glo : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$glo(\tau, f) = \begin{cases} \{\langle t, f, v\rangle \in \mathcal{H} : t = 0\} & \text{if } LP(\tau, f) = LI(\tau, f) = \varnothing \\ (LP(\tau, f) \setminus LS(\tau, f)) \cup LI(\tau, f) & \text{otherwise} \end{cases}$$

where $LP(\tau, f)$, $LS(\tau, f)$ and $LI(\tau, f)$ are defined as follows:

$$LP : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$\begin{aligned} LP(\tau, f) = \{\langle t, f, v\rangle \in \mathcal{H} :& \\ &\text{(a) } Max\{\langle t, f, v\rangle \in OBS : t \sqsubseteq \tau\}, \text{ if } \mathcal{P}(\tau, f) = \varnothing, \\ &\text{(b) } \langle t, f, v\rangle \in S, \text{ if } \langle s, t, S\rangle \in \mathcal{P}(\tau, f)\} \end{aligned}$$

$$LS : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$LS(\tau, f) = \{\langle \_, f, v\rangle \in LP(\tau, f) : \langle s, f, v\rangle \in S \text{ and } \langle s, t, S\rangle \in \mathcal{C}(\tau, f)\}$$

$$LI : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$LI(\tau, f) = \{\langle \tau, f, v\rangle \in \mathcal{H} : \langle \tau, f, v\rangle \in S \text{ and } \langle s, t, S\rangle \in \mathcal{C}(\tau, f)\}$$

Using the information Υ, the greatest lower observations define the contribution of feature values for the feature f at τ and at past time-points with respect to τ. □

Definition 5.2.7 (least upper observations) *The set of all* least upper observations *for the feature f at time-point τ is defined as follows:*

$$luo : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$luo(\tau, f) = UP(\tau, f) \setminus US(\tau, f)$$

where $UP(\tau, f)$ and $US(\tau, f)$ are defined as follows:

$$UP : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$\begin{aligned} UP(\tau, f) = \{\langle s, f, v\rangle \in \mathcal{H} :& \\ &\text{(a) } Min\{\langle s, f, v\rangle \in OBS : \tau \sqsubseteq s\}, \text{ if } \mathcal{F}(\tau, f) = \varnothing, \\ &\text{(b) } \langle s, f, v\rangle \in S, \text{ if } \langle s, t, S\rangle \in \mathcal{F}(\tau, f), \end{aligned}$$

$$US : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{H}}$$
$$US(\tau, f) = \{\langle \_, f, v\rangle \in UP(\tau, f) : \langle t, f, v\rangle \in S \text{ and } \langle s, t, S\rangle \in \mathcal{C}(\tau, f)\}$$

Using the information Υ, the least upper observations define the contribution of feature values for the feature f at future time-points with respect to τ. □

We shall now define the method which determines whether an action is *successfully* executed, that is whether features may be influenced by its execution. Let remind the method we adopted for the fix-point semantics:

$$\begin{aligned} T_\Upsilon(I) = \{\langle \tau, f, v\rangle \in B_\Upsilon :& \\ &\langle \tau, f, v\rangle \in OBS \text{ and } \tau = 0, \text{ or exists } \langle \_, \_, A\rangle \in SCD \text{ such that} \\ &\langle \tau, f, v\rangle \in Consequents(\tau, A) \text{ and } Antecedents(\tau, A) \subseteq I\} \end{aligned}$$

the action A is successfully executed if and only if $Antecedents(\tau, A) \subseteq I$, so that $Consequents(\tau, A) \subseteq I$. Differently from the fix-point semantics, the algebraic semantics does not perform simulation and does not perform a purely deductive temporal reasoning (from premises to consequences), but a combination of abductive and deductive temporal reasoning. Furthermore the OBS part of the scenario description is intended to be a subset of the model set (point 1 of the bottom-up scheme, and point 4 in definition 3.5.1), so that some additional tests are required. The functions are as follows:

$Sat\_Obs : \mathcal{F} \times \mathcal{I} \rightarrow \{true, false\}$
$Sat\_Obs(f, \langle s, t, S\rangle)$ holds if and only if for all $\langle \tau, f, v\rangle \in OBS$ such that $s \sqsubseteq \tau \sqsubseteq t$ is $\langle \tau, f, v\rangle \in S$.

$Sat\_Pre : \mathcal{I} \rightarrow \{true, false\}$
$Sat\_Pre(\langle s, t, S\rangle)$ holds if and only if for all $\langle s, f, v\rangle \in S$ is either $\langle \_, f, v\rangle \in glo(s, f)$ or $\langle \_, f, v\rangle \in luo(t, f)$, and for all $\langle t, g, \_\rangle \in S$ is $Sat\_Obs(g, \langle s, t, S\rangle)$.

$Sat\_Post : \mathcal{I} \rightarrow \{true, false\}$
$Sat\_Post(\langle s, t, S\rangle)$ holds if and only if for all $\langle t, f, v\rangle \in S$ is $\langle \_, f, v\rangle \in luo(t, f)$ and $Sat\_Obs(f, \langle s, t, S\rangle)$.

Note that Sat_Pre is defined in terms of both glo and luo. The Sat_Pre and Sat_Post functions are the point of recursion of the overall definition.

Definition 5.2.8 (consistent union) *Let be O_1 and O_2 sets of candidate partial-states for a world at time t. The operator of* consistent union *is defined as follows:*

$\overset{*}{\cup} : 2^{\mathcal{H}} \times 2^{\mathcal{H}} \rightarrow 2^{\mathcal{H}}$
$\overset{*}{\cup}(O_1, O_2) = \{\sigma_1 \cup \sigma_2 : \sigma_1 \in O_1, \sigma_2 \in O_2 \text{ and } \sigma_1 \cup \sigma_2 \text{ is consistent }\}$

Example 5.2.1 *Let be $\sigma_{11} = \{\langle \_, f, true\rangle, \langle \_, g, true\rangle\}$, $\sigma_{12} = \{\langle \_, f, true\rangle, \langle \_, g, false\rangle\}$, $\sigma_{21} = \{\langle \_, f, true\rangle\}$, and $\sigma_{22} = \{\langle \_, f, false\rangle\}$. Assuming $O_1 = \{\sigma_{11}, \sigma_{12}\}$ and $O_2 = \{\sigma_{21}, \sigma_{22}\}$, we get $O_1 \overset{*}{\cup} O_2 = \{\sigma_{11} \cup \sigma_{21}, \sigma_{12} \cup \sigma_{21}\}$. Assuming $O_1 = \{\sigma_{11}, \sigma_{12}\}$ and $O_2 = \bigvee_i \langle \_, g, v_i\rangle$, we get $O_1 \overset{*}{\cup} O_2 = O_1$.*

Definition 5.2.9 (non-simulative history)
$History : \mathcal{T} \times \mathcal{F} \rightarrow 2^{\mathcal{V}}$
$History(t, f) = \{v \in \mathcal{V} : \langle \_, f, v\rangle \in glo(t, f) \overset{*}{\cup} luo(t, f)\}$

For a given scenario description and a tuple $\langle t, f, v\rangle \in \mathcal{H}$, we say that $\langle t, f, v\rangle$ is true, with respect to the algebraic semantics, if and only if the value v belongs to $History(t, f)$. The scenario description is implicitly given and its explicit reference on History is omitted.

The history function is defined to have values in $2^{\mathcal{V}}$, while the original function (see def. 2.1.1) was defined to have values in $\mathcal{V}$. For a given feature f in fact, the aim is to identify all the values that are intended for the feature f at the given time-point t. For example, suppose to toss a coin. According to the underlying semantics, the effect of tossing a coin results in two possible developments, one where the coin shows the head face and the other where the coin shows the cross face. The function $history(t, face)$, where t is the time-point at which the action of tossing the coin terminates, reports the *head* feature value for one ego-world game simulation, and *cross* for the other. According to the algebraic semantics we have $History(t, face) = \{head, cross\}$, for which there exists at least one intended development where the coin shows the head and another where the coin shows the cross. The problem of specifying in which development the certain feature has the certain value at the certain time-point, it is a problem out of interest for the non-simulative history.

Definition 5.2.10 (fluent) *Given a scenario description Υ and a feature f, $Fluent(f)$ represents the set of all values per f on flowing time, according to Υ.*

$Fluent : \mathcal{F} \rightarrow 2^{\mathcal{T} \times \mathcal{V}}$
$Fluent(f) = \{\langle t, v\rangle \in \mathcal{T} \times \mathcal{V} : v \in History(t, f)\}$

Definition 5.2.11 (state) *Given a scenario description Υ and a time-point τ, $State(\tau)$ represents the state of the world at time-point τ, according to Υ.*

$State : \mathcal{T} \rightarrow 2^{\mathcal{F} \times \mathcal{V}}$
$State(t) = \{\langle f, v\rangle \in \mathcal{F} \times \mathcal{V} : v \in History(t, f)\}$

Definition 5.2.12 (completion set) *Given a scenario description Υ, the set $Comp(\Upsilon)$ represents the set of all states of the world on flowing time.*

$$Comp(\Upsilon) = \{\langle t, f, v\rangle \in \mathcal{H} : \langle f, v\rangle \in State(t)\}$$

The algebraic semantics for Υ is then defined as the *completion set* of Υ. For a given query $\langle t, f, v\rangle \in \mathcal{H}$, the denotation for $\langle t, f, v\rangle$ consists in the truth value corresponding to the characteristic function for $Comp(\Upsilon)$ when applied on $\langle t, f, v\rangle$ itself. Examples of chronicle completion may be found on chapter 6.

5.3 Algebraic and Fix-point Semantics

In this section we show that for all scenario description Υ in $\mathcal{K}sp$-**IA**d**C**i and for all query $\langle t, f, v\rangle$ in $\mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$, the truth value corresponding to the characteristic function for $Comp(\Upsilon)$ when applied on $\langle t, f, v\rangle$ is equal to the truth value corresponding to the characteristic function for $lfp(T_\Upsilon)$ when applied on $\langle t, f, v\rangle$ itself.

Theorem 5.3.1 *For all scenario description Υ in $\mathcal{K}sp$-**IA**d**C**i is*

$$\langle t, f, v\rangle \in T_\Upsilon \uparrow \omega \;\;\Leftrightarrow\;\; \langle t, f, v\rangle \in Comp(\Upsilon)$$

where $\langle t, f, v\rangle$ is in $\mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$.

Proof. Since Υ is in $\mathcal{K}sp$-**IA**d**C**i , we have complete knowledge about the initial state of the world; this knowledge is represented by means of observations $\langle 0, f, v\rangle$ in the OBS part of Υ. Per definition is $T_\Upsilon \uparrow 0 = \{\langle t, f, v\rangle \in \mathcal{B}_\Upsilon : \langle t, f, v\rangle \in OBS \wedge t = 0\}$ and $glo(0, f) = (LP(0, f) \setminus LS(0, f)) \cup LI(0, f) = Max\{\langle t, f, v\rangle \in OBS : t \sqsubseteq 0\} \setminus \varnothing) \cup \varnothing$. Then $\langle t, f, v\rangle \in T_\Upsilon \uparrow 0$ iff $\langle t, f, v\rangle \in glo(0, f)$. Suppose $\langle t, f, v\rangle \in T_\Upsilon \uparrow (\tau - 1)$ iff $\langle t, f, v\rangle \in glo(\tau - 1, f)$. We shall prove that $\langle t, f, v\rangle \in T_\Upsilon \uparrow \tau$ iff $\langle t, f, v\rangle \in glo(\tau, f)$. One of the followings may hold:

- $SCD = \varnothing$. Then is $T_\Upsilon \uparrow \tau = T_\Upsilon \uparrow (\tau - 1) = \ldots = T_\Upsilon \uparrow\ 0$, and $glo(\tau, f) = glo(\tau - 1, f)$ because $LI(\tau, f) = \varnothing$.

- $SCD \neq \varnothing$. Let be $\langle \mathbf{s}, \mathbf{t}, A \rangle$ one of its elements. If $Antecedents(\tau, A) \subseteq T_\Upsilon \uparrow (\tau - 1)$ then $T_\Upsilon \uparrow \tau = T_\Upsilon \uparrow (\tau - 1) \cup Consequents(\tau, A)$, otherwise is $T_\Upsilon \uparrow \tau = T_\Upsilon \uparrow (\tau - 1)$. Likewise exists $[s, t]A \Rrightarrow \bigwedge_{j=1}^{m} S_j$ in LAW and exists a valuation $\theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\}$ such that $s\theta \sqsubset \tau \sqsubseteq t\theta$, $f = f_j$, $f_j \in Influence(\tau, \langle s\theta, t\theta, S_j(\theta) \rangle)$ and $LI(\tau, f_j) = Consequents(\tau, A)$, otherwise is $glo(\tau, f_j) = glo(\tau - 1, f_j)$ because $LI(\tau, f_j) = \varnothing$.

In each of the above situations is $glo(\tau, f) \overset{*}{\cup} luo(\tau, f) = glo(\tau, f)$, in fact for all $\langle t', f, v' \rangle \in luo(\tau, f)$ such that $\langle t', f, v' \rangle \notin glo(\tau, f)$, then also $\langle t', f, v' \rangle \notin T_\Upsilon \uparrow \tau$.
□

Theorem 5.3.2 *For all scenario description Υ in $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$ is*

$$\langle t, f, v \rangle \in \Sigma_{\mathcal{K}sp\text{-}\mathbf{IA}d\mathbf{C}i}(\Upsilon) \;\Leftrightarrow\; \langle t, f, v \rangle \in Comp(\Upsilon)$$

where $\langle t, f, v \rangle$ is in $\mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$.
Proof.

$$\begin{array}{lcl} \langle t, f, v \rangle \in \Sigma_{\mathcal{K}sp\text{-}\mathbf{IA}d\mathbf{C}i}(\Upsilon) & \Leftrightarrow & \text{(theorem 4.3.1)} \\ \langle t, f, v \rangle \in T_\Upsilon \uparrow \omega & \Leftrightarrow & \text{(theorem 5.3.1)} \\ \langle t, f, v \rangle \in Comp(\Upsilon) & & \end{array}$$

Theorem 5.3.3 *For all scenario description Υ in $\mathcal{K}sp$-$\mathbf{RAC}i$ is*

$$\langle \tau, f, v \rangle \in \Sigma_{\mathcal{K}sp\text{-}\mathbf{RAC}i}(\Upsilon) \;\Leftrightarrow\; \langle \tau, f, v \rangle \in Comp(\Upsilon)$$

where $\langle \tau, f, v \rangle$ in $\mathcal{T} \times \mathcal{F}_\Upsilon \times \mathcal{V}$.
Proof. According to theorem 5.3.2 the property holds for the sub-class $\mathcal{K}sp$-$\mathbf{IA}d\mathbf{C}i$. Therefore, according to theorem 4.3.2, the property holds for that subset of the intended model set for scenario descriptions in $\mathcal{K}sp$-$\mathbf{RA}d\mathbf{C}i$ where strict inertia in continuous time and discrete deterministic change are allowed. In order to capture the full class we need the continuous change and the alternative results of actions. Let be $\langle M, H \rangle$ an intended model in $\Sigma(\Upsilon)$ such that exists $[\mathbf{s}, \mathbf{t}]A$ in SCD, exists $[s, t]A \Rrightarrow \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m} S_j$ in LAW and exists a valuation $\theta = \{s/M(\mathbf{s}), t/M(\mathbf{t})\}$ such that $s\theta \sqsubset \tau \sqsubseteq \mathbf{t}\theta$, $f = f_j$, $f_j \in Infl(A, H(s\theta))$, and $H(\tau, f_j) = \varphi_{ij}(s\theta, \tau, t\theta) = v$, that is $\langle \tau, f_j, v \rangle \in H \triangleright S_{ij}(\theta)$. The followings hold:

$$\begin{array}{lcl} f_j \in Infl(A, H(s\theta)) & \Leftrightarrow & f_j \in Influence(\tau, \langle s\theta, t\theta, S_{ij}(\theta) \rangle) \\ & \Rightarrow & \langle s\theta, t\theta, S_{ij}(\theta) \rangle \in \mathfrak{C}(\tau, f_j) \\ & \Rightarrow & \langle \tau, f_j, v \rangle \in LI(\tau, f_j) \\ & \Rightarrow & \langle \tau, f_j, v \rangle \in glo(\tau, f_j) \end{array}$$

Due to the consistent union with $luo(\tau, f_j)$ we have $v \in History(\tau, f_j)$, $\langle f_j, v \rangle \in State(\tau)$ and $\langle t, f_j, v \rangle \in Comp(\Upsilon)$. □

5.4 Algebraic and Underlying Semantics

We shall now show that for all scenario description in $\mathcal{K}$-**RAC***i* algebraic and underlying semantics coincide, that is the model set obtained via the algebraic semantics is equal to the intended model set for all scenario description Υ in $\mathcal{K}$-**RAC***i* . To keep in short with the notation, as we previously did, we write $\langle t, f, v\rangle \in \Sigma(\Upsilon)$ instead of "exists an interpretation $\langle M, H\rangle$ such that the development $\langle \mathcal{B}, M, H, \mathcal{A}, \mathcal{C}\rangle$ is in $Mod(\Upsilon)$ and $H(t, f) = v$".

Definition 5.4.1 *Let Υ be a scenario description in $\mathcal{K}$-**RAC**i . We say that Υ is "empty per f" if and only if Υ does not hold any observation and schedule occurrence for actions involving the feature f.*

If we represent with $\mathcal{F}_\Upsilon$ is the set of all features occurring in Υ, then the set of all features f such that Υ is empty per f is designated with $\mathcal{F} \setminus \mathcal{F}_\Upsilon$.

Lemma 5.4.1 *For all scenario description Υ in $\mathcal{K}$-**RAC**i is*

$$\langle \tau, f, v\rangle \in \Sigma_{\mathcal{K}\textbf{-RAC}i}(\Upsilon) \;\Leftrightarrow\; \langle \tau, f, v\rangle \in Comp(\Upsilon)$$

where $\langle \tau, f, v\rangle \in \mathcal{T} \times \mathcal{F} \setminus \mathcal{F}_\Upsilon \times \mathcal{V}$.
Proof. For all feature $f \in \mathcal{F} \setminus \mathcal{F}_\Upsilon$ is $\langle \tau, f, v\rangle \in \Sigma(\Upsilon)$, for all τ in $\mathcal{T}$ and v in $\mathcal{V}$. In fact, per definition of ego-world game, an initial state σ_0 is given and the history H maps the origo of time-points to σ_0. The state σ_0 is (non deterministicly) given by means of initial observations, hence $\langle 0, f, v\rangle \in \sigma_0$ for all v since Υ is empty per f. The ego-world game develops and the world persists in that value for f until an action is invoked by the ego that may influence the feature. But no action has been invoked that may influence f, since Υ is empty per f; hence the world forever persists in that value for f.

Is $\langle \tau, f, v\rangle \in Comp(\Upsilon)$ as well. The property follows immediately from point *5* of the bottom-up scheme, in fact no one of the points preceding *5* may be applied to Υ, just because Υ is empty per f (precisely $LP(\tau, f)$, $LI(\tau, f)$ and $UP(\tau, f)$ are empty sets). □

Corollary 5.4.1 *For all Υ in $\mathcal{K}$-**RAC**i , $\mathcal{F}_\Upsilon = \varnothing$ implies $Comp(\Upsilon) = \mathcal{H}$.*

Theorem 5.4.1 (soundness and completeness) *For all Υ in $\mathcal{K}$-**RAC**i and $\langle \tau, f, v\rangle$ in $\mathcal{H}$*

$$\langle \tau, f, v\rangle \in \Sigma_{\mathcal{K}\textbf{-RAC}i}(\Upsilon) \;\Leftrightarrow\; \langle \tau, f, v\rangle \in Comp(\Upsilon)$$

Proof. The property holds for all $\langle t, f, v\rangle$ in $\mathcal{T} \times \mathcal{F} \setminus \mathcal{F}_\Upsilon \times \mathcal{V}$, as shown with lemma 5.4.1. Let now prove the property for features in $\mathcal{F}_\Upsilon$, where $\mathcal{F}_\Upsilon \neq \varnothing$. According to theorem 5.3.3 the property holds for all Υ in $\mathcal{K}sp$-**RAC***i* , that is for pure prediction problems, so that we suffices to prove the relation for chronicles in $\mathcal{K}$-**RAC***i* $\setminus \mathcal{K}sp$-**RAC***i* , that is for pure post-diction and pre-post-diction problems. Pure post-diction problems are solved via the *luo* function likewise pure prediction problems are solved via the *glo* function, although with similar converse technique. Concerning pre-post-diction problems, the following case holds according to definitions 3.3.7 and 3.5.1. Exists $[\mathbf{s}, \mathbf{t}]A$ in SCD and a model $\langle M, H\rangle$ such that $M(\mathbf{s}) \sqsubset \tau \sqsubseteq M(\mathbf{t})$, $f \in Infl(A, H(M(\mathbf{s})))$, $H(\tau, f) = v$, and exists $\langle M(\mathbf{s}), g, w\rangle$ precondition of A such that $H(0, g) = H(M(\mathbf{s}), g) = w$ and, since $\Upsilon \cup \{\langle \tau, f, v \rangle\}$ is a pre-post-diction problem, Υ is empty per g for all t in $[0, M(\mathbf{s}))$. Therefore the problem to determine whether $\langle \tau, f, v \rangle$ belongs to $Comp(\Upsilon)$ reduces to the problem to determine whether $\langle M(\mathbf{s}), g, w \rangle$ belongs to $Comp(\Upsilon)$, which is a pure post-diction problem. $\square$

Because of the theorem 5.4.1, the set $History(t, f)$ is the set of all values v of the feature f at time-point t such that $\langle t, f, v \rangle$ is true in at least one intended model. The theorem 5.4.1 proves the following three relevant corollaries.

Definition 5.4.2 (entailment [San94a]) *Let Υ and Υ' be scenario descriptions. We say that Υ* entails *Υ', written $\Upsilon \Vdash \Upsilon'$, if and only if $Mod(\Upsilon) \subseteq Mod(\Upsilon')$.*

Corollary 5.4.2 $\Upsilon \Vdash \Upsilon' \;\Leftrightarrow\; Comp(\Upsilon) \subseteq Comp(\Upsilon')$.

Two or more scenario descriptions are semantically equivalent if and only if they are observationally indistinguishable:

Definition 5.4.3 (equivalence [San94a]) *Let Υ and Υ' be scenario descriptions. We say that Υ is* equivalent *to Υ', written $\Upsilon \cong \Upsilon'$, if and only if $Mod(\Upsilon) = Mod(\Upsilon')$.*

Since the underlying semantics models the observable "feature", and the algebraic semantics correctly captures it, then the following proposition hold: *two or more scenario descriptions are equivalent, with respect to the observable "feature", if and only if they have the same algebraic semantics*, that is:

Corollary 5.4.3 $\Upsilon \cong \Upsilon' \;\Leftrightarrow\; Comp(\Upsilon) = Comp(\Upsilon')$.

Definition 5.4.4 (consistence [San94a]) *Let Υ and Υ' be scenario descriptions. We say that Υ is* consistent *with Υ', written $\Upsilon \,\|\!\diamond \Upsilon'$, if and only if $Mod(\Upsilon) \cap Mod(\Upsilon') \neq \varnothing$.*

Corollary 5.4.4 $\Upsilon \,\|\!\diamond \Upsilon' \;\Leftrightarrow\; Comp(\Upsilon) \cap Comp(\Upsilon') \neq \varnothing$.

A scenario description is *inconsistent* if and only if it has an empty completion set. It is easy to show that for every scenario description Υ in $\mathcal{K}$-**RAC***i* is $Comp(\Upsilon) \neq \varnothing$, that is Υ is a consistent scenario description. The consistency test for a scenario description Υ consists in checking for all feature $f \in \mathcal{F}$, whether exists a time-point $t \in \mathcal{T}$ such that $glo(t, f) \overset{*}{\cup} luo(t, f) \neq \varnothing$. The $History$ function plays a role of prime relevance in this respect, since it performs that consistency test for a given time-point t and feature f. In case of success, a non empty set of values are generated from the evaluation of the consistent union function, while an empty set is reported in case of failure. More complex methods could be involved in case of failure, such as belief revision or some other peculiar reasoning to handle the problem of *surprises in features*. A way to deal with the problem of misperception in observations, allowing for occasional incorrect observations of feature values, was recently investigated by S. Coradeschi in [Cor95, Cor96], where the case was designated with the $\mathcal{M}_0$ epistemological characteristics within the Features and Fluents Framework. The $History$ function may be extended accordingly.

5.5 Computing the completion set

Let now consider the problem of computing the completion set.

Proposition 5.5.1 *For all scenario description Υ in $\mathcal{K}$-**IA** the set $Comp(\Upsilon)$ is a recursively enumerable set.*
PROOF. We can always associate to every time-point $\tau \in \mathbb{N}$ a set of states, one state for each intended model. Each state at τ consists in a finite set of tuples in $\mathcal{F} \times \mathcal{V}$, where features in $\mathcal{F} \setminus \mathcal{F}_\Upsilon$ have "unknown" values, as shown with lemma 5.4.1. Since time-points (natural numbers) are recursively enumerable, then the proposition follows. $\square$

Real numbers are not recursively enumerable, then the similar proposition does not hold for generic scenario descriptions in $\mathcal{K}$-**RAC***i* . The set $Comp(\Upsilon)$ is so extensive that no algorithm could possibly list all its elements, so that *intended models for $\mathcal{K}$-**RAC**i are not effectively computable models.* That discouraging negative result may drive towards the formalization of purely discrete worlds, but would we really ask for more information that we could ever hope to use ? The problem to compute the *whole* intended model set for scenario descriptions in $\mathcal{K}$-**RAC***i* does not seem indeed a realistic problem, and neither does for $\mathcal{K}$-**IA** . When partial fluents are computable functions, we can always associate a value to a whatever feature at a whatever point in time, hence we can always answer the basic question "*Which is the intended value for the feature f at time-point τ ?*", that is the real problem we are dealing with. Furthermore, in case of finite scenario descriptions, the image of the $History$ function consists in a finite set of feature values, and the problem to say whether a given observation $\langle t, f, v\rangle$ is a theorem of the calculus is a decidable problem because it is decidable whether $v \in History(t, f)$. We can also restrict the $State$ function to those features occurring in the scenario description:

$$KState(t) = \{\langle f, v\rangle \in \mathcal{F}_\Upsilon \times \mathcal{V} : v \in History(t, f)\}$$

Naturally $\mathcal{F}_\Upsilon$ is a finite set, since Υ is finite, and $KState(t)$ is a finite set too. Then, the problem of computing $Comp(\Upsilon)$ can be more reasonably reduced to the problem of computing a finite number of "k-states", since we know feature values in $\mathcal{F} \setminus \mathcal{F}_\Upsilon$ range over the whole $\mathcal{V}$ set (lemma 5.4.1). The schedule in fact is a finite set, it specifies a finite number of action occurrences, and each specified action is assumed to have a (possibly unspecified) finite duration. It follows that in every intended model for the scenario description, changes due to an action are limited to a finite interval in time. Since no other events are allowed in $\mathcal{K}$-**RAC***i* besides those specified in SCD, then the set of all intervals where persistence arise is a finite set too. Concerning trajectories, suppose to have a feature f influenced in $[s, t]$, and let be φ its partial fluent. If φ emulates an encapsulated semantics or describes a discrete trajectory, then it has a finite number of values varying τ in $[s, t]$. If φ describes a continuous trajectory, and the case allows it, we can more appropriately draw its graph. In any case φ is a given computable function and, for a given time-point τ, we can always compute $\varphi(s, \tau, t)$. Then, let be $\mathcal{T}_K$ the following set:

$$\mathcal{T}_K = \{0\} \cup \{M(\mathbf{t}) \in \mathcal{T} : \langle \mathbf{t}, \_, \_\rangle \in SCD \text{ or } \langle \_, \mathbf{t}, \_\rangle \in SCD\}$$

We define the *kernel* of the completion set for Υ the following set:

$$Ker(\Upsilon) = \{\langle t, f, v\rangle \in \mathcal{T}_K \times \mathcal{F}_\Upsilon \times \mathcal{V} : \langle f, v\rangle \in KState(t)\}$$

which is the set of all memorable events for a world, let say its "album of pictures". Naturally $Ker(\Upsilon)$ is just one decidable subset of $Comp(\Upsilon)$. When needed, and apart from the kernel, feature values at specific time-points during actions execution may always be computed.

This result meets the classical requirement for every methodology of commonsense reasoning, where mathematics is used for characterizing features during time periods where their values change, and logic is used for characterizing them during time periods where they are persistent, that is over the discontinuity points. For example the problem of characterizing the set of those time-points for which a given feature f has a certain given value v

$$Holds\_at(f, v) = \{t \in \mathcal{T} : v \in History(t, f)\}$$

involves numerical analysis techniques over those periods where actions influencing f are performed, and logic over the discontinuity points.

Chapter 6

Examples of chronicle completion

Chronicle completion has been studied in the A.I. literature under the names of *prediction* and *post-diction*. The prediction problem literally consists in: *given an initial state obtained by initial observations, the set of rules governing change, and a sequence of actions, predict a state at a future point in time.* The post-diction problem is the converse similar notion. The concept of scenario completion subsumes both prediction and post-diction, but allows also for the case where one concludes facts about the state of the world both before, during, and after the actions execution.

The followings are some non exhaustive examples of chronicle completion using the algebraic semantics. In this chapter, instantiated action alternatives are represented with their corresponding action occurrence in SCD.

6.1 The Yale shooting problem

The Yale shooting problem is a well-known pure prediction problem. The scenario is described at page 31. Which are the intended values for the feature a (alive) after the shooting ?

$$History(\tau, a) = \{v \in \mathcal{V} : \langle \_, a, v\rangle \in glo(\tau, a)\overset{*}{\cup} luo(\tau, a)\} = ?$$

$$\begin{array}{rlrl}
\mathcal{C}(\mathbf{s}_2, l) & = \varnothing & \mathcal{C}(\tau, a) & = \varnothing \\
LI(\mathbf{s}_2, l) & = \varnothing & LI(\tau, a) & = \varnothing \\
\mathcal{P}(\mathbf{s}_2, l) & = \{\langle \mathbf{s}_1, \mathbf{t}_1, Load\rangle\} & \mathcal{P}(\tau, a) & = \{\langle \mathbf{s}_2, \mathbf{t}_2, Fire\rangle\} \\
LP(\mathbf{s}_2, l) & = \{\langle \mathbf{t}_1, l, 1.0\rangle\} & LP(\tau, a) & = \{\langle \mathbf{t}_2, a, 0.0\rangle\} \\
LS(\mathbf{s}_2, l) & = \varnothing & LS(\tau, a) & = \varnothing \\
glo(\mathbf{s}_2, l) & = \{\langle \mathbf{t}_1, l, 1.0\rangle\} & glo(\tau, a) & = \{\langle \mathbf{t}_2, a, 0.0\rangle\}
\end{array}$$

Where τ is such that $\mathbf{t}_2 \sqsubseteq \tau$. The set $\mathcal{P}(\tau, a)$ is well justified, in fact is $\langle \mathbf{s}_2, \mathbf{t}_2, Fire\rangle \in SCD$, $\mathbf{s}_2 \sqsubseteq \mathbf{s}_2 \sqsubseteq \mathbf{t}_2$, $\langle \_, l, 1.0\rangle \in Preconditions(\mathbf{s}_2, Fire)$ and $glo(\mathbf{s}_2, l) = \{\langle \mathbf{t}_1, l, 1.0\rangle\}$, that is $Fire$ is successfully executed. Also $\mathcal{P}(\mathbf{s}_2, l)$ is well justified, in fact $\langle \mathbf{s}_1, \mathbf{t}_1, Load\rangle \in SCD$, and $Load$ has no preconditions to test. Therefore the turkey is dead after the shooting, in fact is $History(\tau, a) =$

$\{0.0\}$, where $luo(\tau, a) = \{\langle \tau, a, unknown\rangle\}$ because no information is available at time-points later than τ.

The problem is a pure prediction problem, in fact the Least Upper Observations give no meaningful contribution.

6.2 Achilles and the tortoise

The paradox of Zeno is a pure prediction problem. The scenario is described at page 33. Suppose M such that $M(\mathbf{s}) = 0$, $V(Achilles) = 1.1$ and $V(Tortoise) = 1$, and let be $d = 5$ the initial advantage of the Tortoise. There exists a point in time and space where Achilles reaches the Tortoise ? Because of the initial assumptions, and the known solution ($\tau^* = \frac{5}{1.1-1} = 50$), is

$$\begin{array}{rlrl}
\mathcal{C}(\tau^*, p(A)) & = \{\langle \mathbf{s}, t, Run(A)\rangle\} & \mathcal{C}(\tau^*, p(T)) & = \{\langle \mathbf{s}, t, Run(T)\rangle\} \\
LI(\tau^*, p(A)) & = \{\langle \tau^*, p(A), 55\rangle\} & LI(\tau^*, p(T)) & = \{\langle \tau^*, p(T), 55\rangle\} \\
\mathcal{P}(\tau^*, p(A)) & = \varnothing & \mathcal{P}(\tau^*, p(T)) & = \varnothing \\
LP(\tau^*, p(A)) & = \varnothing & LP(\tau^*, p(T)) & = \varnothing \\
LS(\tau^*, p(A)) & = \varnothing & LS(\tau^*, p(T)) & = \varnothing \\
glo(\tau^*, p(A)) & = \{\langle \tau^*, p(A), 55\rangle\} & glo(\tau^*, p(T)) & = \{\langle \tau^*, p(T), 55\rangle\}
\end{array}$$

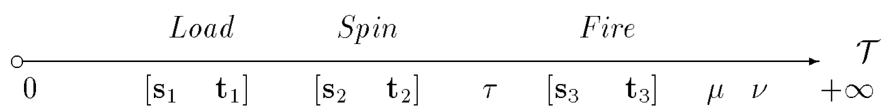
$$History(\tau^*, position(Achilles)) = History(\tau^*, position(Tortoise)) = \{55\}$$

that is Achilles reaches the Tortoise at time 50 and space 55.

6.3 The Russian turkey shooting problem

The Russian Shooting Problem is another example of pure prediction problem and, unlike the Yale Shooting Problem, involves the full **A** ontological assumption.

Scenario 6.3.1 *The turkey is initially* alive *and the gun is not* loaded*. Successively, a* load*, a* spin *(of the chamber) and a* shoot *event occur. One expects the turkey to be dead afterwards, if the gun was loaded at the time of firing, alive otherwise. (B. Haugh [Hau87], Goodwin and Trudel [GT89])*



Let a for "alive" and l for "loaded" be real-valued feature symbols, and let *Load*, *Spin* and *Fire* be the action symbols. In the language of *F&F* the problem domain has the following description in $\mathcal{K}sp$-**RA** :

$$
\begin{array}{ll}
\text{OBS:} & [\mathbf{s}_0]a\hat{=}1.0 \\
 & [\mathbf{s}_0]l\hat{=}0.0 \\
 & [\Box] \sqsubseteq (\mathbf{t}_1, \mathbf{s}_2)\hat{=}true \\
 & [\Box] \sqsubseteq (\mathbf{t}_2, \mathbf{s}_3)\hat{=}true \\
\text{SCD:} & [\mathbf{s}_1, \mathbf{t}_1]Load \\
 & [\mathbf{s}_2, \mathbf{t}_2]Spin \\
 & [\mathbf{s}_3, \mathbf{t}_3]Fire \\
\text{LAW:} & [s,t]Load \Rrightarrow \forall\, \tau \in [s,t]\,.\,[\tau]l\check{=}to\_load(s,\tau,t) \\
 & [s,t]Spin \Rrightarrow \forall\, \tau \in [s,t]\,.\,[\tau]l\check{=}\varphi_1(s,\tau,t) \vee [\tau]l\check{=}\varphi_2(s,\tau,t) \\
 & [s,t]Fire \Rrightarrow \forall\, \tau \in [s,t]\,.\,[\tau]l\check{=}to\_fire(s,\tau,t) \wedge [\tau]a\check{=}to\_die(s,\tau,t)
\end{array}
$$

where s, t and τ are variables and partial fluents are the known ones, except for φ which is as follows:

$$
\varphi_1(s,\tau,t) = \begin{cases} unknown & \text{if } \tau \in (s,t) \\ 1.0 & \text{if } \tau = t \end{cases}
$$

$$
\varphi_2(s,\tau,t) = \begin{cases} unknown & \text{if } \tau \in (s,t) \\ 0.0 & \text{if } \tau = t \end{cases}
$$

Let complete the chronicle.

Proof. Is $History(\tau, l) = \{1.0, 0.0\}$, in fact:

$$
\begin{array}{rlrl}
\mathcal{C}(\tau,l) & = \varnothing & & \\
LI(\tau,l) & = \varnothing & & \\
\mathcal{P}(\tau,l) & = \{\langle \mathbf{s}_1, \mathbf{t}_1, Load\rangle, \langle \mathbf{s}_2, \mathbf{t}_2, Spin\rangle\} & \mathcal{F}(\tau,l) & = \{\langle \mathbf{s}_3, \mathbf{t}_3, Fire\rangle\} \\
LP(\tau,l) & = \{\langle \mathbf{t}_2, l, 1.0\rangle, \langle \mathbf{t}_2, l, 0.0\rangle\} & UP(\tau,l) & = \{\langle \mathbf{s}_3, l, unknown\rangle\} \\
LS(\tau,l) & = \varnothing & US(\tau,l) & = \varnothing \\
glo(\tau,l) & = \{\langle \mathbf{t}_2, l, 1.0\rangle, \langle \mathbf{t}_2, l, 0.0\rangle\} & luo(\tau,l) & = \{\langle \mathbf{s}_3, l, unknown\rangle\}
\end{array}
$$

Proof. Is $History(\mu, l) = \{0.0\}$, in fact:

$$
\begin{array}{rlrl}
\mathcal{C}(\mu,l) & = \varnothing & & \\
LI(\mu,l) & = \varnothing & & \\
\mathcal{P}(\mu,l) & = \{\langle \mathbf{s}_2, \mathbf{t}_2, Spin\rangle \langle \mathbf{s}_3, \mathbf{t}_3, Fire\rangle\} & \mathcal{F}(\mu,l) & = \varnothing \\
LP(\mu,l) & = \{\langle \mathbf{t}_2, l, 0.0\rangle, \langle \mathbf{t}_3, l, 0.0\rangle\} & UP(\mu,l) & = \varnothing \\
LS(\mu,l) & = \varnothing & US(\mu,l) & = \varnothing \\
glo(\mu,l) & = \{\langle \mathbf{t}_2, l, 0.0\rangle, \langle \mathbf{t}_3, l, 0.0\rangle\} & luo(\mu,l) & = \{\langle \mu, l, unknown\rangle\}.
\end{array}
$$

Proof. Is $History(\mu, a) = \{1.0, 0.0\}$, in fact:

$$
\begin{array}{rlrl}
\mathcal{C}(\mu,a) & = \varnothing & & \\
LI(\mu,a) & = \varnothing & & \\
\mathcal{P}(\mu,a) & = \{\langle \mathbf{s}_3, \mathbf{t}_3, Fire\rangle\} & \mathcal{F}(\mu,a) & = \varnothing \\
LP(\mu,a) & = \{\langle \mathbf{s}_0, a, 1.0\rangle, \langle \mathbf{t}_3, a, 0.0\rangle\} & UP(\mu,a) & = \varnothing \\
LS(\mu,a) & = \varnothing & US(\mu,a) & = \varnothing \\
glo(\mu,a) & = \{\langle \mathbf{s}_0, a, 1.0\rangle, \langle \mathbf{t}_3, a, 0.0\rangle\} & luo(\mu,a) & = \{\langle \mu, a, unknown\rangle\}.
\end{array}
$$

The reader may verify as follows. If we further assume $\langle \nu, a, 0.0\rangle \in OBS$ and $\mathbf{t}_3 \sqsubseteq \mu \sqsubseteq \nu$, then is $State(\tau) = \{\langle l, 1.0\rangle, \langle a, 1.0\rangle\}$ and $State(\mu) = \{\langle l, 0.0\rangle, \langle a, 0.0\rangle\}$. If we assume instead $\langle \nu, a, 1.0\rangle \in OBS$, then is $State(\tau) = State(\mu) = \{\langle l, 0.0\rangle, \langle a, 1.0\rangle\}$.

6.4 The Stanford murder mystery

The Stanford Murder Mystery is a *pre-post-diction* problem.

Scenario 6.4.1 *A turkey is initially alive, a shooting event takes place and a dead turkey is found. What was true initially and what happened ? (A. Baker [Bak89])*

Let a for "alive" and l for "loaded" be real-valued feature symbols, and let $Load$ and $Fire$ be the action symbols. In the language of *F&F* the problem domain has the following description in $\mathcal{K}$-$\mathbf{RA}d$:

OBS:	$[\mathbf{s}_0]a\hat{=}1.0$
	$[\mathbf{s}_0]a\hat{=}0.0$
	$[\Box] \sqsubseteq (\mathbf{t}, \mathbf{s}_0)\hat{=}true$
SCD:	$[\mathbf{s}, \mathbf{t}]Fire$
LAW:	$[s,t]Fire \Rrightarrow \forall\, \tau \in [s,t]\,.\,[\tau]l\check{=}to\_fire(s,\tau,t)\wedge[\tau]a\check{=}to\_die(s,\tau,t)$

where s, t and τ are variables, while partial fluents are the known ones. We expect to observe $\langle \mathbf{s}, l, 1.0\rangle$ (pure post-diction problem) and $\langle \mathbf{t}, l, 0.0\rangle$ (pre-post-diction problem).

Proof. Is $History(\mathbf{s}, l) = \{1.0\}$, in fact:

$$
\begin{array}{rlrl}
\mathcal{C}(\mathbf{s},l) & = \varnothing & & \\
LI(\mathbf{s},l) & = \varnothing & & \\
\mathcal{P}(\mathbf{s},l) & = \varnothing & \mathcal{F}(\mathbf{s},l) & = \{\langle \mathbf{s},\mathbf{t},Fire\rangle\} \\
LP(\mathbf{s},l) & = \varnothing & UP(\mathbf{s},l) & = \{\langle \mathbf{s},l,1.0\rangle\} \\
LS(\mathbf{s},l) & = \varnothing & US(\mathbf{s},l) & = \varnothing \\
glo(\mathbf{s},l) & = \{\langle \mathbf{s},l,unknown\rangle\} & luo(\mathbf{s},l) & = \{\langle \mathbf{s},l,1.0\rangle\}.
\end{array}
$$

Let show that $Sat\_Post(\langle \mathbf{s}, \mathbf{t}, Fire\rangle)$ hold. Is $\langle \mathbf{s}, \mathbf{t}, Fire\rangle \in SCD$, $\mathbf{s} \sqsubseteq \mathbf{s} \sqsubseteq \mathbf{t}$, $Postconditions(\mathbf{s}, Fire) = \{\langle \mathbf{t}, l, 0.0\rangle, \langle \mathbf{t}, a, 0.0\rangle\}$, $luo(\mathbf{t}, l) = unknown$ and $luo(\mathbf{t}, a) = \{\langle \mathbf{s}_0, a, 0.0\rangle\}$. We remind that "*unknown*" is an abbreviation for $\bigvee_i v_i$, varying i over all possible values v_i in $\mathcal{V}$, then is $\langle \_, l, 0.0\rangle \in luo(\mathbf{t}, l)$. □

Proof. Is $History(\mathbf{t}, l) = \{0.0\}$, in fact:

$$
\begin{array}{rlrl}
\mathcal{C}(\mathbf{t},l) & = \{\langle \mathbf{s},\mathbf{t},Fire\rangle\} & & \\
LI(\mathbf{t},l) & = \{\langle \mathbf{t},l,0.0\rangle\} & & \\
\mathcal{P}(\mathbf{t},l) & = \varnothing & \mathcal{F}(\mathbf{t},l) & = \varnothing \\
LP(\mathbf{t},l) & = \varnothing & UP(\mathbf{t},l) & = \varnothing \\
LS(\mathbf{t},l) & = \varnothing & US(\mathbf{t},l) & = \varnothing \\
glo(\mathbf{t},l) & = \{\langle \mathbf{t},l,0.0\rangle\} & luo(\mathbf{t},l) & = \{\langle \mathbf{t},l,unknown\rangle\}.
\end{array}
$$

Let show that $Sat\_Pre(\langle \mathbf{s}, \mathbf{t}, Fire\rangle)$ hold. Is $\langle \mathbf{s}, \mathbf{t}, Fire\rangle \in SCD$, $\mathbf{s} \sqsubseteq \mathbf{t} \sqsubseteq \mathbf{t}$, $\langle \_, l, 0.0\rangle \in Postconditions(\mathbf{t}, Fire)$ and $luo(\mathbf{s}, l) = \{\langle \mathbf{s}, l, 1.0\rangle\}$, therefore is $\langle \_, l, 1.0\rangle \in luo(\mathbf{s}, l)$. □

In literature, the Stanford Murder Mystery always appeared as a pure post-diction problem (also in [San94a, Doh94a]), while the problem of determining whether the gun is not loaded after the shooting is a genuine pre-post-diction problem. In fact, abductive consequence of the observed dead turkey is the gun being initially loaded, that is $luo(\mathbf{s}, l) = \{\langle \mathbf{s}, l, 1.0\rangle\}$, so that the firing action satisfies all the necessary conditions for being successfully executed (note that $glo(\mathbf{s}, l) = \{\langle \mathbf{s}, l, unknown\rangle\}$), then the gun becomes unloaded (new information from $glo(\mathbf{t}, l)$) and the turkey dies (explanation of an observed fact from $glo(\mathbf{t}, a)$). If the turkey were not dead at $\mathbf{s}_0$, then the firing action could not be successfully executed, and for all τ and in every intended model, the turkey is alive and the gun, due to the backward persistence, is unloaded.

6.5 Post-diction and abduction

The reader may easily experiment that abduction does not solve all possible post-diction problems. Post-diction, in general, is *not* abduction.

Example 6.5.1 *Suppose a scenario description where* $OBS = \{\langle 9, f, true\rangle\}$*, while* SCD *and* LAW *are empty. The query* $\langle 5, f, ?\rangle$ *is a post-diction problem that abduction does not solve because of the absence of successfully executed actions involving the feature* f*. The intended answer is obviously* $History(5, f) = \{true\}$*, in fact is* $glo(5, f) = \{\langle 5, f, unknown\rangle\}$ *and* $luo(5, f) = \{\langle 9, f, true\rangle\}$.

Example 6.5.2 *Suppose a scenario description where* $OBS = \{\langle 8, l, 1.0\rangle, \langle 13, a, 0.0\rangle\}$*, the schedule is* $SCD = \{\langle 9, 12, Fire\rangle\}$*, and* LAW *contains the action law for* $Fire$*. The query* $\langle 5, a, ?\rangle$ *is a problem that abduction does not solve, since no actions are executed where the required feature hold as a precondition. The intended answer is obviously* $History(5, a) = \mathcal{V}$*, in fact is* $glo(5, a) = luo(5, a) = \{\langle 5, a, unknown\rangle\}$.

Chapter 7

The calculus

A logic program is a set of axioms, or rules, defining relationships between objects. A computation of a logic program is a deduction of consequences of the program. A program defines a set of consequences, which is its meaning. The art of logic programming is constructing concise and elegant programs that have the desired meaning.

Through solving the exercises, the reader may find, however, that there is a difference between recognizing and constructing elegant logic programs.

L. Sterling and E. Shapiro – The Art of Prolog.

In this chapter we define a preferential logic for reasoning about actions and change. The logic, in the genuine spirit of the systematic approach, will be assessed and compared. Exploiting the non-simulative algebraic semantics defined in chapter 5, we show that much of the work concerning the design of preferential logics for Features and Fluents is a complete work, so that their definition and itself implementation results to be a well driven task. The proposed method includes the design of an automatic theorem prover.

7.1 Preliminaries

7.1.1 Logic Programming

Let recall the basic notions on (positive) logic programming. Any concept not formally defined refers to classical papers from the literature. Let L be the Horn clause subset of the First Order Predicate Calculus, and let P be a positive logic program (a finite set of definite Horn clauses). The *Herbrand universe* U_L for L is the set of all ground terms formed by using constants and function symbols in L, while the *Herbrand base* B_L for L is the set of all ground atoms formed by using predicate symbols from L with ground terms from U_L as arguments. An *Herbrand interpretation*, since the assignment to constant and function symbols is fixed, is usually identified with a subset of the Herbrand base. Herbrand models enjoy the *model intersection property*, for which the intersection of a finite (non zero) number of Herbrand models for a sentence P is still a model. Since every P in L has B_P as an Herbrand model, the set of all Herbrand models for P is not empty, thus the intersection of all the Herbrand models for P is again a model, called the *least Herbrand model* M_P for P. Although intended interpretations for P can differ from M_P, since Van Emden and Kowalski [EK76] proved that M_P is the set of all atoms A in B_P such that A is a logical consequence of P, then the least Herbrand model has been adopted as the standard interpretation for a whatever P in L. A deeper characterization of M_P has been obtained using the partial order relation of set inclusion and fix-point theory [Kow74]. Let be 2^{B_P} the set of all Herbrand interpretations of P, that is a complete lattice under the partial order of set inclusion $\subseteq$. The top element of this lattice is B_P and the bottom element is $\varnothing$. An continuous operator $T_P: 2^{B_P} \to 2^{B_P}$ has been defined as follows:

$$\begin{aligned} T_P(I) = \{A \in B_P :& \\ & A \leftarrow A_1 \wedge \ldots \wedge A_n \text{ is a ground instance} \\ & \text{of a clause in } P \text{ and } \{A_1, \ldots, A_n\} \subseteq I\} \end{aligned}$$

The least Herbrand model has then been characterized in terms of the ordinal powers of T_P, by means of the Knaster-Tarski fix-point theorem and the Kleene theorem on the limit ordinal of continuous operators over complete lattices:

$$M_P = lfp(T_P) = glb\{I : T_P(I) \subseteq I\} = T_P \uparrow \omega$$

where $lfp(T_P)$ is the least fix-point of T_P, $glb\{I : T_P(I) \subseteq I\}$ is the greatest lower bound of pre-fix-points for T_P, and $T_P \uparrow \omega$ is the ordinal (upward) power of T_P with ω first infinite ordinal. The semantics of a logic program P describes the *intended meaning* of the program by specifying its logical consequences, that is for all $p(t_1, \ldots, t_n)$ in B_L

$$P \vDash p(t_1, \ldots, t_n) \;\Leftrightarrow\; p(t_1, \ldots, t_n) \in T_P \uparrow \omega$$

The Horn clause logic is also a correct and complete logic [EK76]. We finally recall the least Herbrand model for P is not a compositional model-theoretic semantics, in fact for all sentences P_1 and P_2 it does not hold that the least Herbrand model for $P_1 \cup P_2$ equals the (semantical) composition of M_{P_1} and M_{P_2}, while the T_P operator does it.

7.1.2 Meta Programming

The conceptual basis of logic programming is deduction from a single fixed theory. Deduction from varying or alternative theories is allowed by means of meta-interpretation. To fix the ideas, we give two examples. The following is the famous *vanilla* meta-circular interpreter for positive logic programming [SB86, SS86b, SM92], where deduction from the single fixed theory P is replaced by deduction from a possibly varying single fixed theory.

1. $demo(true) \leftarrow$
2. $demo(X\&Y) \leftarrow demo(X) \wedge demo(Y)$
3. $demo(X) \leftarrow clause(X, Body) \wedge demo(Body)$

The predicate *clause* is the meta-level representation of the object program P, using the non-ground representation for object-level variables, and consists in facts of the form $clause(A, A_1\& \ldots \&A_n) \leftarrow$ for every clause $A \leftarrow A_1, \ldots, A_n$ $(n \geq 1)$ in P, and facts of the form $clause(A, true) \leftarrow$ for every fact $A \leftarrow$ in P.

The predicate *demo* is the Horn clause representation of the entailment relation for definite logic programs, that is [LR93]:

$$p(t_1,\ldots,t_n) \in T_P \uparrow \omega \;\;\Leftrightarrow\;\; demo(p(t_1,\ldots,t_n)) \in T_{V_P} \uparrow \omega$$

where V_P is the program *demo* augmented with the definition of *clause*, and T is the fix-point S-semantics. Exists a variant, namely the *multi-vanilla* meta-circular interpreter for positive logic programming [Ste84, Ste85], where deduction from the single fixed theory P is replaced by deduction from possibly varying alternative theories:

1. $demo(true, Theory) \leftarrow$
2. $demo((X\&Y), Theory) \leftarrow demo(X, Theory) \wedge demo(Y, Theory)$
3. $demo(X, Theory) \leftarrow clause\_t(X, Body, Theory) \wedge demo(Body, Theory)$

The predicate *clause_t* is the meta-level representation of the object programs P_i, using the non-ground representation for object-level variables, and consists in facts of the form $clause\_t(A, A_1\&\ldots\&A_n, t_i) \leftarrow$ for every clause $A \leftarrow A_1,\ldots,A_n$ $(n \geq 1)$ in P_i, and facts of the form $clause\_t(A, true, t_i) \leftarrow$ for every fact $A \leftarrow$ in P_i.

Meta-programming, as clear from the above examples, is the treatment of programs as data. In spite of its wide use in logic programming, meta-programming can not be considered as being an its own feature or technique. The fundamental property is, in fact, the equivalence of programs and data. This property was initially satisfied by the von-Neumann model, but usually lost in the development of formal programming languages. For those languages which allow the equivalence of programs and data, meta-interpretation is just the technique for exploiting such property, and simply consists in defining programs (meta-programs) that treat other programs (object-programs) as data. Meta-programs may also be programs which instructions treat themselves as data. Interpretation, partial evaluation [Tak86], compilation and debugging, are instances of meta-programming. Within logic programming, the first investigations are due to Bowen and Kowalski [BK82] on the amalgamation of language and meta-language. Interesting papers are also [Esh86, Sub88, HL88, HL89]. The well known drawback of the relatively poor efficiency of meta-programs, when compared to procedural languages, was recovered by partial evaluation techniques, as in [SS86a].

Exists literature which relates meta-programming, knowledge representation and common sense reasoning. A first result in this direction is still due to Bowen and Kowalski (op. cit.), where amalgamation of language and metalanguage is shown to be an alternative to the non-monotonic logics of McDermott and Doyle [MD80] and the default logic of Reiter [Rei80]. Far from being complete, we also cite Bowen [Bow85] on the relation between meta-programming and knowledge representation, some works of Sterling on expert systems [Ste84, Ste85], and a more recent work on belief revision [Gär92].

One of the aims and main challenge of meta-programming in logic programming, as stated by Bowen, is to provide powerful (meta) programming languages suitable for artificial intelligence applications, apart from the additional problem of reconciling the extended semantics with the fully logical semantics of logic programming. The methodology of research inherited from the autonomous field of non-monotonic logics for common sense reasoning, which is the formal approach to Artificial Intelligence, was the example based research methodology.

In this chapter, we give a novel contribution in this respect, adopting the Features and Fluents systematic methodology.

7.2 The base logic

As defined by Shoham [Sho89], preferential logics are the result of associating with a standard logic a preference relation on models. A standard logic is a logic with the usual compositional model-theoretic semantics. The Horn Clause Logic provably is one of those logics of interest for the Formal Artificial Intelligence:

Theorem 7.2.1 *The Horn Clause Logic is a Shoham's preferential logic.*
PROOF. The Horn Clause Logic is a standard logic, in fact it has a compositional model-theoretic semantics (Kowalski's T_P operator). Let $\sqsubset$ be the partial order on interpretations per $\mathcal{L}$ defined as follows: for all I_1 and I_2 Herbrand interpretations, $I_1 \sqsubset I_2$ iff $I_2 \subset I_1$ (we neglect equivalent models). According to def. 2.2.1 the Horn Clause Logic $\mathcal{L}_{\sqsubset}$ is a Shoham's preferential logic. □

The pure paradigm allows the representation of temporal expressions by means of ordinary atoms, which semantics is definable by proper logic programs. In the general lines of chapter 3, all functions and predicates of the temporal signature Σ shall have their explicit definition; for example the relations $\sqsubset$ and $\sqsubseteq$, and their converse relations $\sqsupset$ and $\sqsupseteq$, shall be regarded as new predicate symbols which intended interpretation is the respective relation on members of the chosen temporal domain. Concerning partial fluents, which are ternary functions, they shall be converted to quaternary predicates where the additional parameter shall ideally consists in the result of the partial fluent evaluation. Depending on the chosen temporal structure, temporal expressions and partial fluents may involve complex operators, and their extra layer of interpretation will quickly converge towards a genuine *constraint solver*. Therefore a more appropriate compositional model-theoretic semantics for our base logic is Jaffar's T_P operator for the *Constraint Logic Programming* scheme [JL87, JM94], which merges the above declarative paradigm of Logic Programming with a declarative paradigm of constraint solving (dependent on the application domain), with the additional advantage of having computationally efficient, specialized, and typed arithmetics. The T_P operator is as follows:

$$
\begin{array}{l}
T_P(I) = \{d \in \mathcal{D}\text{-base}: \\
\quad \text{exists a clause } A \leftarrow C_1 \wedge \ldots \wedge C_m \wedge A_1 \wedge \ldots \wedge A_n \text{ in } P \\
\quad \text{and exists a } \mathcal{D}\text{-valuation } \theta \text{ such that} \\
\quad \text{(a) } \mathcal{D} \models (A\theta = d), \\
\quad \text{(b) } \mathcal{D} \models C_1\theta \wedge \ldots \wedge C_m\theta, \text{ and} \\
\quad \text{(c) } \{A_1\theta, \ldots, A_n\theta\} \subseteq I\}
\end{array}
$$

The unification algorithm, defined over the Herbrand universe, is replaced with a constraint solver, defined over the mathematical structure $\mathcal{D}$. The CLP scheme depends on a choice $\mathcal{X} = \langle \Sigma, \mathcal{D}, \mathcal{L}, \mathcal{A} \rangle$ of the application domain and its constraints, where Σ is a signature, $\mathcal{D}$ is a Σ-structure, $\mathcal{L}$ is a class of Σ-formulas (the class of constraint which can be expressed) and $\mathcal{A}$ is a first-order

Σ-theory (an axiomatization of properties of $\mathcal{D}$). For each choice of $\mathcal{X}$, the logic $CLP(\mathcal{X})$ is defined. We remind that in the special case where Σ is the set of all constants and functions of a positive logic program P together with the binary predicate symbol $=$, the domain is the Herbrand universe and constraints are equations between terms, Constraint Logic Programming and Logic Programming coincide. For our purposes, the choice of $\mathcal{X}$ will depend on the ontological assumptions: the signature Σ and the structure $\mathcal{D}$ will be a temporal signature (master clock function excluded) and a temporal structure in the general lines of chapter 3. The best structure $\mathcal{D}$ would be a composite temporal-spatial structure but, for simplicity, we can assume the spatial structure as being isomorphic to the temporal one (like in the "Achilles and the tortoise" example). Therefore, the $CLP(\mathcal{R})$ logic [JMSY92], with linear real-arithmetic constraints, can be a base logic for $\mathcal{K}$-**R** super-classes. We remind that, in general, the temporal structure may be non-linear (branching time case).

7.3 The model preference

What is the range of applicability of the base logic within the Features and Fluents framework ? The question is, properly, whether the standard preference relation on Herbrand models is adequate for our purposes.

Theorem 7.3.1 *The Horn Clause Logic is a monotonic preferential logic.*
PROOF. According to def. 2.2.7, HCL is a Shoham's monotonic preferential logic, in fact for all A and B set of definite clauses and C set of unit clauses, if $C \subset T_A \uparrow \omega$ then also $C \subset T_{A \cup B} \uparrow \omega$. $\square$

Suppose to rewrite problem domains in $\mathcal{K}$-**RAC***i* to Horn theories, via a certain mapping $\star$. If Υ is a problem domain, we represent with $\Upsilon^\star$ its associated Horn theory. The least fix-point of T_P, where $P = \Upsilon^\star$, is a single model which fails to include much of the intended model set, and we may also obtain an empty model even when models may be intended. In fact, as shown with the non-simulative algebraic semantics, deductive-like and abductive-like reasoning, together with the temporal persistence, determine the overall range of applicability. In particular what emerges from the algebraic semantics is the *dependence* of the deductive-like reasoning from the abductive-like reasoning itself (successful preconditions are determined in terms of both the greatest lower and least upper observations), while the T_P operator just defines the fix-point semantics of a purely deductive atemporal reasoning. Therefore we need an operation of recovery of the model set, in order to obtain the intended one. This operation may be superimposed on the base logic, that is instead of define the semantics of the logic from the very beginning, like a better T_P operator, we just define an extension of the base logic in meta-theoretic terms, that is the semantics is expressed in terms of theories instead of models, and the underlying [1] semantics remains that of the base logic. This is a conservative approach where the extension of the base logic is defined in terms of meta-theoretic relationships between object level theories; in our case, reasoning (pure deduction) from a single fixed theory (one set P of premises) is replaced by reasoning from several

[1] The term *underlying* is here meant as "lays under", and must not be confused with "intended".

theories (OBS, SCD and LAW), where each theory is allowed to grow passing time. We extend the original object language of the base logic including that portion of the metalanguage which deals with the new entailment relation. The resulting system will be shown to have greater problem-solving power than the original object language system alone, and we will refer to it as *The Calculus of Fluents*. Adopting the entailment relation that is associated with the algebraic semantics, we define a meta-theoretic extension of the base logic using the non ground representation of the object level variables. The meta-interpreter consists in the Horn clause representation of that relation, that is for all scenario description Υ in $\mathcal{K}$-**RAC**i and for all observation $\langle t, f, v\rangle$ in $\mathcal{H}$, the predicate symbol *demo*, in the context Δ, represents the relation *Comp* if and only if

$$\langle t, f, v\rangle \in Comp(\Upsilon) \quad \Leftrightarrow \quad demo(t, f, v) \in T_{\Delta \cup \Gamma} \uparrow \omega$$

where Δ is the Horn clause representation of the new entailment relation (the algebraic semantics) using the non-ground representation for object-level variables, and Γ is the meta-level representation of the Horn theory $\Upsilon^{\star}$.

7.3.1 The meta-interpretive definition

The following enhanced meta-interpreter consists in the Horn clause representation of the algebraic semantics we defined so far.

Influence:

A1. $influence(Tau, (A, B, Body), F) \leftarrow$
$\quad A < Tau, Tau \leq B,$
$\quad member\_iaa(holds(B, F, \_), Body),$
$\quad sat\_pre((A, B, Body))$

Current Action Set:

A2. $cas(Tau, F, I) \leftarrow setof((A, B, Body), ($
$\quad clause\_t(do(S, T, Action), true, scd),$
$\quad solve(S, A), solve(T, B),$
$\quad A < Tau, Tau \leq B,$
$\quad clause\_t(do(A, B, Action), Body, law),$
$\quad influence(Tau, (A, B, Body), F)), I)$

Past Action Set:

A3. $pas(Tau, F, I) \leftarrow$
$\quad l(Tau, F, L), max(L, I1),$
$\quad setof((A1, B1, Body1), ($
$\qquad member((A1, B1, Body1), L),$
$\qquad member((A2, \_, Body2), L),$
$\qquad B1 \leq A2,$
$\qquad forall(member\_iaa(holds(B1, F, V), Body1),$
$\qquad\quad not\_member\_iaa(holds(A2, F, V), Body2))), I2),$
$\quad union(I1, I2, I)$

A4. $l(Tau, F, L) \leftarrow setof((A, B, Body), ($
$\quad clause\_t(do(S, T, Action), true, scd),$
$\quad solve(S, A), solve(T, B), B < Tau,$
$\quad clause\_t(do(A, B, Action), Body, law),$
$\quad influence(B, (A, B, Body), F)), L)$

Future Action Set:

A5. $fas(Tau, F, I) \leftarrow$
$u(Tau, F, U), min(U, I1),$
$setof((A2, B2, Body2), ($
$member((A1, B1, Body1), U),$
$member((A2, B2, Body2), U),$
$B1 \leq A2,$
$forall(member\_iaa(holds(B2, F, V), Body2),$
$not\_member\_iaa(holds(A1, F, V), Body1))), I2),$
$union(I1, I2, I)$

A6. $u(Tau, F, U) \leftarrow setof((A, B, Body), ($
$clause\_t(do(S, T, Action), true, scd),$
$solve(S, A), solve(T, B), Tau \leq A,$
$clause\_t(do(A, B, Action), Body, law),$
$member\_iaa(holds(A, F, \_), Body), sat\_post((A, B, Body))), U)$

Greatest Lower Observations:

A7. $glo(Tau, F, Obs) \leftarrow$
$li(Tau, F, LI), lp(Tau, F, LP), ls(Tau, F, LS),$
$((LI = [\,], LP = [\,]) \rightarrow Obs = [(0, F, unknown)]; merge(LP, LS, LI, Obs)).$

A8. $lp(Tau, F, Obs) \leftarrow$
$pas(Tau, F, P),$
$(P = [\,] \rightarrow (setof((B, F, V), ($
$clause\_t(holds(B, F, V), true, obs),$
$B \leq Tau), H),$
$max(H, Obs));$
$setof((B, F, V), ($
$member((\_, B, Body), P),$
$member\_iaa(holds(B, F, V), Body)), Obs))$

A9. $ls(Tau, F, Obs) \leftarrow$
$cas(Tau, F, C),$
$lp(Tau, F, LP),$
$setof((\_, F, V), ($
$member((\_, F, V), LP),$
$member((A, \_, Body), C),$
$member\_iaa(holds(A, F, V), Body)), Obs)$

A10. $li(Tau, F, Obs) \leftarrow$
$cas(Tau, F, C),$
$setof((Tau, F, V), ($
$member((\_, \_, Body), C),$
$member\_iaa(holds(Tau, F, V), Body)), Obs)$

Least Upper Observations:

A11. $luo(Tau, F, Obs) \leftarrow$
$up(Tau, F, UP),$
$us(Tau, F, US),$
$merge(UP, US, [\,], Obs)$

A12. $up(Tau, F, Obs) \leftarrow$
$\quad fas(Tau, F, FAS),$
$\quad (FAS = [\,] \rightarrow (setof((B, F, V), ($
$\qquad\qquad clause\_t(holds(B, F, V), true, obs),$
$\qquad\qquad Tau \leq B), H),$
$\qquad\quad min(H, Obs));$
$\qquad\quad setof((B, F, V), ($
$\qquad\qquad member((A, \_, Body), FAS),$
$\qquad\qquad member\_iaa(holds(A, F, V), Body)), Obs))$

A13. $us(Tau, F, Obs) \leftarrow$
$\quad cas(Tau, F, CAS),$
$\quad up(Tau, F, UP),$
$\quad setof((\_, F, V), ($
$\qquad member((\_, F, V), UP),$
$\qquad member((A, \_, Body), CAS),$
$\qquad member\_iaa(holds(A, F, V), Body)), Obs)$

The "sat" predicates:

A14. $sat\_obs(F, (S, T, Body)) \leftarrow$
$\quad forall(clause\_t(holds(Tau, F, V), true, obs),$
$\qquad S \leq Tau, Tau \leq T),$
$\qquad member\_iaa(holds(Tau, F, V), Body))$

A15. $sat\_pre((S, T, Body)) \leftarrow$
$\quad forall(member\_iaa(holds(S, F, V), Body),$
$\qquad (glo(S, F, V); luo(T, F, V))),$
$\quad forall(member\_iaa(holds(T, G, \_), Body),$
$\qquad sat\_obs(G, (S, T, Body))))$

A16. $sat\_post((S, T, Body)) \leftarrow$
$\quad forall(member\_iaa(holds(S, F, V), Body),$
$\qquad (luo(T, F, V), sat\_obs(F, (S, T, Body))))$

The predicate "demo":

A17. $history(T, F, V) \leftarrow$
$\quad glo(T, F, GLO),$
$\quad luo(T, F, LUO),$
$\quad consistent\_union(GLO, LUO, V)$

A0. $demo(TimePoint, Feature, Values) \leftarrow$
$\quad history(TimePoint, Feature, Values)$

The kernel of the completion set:

A18. $kstate(Tau, Obs) \leftarrow$
$\quad features(FF),$
$\quad setof((F, V), (member(F, FF), history(Tau, F, V)), Obs)$

A19. $kernel(Obs) \leftarrow$
$\quad breakpoints(B),$
$\quad setof((T, F, V), (member(T, B), kstate(T, S), member((F, V), S)), Obs)$

We assume for granted a number of auxiliary predicates, such as union, intersection, subtraction, membership and all the common predicates for manipulating sets. The $setof$ auxiliary predicate shall be regarded as a new predicate symbol $setof(x, property(x), S)$ whose intended meaning is $S = \{x \mid property(x)\}$.

The intended meaning of the consistent union axiom is clear from chapter 5, and we give for granted its simple definition, which is a variant of the union predicate. The $merge(A, B, C, D)$ auxiliary predicate shall be regarded as a new predicate symbol whose intended meaning is $D = (A \setminus B) \cup C$. We also use the $forall(x, property(x))$ and $exists(x, property(x))$ auxiliary axioms, which shall be regarded as new predicate symbols with their conventional meaning.

The object-level representation of the problem domain

The algebraic semantics suggests a certain representation of scenario descriptions, as clear from the *clause_t* occurrences. Assuming global features and unique names for features and actions, the representation is as follows:

(1) for every observation $[t]f\hat{=}v$ in the OBS part of Υ, there is an atomic formula $holds(t, f, v) \leftarrow$ in $\Upsilon^\star$, where $t \in \mathcal{T}$, $f \in \mathcal{F}$, and $v \in \mathcal{V}$;

(2) for every action occurrence $[\mathbf{s}, \mathbf{t}]A$ in the SCD part of Υ, there is an atomic formula $do(s, t, a) \leftarrow s\ is\ \mathbf{s}, t\ is\ \mathbf{t}$ in $\Upsilon^\star$, where $\mathbf{s}$ and $\mathbf{t}$ are temporal expressions, s and t are variables, $A \in \mathcal{E}$ and a is the syntactical counterpart of A.

(3) for every action law $[s, t]A \Rrightarrow \bigvee_{i=1}^{n}(\bigwedge_{j=1}^{m} S_{ij})$ in the LAW part of Υ, there is a clause $do(s, t, A) \leftarrow \bigvee_{i=1}^{n}(\bigwedge_{j=1}^{m} S_{ij}^\star)$ in $\Upsilon^\star$, where s and t are variables. The disjunction in the body is the standard shorthand for the finite database

$$\begin{array}{l} do(s, t, A) \leftarrow \bigwedge_{j=1}^{m} S_{1j}^\star \\ do(s, t, A) \leftarrow \bigwedge_{j=1}^{m} S_{2j}^\star \\ \qquad\vdots \\ do(s, t, A) \leftarrow \bigwedge_{j=1}^{m} S_{nj}^\star \end{array}$$

where each element represents an action alternative. The crucial point of the overall transformation is $S_{ij}^\star$. The trajectory formula S_{ij}, that is

$$\forall\, \tau \in [s, t]\,.\,[\tau]f_{ij} \check{=} \varphi_{ij}(s, \tau, t)$$

is translated to

$$\leftarrow s \leq \tau \leq t, v_{ij} = \varphi_{ij}(s, \tau, t), holds(\tau, f_{ij}, v_{ij})$$

As clear from the algebraic semantics and the itself meta-interpreter, the constraint $s \leq \tau \leq t$ is superfluous. According to this representation, the standard [Kow74] procedural and declarative reading for clauses should not be adopted for clauses representing action laws, in fact their head is required to be completely instantiated in order to compute the answers from the body. The *intended reading*, which differs from the standard one, is fully recovered by the meta-interpreter. Of course the standard representation is always possible, and involves disjunctive clauses with constraints.

The rewriting of Υ *to* $\Upsilon^\star$ *is a non forgetting and non augmenting transformation.* The set $\Upsilon^\star$ is an axiomatization of the problem domain expressed by Υ.

The meta-level representation of the object-level

The definition of $clause\_t$ is the meta-level representation Γ of the object theory $\Upsilon^\star$ and, as shown, consists in facts of the form

$$clause\_t(A, B_1 \& \ldots \& B_k, Theory) \leftarrow$$

for every clause $A \leftarrow B_1, \ldots, B_k$ $(k \geq 1)$ in $\Upsilon^\star$ and facts of the form

$$clause\_t(A, true, Theory) \leftarrow$$

for every fact $A \leftarrow$ in $\Upsilon^\star$, like in *multi-vanilla* [Ste84, Ste85]. Three theories are here involved: *obs*, *scd* and *law*, each of which refers to the corresponding partition in Υ.

The rewriting of $\Upsilon^\star$ *to* Γ *is a non forgetting and non augmenting transformation.*

On the intended reading of clauses

The granularity [SS86b] of the meta-interpreter allows the visibility of the selection rule and the application of the resolution principle, hence it is "fine" enough for the purposes of an alternative reading. Deduction from the single fixed theory $\Upsilon^\star$ is replaced by deduction from possibly varying alternative theories, those theories specified by Υ. The order in which statements are given by a perception device, or written in the theory, is not relevant; information is retrieved like in a database and no matter whether a certain occurrence is retrieved earlier or later than the others, in fact all matching occurrences are taken into account. The selection rule for the atom in the body consists in the $member\_iaa$ auxiliary axiom, which definition is as follows:

$$\begin{aligned}&member\_iaa(holds(t, f, v), Body) \leftarrow\\&\quad conjunction\_to\_list(Body, List),\\&\quad member(holds(t, f, v), List)\end{aligned}$$

The $conjunction\_to\_list(Body, List)$ predicate converts a clause body $Body$, which is a conjunction of atoms, into a list $List$ of atoms. Per construction, there exists exactly one occurrence of the $holds$ atom for each feature in a given body. Variables in $List$ must be the same variables occurring in $Body$.

On ground scenario descriptions

When dealing with scenario descriptions in $\mathcal{K}$g-**RAC***i* , the following proposition holds:

Definition 7.3.1 (language independence) *A definite program P with underlying language L_P is* language independent *if and only if, for any extension L'_P of L_P, its least L'_P-Herbrand model is equal to its least L_P-Herbrand model. (De Schreye and Martens [SM92])*

Proposition 7.3.1 *For all Υ in $\mathcal{K}g$-***RAC***i* *, $\Upsilon^\star$ is language independent.*
Proof. The theory $\Upsilon^\star$ is a set of definite Horn clauses, and all its clauses are *range restricted*, that is any variable that appears in the head of a clause, also

appears in its body. Since $\Upsilon^\star$ is range restricted, we can apply the result of De Schreye and Martens [SM92] for which *range-restricted definite programs are language-independent.*

Language independence is usually satisfied by deductive data base programs. Language independent programs can compute ground answers only.

7.4 Assessment

If Υ is a scenario description in $\mathcal{K}$-**RAC***i* , then Δ will denote the definition of *demo*, Γ will denote the definition of *clause_t*, and $\Delta \cup \Gamma$ will denote the logic program consisting of the above meta-interpreter and the meta-level representation of the clauses in $\Upsilon^\star$. $\Delta \cup \Gamma$ is designed according to Kowalski's [Kow79],

$$Algorithm = Logic + Control$$

in fact $\Delta \cup \Gamma$ is an algorithm, where Γ is the logic component which "specifies the knowledge to be used in solving problems", and Δ is the control component which "determines the problem-solving strategies by means of which that knowledge is used". The algorithm we defined, is bounded to dynamic meta-level theories and a static meta-interpreter oriented towards the pure control of deduction. With a different language we can say that Γ is a problem domain, $T' = T_\Delta$ is the fix-point operator of a *problem-domain independent* preferential logic, and $T'_\Gamma \uparrow \omega$ is the solution of all possible problems that may be associated with Γ. Although domain independent, the preferential logic has to be *class dependent*, that is Δ shall be adequate for solving problems associated with all possible problem domains Γ in a certain class ($\mathcal{K}$-**RAC***i* in our specific case). While the property of domain independence is given for granted, the class dependence must be formally proved by means of an assessment result, which *certifies* the range of applicability of the specific logic.

We now show the correspondence between the models of $\Delta \cup \Gamma$ and the intended models of Υ, for all Υ in $\mathcal{K}$-**RAC***i* .

Theorem 7.4.1 (soundness and completeness) *For all scenario description Υ in $\mathcal{K}$-**RAC**i is*

$$\langle t, f, v\rangle \in \Sigma_{\mathcal{K}\text{-}\mathbf{RAC}i}\ (\Upsilon) \;\Leftrightarrow\; demo(t, f, v) \in T_{\Delta\cup\Gamma} \uparrow \omega$$

PROOF.

$$\begin{array}{lll} \langle t, f, v\rangle \in \Sigma_{\mathcal{K}\text{-}\mathbf{RAC}i}\ (\Upsilon) & \Leftrightarrow & \text{(theorem 5.4.1)} \\ \langle t, f, v\rangle \in Comp(\Upsilon) & \Leftrightarrow & \text{(per construction}^1) \\ demo(t, f, v) \in T_{\Delta\cup\Gamma} \uparrow \omega & & \end{array}$$

1. *demo* is the Horn clause representation of the algebraic semantics. □

7.5 Comparisons

According to Sandewall's systematic approach, we now compare the Fluent Calculus with some of the most interesting approaches in literature. The comparisons are made by means of set inclusion with respect to the range of applicability of the compared methods.

7.5.1 FC and the $\mathcal{A}$ Language compared

In a recent work, M. Thielscher [Thi94] provided an assessing analysis of the *Action Description Language* $\mathcal{A}$ of Gelfond and Lifschitz [GL92, GL93], for which the range of applicability of the $\mathcal{A}$ language results to be the $\mathcal{K}$-$\mathbf{I}bs\mathbf{A}d$ class of reasoning problems.

The Fluent Calculus, which actual range of applicability is $\mathcal{K}$-$\mathbf{RAC}i$, has a broader range of applicability than the $\mathcal{A}$ language, since $\mathcal{K}$-$\mathbf{I}bs\mathbf{A}d$ is a subclass of $\mathcal{K}$-$\mathbf{RAC}i$.

7.5.2 FC and the Abductive Logic Programming compared

Theorem 7.5.1 *The Abductive Logic Programming with integrity constraints (ALP) is correctly applicable to all chronicles in* $\mathcal{K}$-$\mathbf{I}bs\mathbf{A}d$ *.*
PROOF. ALP is correct and complete with respect to the $\mathcal{A}$ language, as proved by Denecker in [Den93, section 6.3]. Per transitivity on the work of Thielscher (op. cit.), ALP is correctly applicable to all problems in $\mathcal{K}$-$\mathbf{I}bs\mathbf{A}d$. □

The Fluent Calculus, which actual range of applicability is $\mathcal{K}$-$\mathbf{RAC}i$, has a broader range of applicability than ALP (with respect to the Features and Fluents framework), since $\mathcal{K}$-$\mathbf{I}bs\mathbf{A}d$ is a subclass of $\mathcal{K}$-$\mathbf{RAC}i$.

7.5.3 FC and the PMON Logic compared

The *Point-wise Minimization of Occlusion with No-change premises* (PMON) is the best entailment method defined in [San94a]. The method, which uses the occlusion concept, is correct [San94a] for all chronicles in $\mathcal{K}$-$\mathbf{IA}$. The PMON entailment subsumes the followings (see [San94a]):

(1) the *Prototypical Global Minimization* (McCarthy, 1984), which has been assessed for chronicles in $\mathcal{K}r$-$\mathbf{I}s\mathbf{A}z$; (2) the *Original Chronological Minimization* (Kautz, 1986), which has been assessed for chronicles in $\mathcal{K}sp$-$\mathbf{I}s\mathbf{A}d$; (3) the *Prototypical Chronological Minimization*, which has been assessed for chronicles in $\mathcal{K}p$-$\mathbf{IA}ex$; (4) the *Prototypical Chronological Minimization with Filtering*, which has been assessed for chronicles in $\mathcal{K}$-$\mathbf{IA}ex$; (5) the *Global Minimization of Occlusion with No-change Premises*, which has been assessed for chronicles in $\mathcal{K}r$-$\mathbf{I}s\mathbf{A}$, and (6) the *Chronological Minimization of Occlusion and Change*, which has been assessed for chronicles in $\mathcal{K}$-$\mathbf{IA}e$. The PMON entailment is also equivalent to the *Chronological Assignment and Minimization of Occlusion and Change*, which has been assessed for the whole $\mathcal{K}$-$\mathbf{IA}$ class.

Doherty and Łukaszewicz [DL94] gave a (never proved equivalent) circumscriptive formulation of PMON, called "PMON circumscription". Under certain restrictions, Doherty [Doh94b, Lif88] showed the PMON circumscription reducible to a first order formula. Then Doherty [Doh94a] defined a method to compile $\mathcal{K}$-$\mathbf{IA}$ scenario descriptions to "action scenario descriptions": given a scenario description in $\mathcal{K}$-$\mathbf{IA}$, the corresponding action scenario description consists in a set of first-order formulas generated by expanding schedule's elements with involved action laws and including explicit occurrences of the *occlude* predicate. To fix the ideas, this is the action scenario description corresponding to a ground instance of the Yale Shooting Problem:

obs1	$Holds(0, a) \wedge \neg Holds(0, l)$
scd1	$\exists t.2 \leq t < 4 \wedge \forall t'.(t < t' \leq 4 \rightarrow Holds(t', l))$
	$\wedge \forall t'.(2 < t' \leq 4 \rightarrow Occlude(t', l)]$
scd2	$Holds(6, l) \rightarrow$
	$([\exists t.6 \leq t < 8 \wedge \forall t'(t < t' \leq 8 \rightarrow \neg Holds(t', a))]$
	$\wedge \forall t'.(< t' \leq 8 \rightarrow Occlude(t', a)]$
	$\wedge [\exists t.6 \leq t < 8 \wedge \forall t'(t < t' \leq 8 \rightarrow \neg Holds(t', l))]])$
	$\wedge \forall t'.(6 < t' \leq 8 \rightarrow Occlude(t', l)]$

The intended meaning of an action scenario description is obtained by applying the first order PMON circumscription on the action scenario description itself. The PMON Logic is defined as the first order PMON circumscription axiom union the action scenario description of interest. Clearly the PMON Logic is a domain dependent logic.

Due to the range of applicability of Sandewall's PMON entailment, the PMON Logic is currently object of investigations and implementation (see [San95] and [DLS94]).

The Fluent Calculus, which actual range of applicability is $\mathcal{K}$-**RAC**i, has a broader range of applicability than the PMON Logic, since $\mathcal{K}$-**IA** is a subclass of $\mathcal{K}$-**RAC**i. Additional and important advantages of FC with respect to the PMON Logic are the domain independence and the trajectory semantics for continuous change, while the PMON Logic has just an encapsulated semantics for chronicles in linear discrete time and truth valued features.

7.5.4 FC and the Event Calculus compared

The *Event Calculus*, that is a logic based calculus for reasoning about events and time, has been proposed by R. Kowalski in [KS86] together with a number of variants [SK95]. The Horn clause subset of the first order predicate logic is used, augmented with negation-as-failure [Cla78]. The Event Calculus (a simplification of it) is the most popular Logic Programming approach to temporal reasoning, logic-based learning and planning, it has been developed according to the *example based methodology*, that was the common methodology in late 80's, and its range of applicability it has never been assessed. The simplification of interest [Bru95, Mis91] consists in the following two frame axioms:

$$\begin{array}{l} holds\_at(P, T) \leftarrow \\ \quad happens(\_, ET, Ev), ET < T, \\ \quad initiates(Ev, P), succeeds(Ev), \\ \quad not\ clipped(ET, P, T). \end{array}$$

$$\begin{array}{l} clipped(T1, P, T2) \leftarrow \\ \quad happens(BT, ET, Ev), T1 \leq BT, ET < T2, \\ \quad terminates(Ev, P), succeeds(Ev). \end{array}$$

The axiom $holds\_at(P, T)$ is true when the property P holds at time T. The predicate $happens(ST, ET, Ev)$ is true when the event Ev takes place between

BT and ET, points in time. Time-points are compared with the usual order relations. The predicate $initiates(Ev, P)$ is true when the event Ev initiates a period during which property P holds, while $terminates(Ev, P)$ is true when the event Ev terminates any ongoing period during which property P holds. Both *initiates* and *terminates* are defined in terms of the predicate *act*, and check whether Ev influences P. The predicate $succeeds(Ev)$, which definition depends on *act*, *happens* and *holds_at*, checks for preconditions of the event Ev; that predicate is somehow similar to our "*sat*" predicates. The *not* operator is interpreted as negation-as-failure. The use of negation-as-failure in *holds_at* gives a very simple form of temporal persistence. The axiom $clipped(T1, P, T2)$ is true when the property P ceases to hold at some time between $T1$ and $T2$. The problem domain is fully described by a database of *happens*, *act*, *initiates*, *terminates* and *succeeds* predicate definitions. The following is a well known example of problem domain specification in EC.

$initially(alive) \leftarrow$
$happens(0, 0, start) \leftarrow$

$happens(2, 4, e1) \leftarrow$
$happens(6, 8, e2) \leftarrow$
$act(e1, load) \leftarrow$
$act(e2, fire) \leftarrow$

$initiates(start, P) \leftarrow initially(P)$
$initiates(Ev, loaded) \leftarrow act(Ev, load)$
$terminates(Ev, alive) \leftarrow act(Ev, fire)$
$terminates(Ev, loaded) \leftarrow act(Ev, fire)$
$succeeds(start) \leftarrow$
$succeeds(Ev) \leftarrow act(Ev, load)$
$succeeds(Ev) \leftarrow act(Ev, fire), happens(BT, \_, Ev), holds\_at(loaded, BT)$

The specification is close to the Yale shooting scenario; what is missing from being complete is an explicit "$not\ initially(loaded) \leftarrow$", which is not a legal formula in the present formalism. We can see a certain resemblance of the structure of a problem specification in EC with the structure of a scenario description in $\mathcal{K}$-**IA** , in fact the *initially* and *happens* statements for features are like OBS, the *happens* and *act* statements for actions are like SCD, and the *initiates*, *terminates* and *succeeds* statements are like LAW. We can also observe the frame axiom "*clipped*" is like "*occluded*" in def. 2.1.16. Although the Yale problem has a very simple domain, the EC specification requires careful reading, and more complex problems require an even more great deal of patience.

Let call P the program consisting of the frame axioms together with the above problem domain specification. The following correct conclusions hold:

$$P \vdash_{SLDNF} holds\_at(loaded, 6)$$
$$P \vdash_{SLDNF} not\ holds\_at(loaded, 9)$$
$$P \vdash_{SLDNF} not\ holds\_at(alive, 9)$$

A not correct conclusion is $P \vdash_{SLDNF} not\ holds\_at(loaded, 0)$, in fact the underlying semantics for the Yale shooting scenario diminished of the initial observation $[0]loaded \hat{=} true$, allows one more intended conclusion where the gun is not initially loaded. Furthermore: (1) the feature *loaded* is false in $[0,4]$, true in $[5,8]$ and false in $[9,+\infty)$, while it should be unknown ("true or false") in $[0,2]$, unknown at 3, true in $[4,6]$, unknown at 7 and false during $[8,+\infty)$; (2) the feature *alive* is true in $[0,8]$ and false in $[9,+\infty)$, while it should be true in $[0,6]$, unknown at 7 and false in $[8,+\infty)$. Hence the Event Calculus does *not* infer the models that are intended for the Yale Shooting Problem [2]. We remind the Yale Shooting Problem falls within $\mathcal{K}sp$-$\mathbf{I}b\mathbf{A}d$, the simplest meaningful class of the Features and Fluents framework. Therefore:

> *The Event Calculus (at least in the flavor we considered) has no (meaningful) range of applicability within the Features and Fluents framework.*

The difficulty that emerges in comparing the Event Calculus with the Fluent Calculus consists in having a different underlying semantics. However, it is relatively simple to manipulate the Event Calculus according to its present extensions in literature, and obtain an intuitively plausible calculus. In fact, we can initially adjust EC as follows:

$$\begin{array}{l} holds\_at(P,T) \leftarrow \\ \quad happens(\_,ET,Ev), ET \leq T, \\ \quad initiates(Ev,P), succeeds(Ev), \\ \quad not\ clipped(ET,P,T). \end{array}$$

$$\begin{array}{l} clipped(T1,P,T2) \leftarrow \\ \quad happens(BT,ET,Ev), T1 < BT, ET \leq T2, \\ \quad terminates(Ev,P), succeeds(Ev). \end{array}$$

and observe that the modified axioms solve for (almost) all of the above problems, except for those intervals where the features change, in fact: (1) the feature *loaded* is now false in $[0,3]$, true in $[4,7]$ and false in $[8,+\infty)$, while it should be unknown in $[0,3]$ (in fact "$[0]loaded \hat{=} true$" is still missing from the problem domain) and unknown at 7; (2) the feature *alive* is now true in $[0,7]$ and false in $[8,+\infty)$, while it should be unknown at 7.

Some of these remaining problems can be solved via Shanahan's extension of EC to continuous change (see [Sha90]):

$$\begin{array}{l} holds\_at(P,T) \leftarrow \\ \quad happens(\_,ET,Ev), ET \leq T, \\ \quad initiates(Ev,Q), suceeds(Ev), \\ \quad not\ clipped(ET,Q,T), \\ \quad trajectory(Q,ET,P,T). \end{array}$$

where the predicate *trajectory* is like Sandewall's trajectory function $Trajs$ (def. 2.1.17). In order to perform backward reasoning, it is possible to augment

[2] The paper [DMB92] cites the Features and Fluents framework and claims (no assessment result is proved) the temporal reasoning in abductive Event Calculus adequate to solve "important benchmarks for temporal reasoning systems". Their adopted frame axioms are substantially identical to the above ones and, surprisingly, they count the Yale Shooting Problem, the Walking Turkey Shooting, the Russian Turkey Shooting and the Stanford Murder Mystery, in the list of the solved problems.

the calculus with an abductive procedure á la Eshghi (see [Esh88, EK89]). If we further extend the resulting calculus with similar ideas as in the work of Belleghem et al. [BDS94], then EC should reach the $\mathcal{K}$-$\mathbf{R}s\mathbf{A}d$ class of reasoning problems. We should proceed in this direction for a while, in constructing a new Event Calculus as the resulting calculus of several extensions, in order to *adjust* the reasoning process and *approach* the intended model set for $\mathcal{K}$-$\mathbf{RAC}i$. At that point a real assessment result will conclude the analysis. Such assessing analysis and the construction of the new Event Calculus, as implicit, is outside the scope of this thesis.

Assessment apart, the real problem that comes with EC is the *domain dependence*, that is the calculus and the problem are taken as a whole: the frame axioms *holds_at* and *clipped* are strictly defined in terms of the *initially*, *happens*, *initiates*, *terminates* and *succeeds* predicates, involving both the domain specification and the semantics.

7.6 On the Knowledge Level and its performances

The "ego", according to Sandewall's definition, corresponds to Newell's "Knowledge Level" [New82] of an intelligent agent, and is intended to be that part of the agent where reasoning is performed. Previous sections showed, without loss of generality, how the ego can be identified with its own knowledge base, consisting in a finite number of theories, object-level and meta-level ones. As an instance of Kowalski's "Algorithm = Logic + Control", and as a rediscovered of Sterling's [Ste84] "Expert System = Knowledge + Meta-interpreter", we here obtained the equation

$$\textit{Knowledge Level} = \textit{Object Level} + \textit{Meta Level}$$

Our construction of a calculus for computing feature values at different points in time, perfectly fits with Newell's understanding of the knowledge level [New82]:

> "*Logic is just a representation of knowledge. It is not the knowledge itself, but a structure at the symbol level.*" (page 110)
>
> "*Logic is the appropriate tool for analyzing the knowledge level, though it is often not the preferred representation to use for a given domain.*" (page 121)
>
> "*To determine what knowledge is exactly in the representation and to characterize it requires the use of logic.*" (page 121)

The Knowledge Level has been here reduced to the Symbol Level, via the property of equivalence of programs and data. Meta-interpreters, which are the technique for exploiting this property, are a very nice and simple tool to experiment the extent of new entailments, and have benefit from the direct experience of test examples. We remind the "ego" we defined is actually bounded to dynamic meta-level theories (the knowledge) and a static meta-interpreter oriented towards the pure control of deduction.

A session with the system

We developed a working prototype of the calculus, precisely a simplification of it for the $\mathcal{K}$g-**RAC***i* class of reasoning problems. The code is executable by a conventional PROLOG interpreter, and may also run on top of the Epsilon system [CFL+87, LMK87, CDF+86, DLS88, CFL+88]. The agent's vehicle consists in the poor hardware of "maya", an Intel Pentium-75 running LinuX, so that its perception device simply consists in a keyboard and its actuation device is the screen output. Let now call the system:

```
<serbr@maya> [~/tmp]
--> fc

 The Fluent Calculus - Version 1.0, Class Kg-RACi
 Copyright (c) 1997 by Sergio Brandano. All rights reserved.

 type "predicates" to view the list of available predicates.

1 :-
```

The system (the ego) is now ready and running. It actually has no knowledge in fact, for example:

```
1 :- history(3.1415927, a, V).

V = [unknown] ;

No
2 :-
```

thing that happens for a whatever feature at a whatever point in time (corollary 5.4.1). We now ask the ego to load the scenario description for the Yale Shooting Problem and to show us its resulting knowledge base:

```
2 :- load('yss.fc').

Yes
3 :- show_kb.

> Observations <
holds(0, a, 1).
holds(0, l, 0).

> Schedule <
do(5.15, 5.17, load).
do(5.3, 6.32, fire).

> Action Laws <
do(Z, X, load) :- holds(Y, l, to_load(Z, Y, X)).
do(Z, X, fire) :- holds(Y, l, to_fire(Z, Y, X)), holds(Y, a, to_die(Z, Y, X)).

Yes
4 :-
```

The *load*/1 predicate, reads from the file system the file that is specified in its parameter and maps its content $\Upsilon^\star$ from the object level to its meta-level representation Γ, storing the result in memory. What is the kstate of the world after the shooting ?

```
4 :-  show_kstate(7.13).

> kstate of the world at 7.13 <

< a, [0] >
< l, [0] >

Yes
5 :-
```

that is the gun is unloaded ($\langle l, 0\rangle$) and the turkey is not alive ($\langle a, 0\rangle$). What about a whatever feature f different from a or l ?

```
5 :- history(3.1415927, f, V).

V = [unknown] ;

No
6 :-
```

We now ask for the kernel of the completion set:

```
6 :- show_kernel.

> Kernel <

<    0, a, [1] >
<    0, l, [0] >
< 5.15, a, [1] >
< 5.15, l, [0] >
< 5.17, a, [1] >
< 5.17, l, [1] >
<  5.3, a, [1] >
<  5.3, l, [1] >
< 6.32, a, [0] >
< 6.32, l, [0] >

Yes
7 :-
```

The Closed World Assumption rule [Rei78] applies as follows: if a ground tuple $\langle t, f, v\rangle$ is not true, that is it is not a logical consequence of its scenario description Υ, that is $\langle t, f, v\rangle$ is false in every intended model for Υ, then infer $\neg\langle t, f, v\rangle$, that is $\langle t, f, \neg v\rangle$. In case of binary features, like truth-valued features, this rule solves, while for multi-valued features the use of the CWA does not provide the intended answer anymore, in fact it can never state for which value the tuple is true, apart from the trivial case of a finite $\mathcal{V}$ domain for which we could proceed exhaustively. Unlike the CWA, in our approach we can leave unspecified the value v, and the history does provides the intended value, the one for which the tuple is intended to be true. For example, let see the performance of our favorite turkey actor ...

```
7 :- history(5.927,a,V).

V = [0.788756] ;

No
8 :-
```

that is the turkey is not properly alive, nor dead. As shown, the axioms can be used to interact with the system. By augmenting the definition of Δ with few meta-programs to assert tuples, we get a Meta Level capable of knowledge acquisition, so that the problem domain may change in a dynamic fashion. Passing time we can add new observations in OBS, new action laws in LAW, or a schedule (plan) generated at run time in SCD. In this way the ego can be aware of the surrounding world and adapt its reasoning to incoming situations. For example, suppose we like to add the occurrence $do(8.30, 8.303, Load)$ in the actual knowledge base:

```
8 :- assert_scd(8.30, 8.303, load).

Yes
9 :-
```

the schedule is now updated, in fact:

```
9 :- show_scd.

> Schedule <
do(5.15, 5.17, load).
do(5.3, 6.32, fire).
do(8.3, 8.303, load).

Yes
10 :-
```

This is how the *perception process* takes place, and no extra information is required, since the system is domain independent and answers are generated at run time. The gun is now loaded at every time-point after 8.303.

Let make a test:

```
10 :-  show_kstate(9.12).

> kstate of the world at 9.12 <

< a, [0] >
< l, [1] >

Yes
11 :-
```

Let now spin the chamber of the gun:

```
11 :- assert_law(A, B, spin, holds(T, l, to_spin(A,T,B))).

Yes
12 :- assert_scd(5.19, 5.22, spin).

Yes
13 :- show_kb.

> Observations <
holds(0, a, 1).
holds(0, l, 0).

> Schedule <
do(5.15, 5.17, load).
do(5.3, 6.32, fire).
```

```
do(8.3, 8.303, load).
do(5.19, 5.22, spin).

> Action Laws <
do(Z, X, load) :- holds(Y, l, to_load(Z, Y, X)).
do(Z, X, fire) :- holds(Y, l, to_fire(Z, Y, X)), holds(Y, a, to_die(Z, Y, X)).
o(Z, X, spin) :- holds(Y, l, to_spin(Z, Y, X)).

Yes
14 :-
```

What is the kstate of the world at 5.30 ?

```
14 :-  show_kstate(5.30).

> kstate of the world at 5.30 <

< a, [1] >
< l, [1,0] >

Yes
15 :-
```

What is the kstate of the world at 6.32 ?

```
15 :-  show_kstate(6.32).

> kstate of the world at 6.32 <

< a, [1,0] >
< l, [0] >

Yes
16 :-
```

What is the kstate of the world at 9.12 ?

```
16 :-  show_kstate(9.12).

> kstate of the world at 9.12 <

< a, [1,0] >
< l, [1] >

Yes
17 :-
```

Now we are tired, and we leave the theater:

```
17 :- bye.

 see you! ...
```

Each session is registered into the “session.log” file of the system.

Performances of the system

The number of logical inferences for a meta-interpreter determines the overall performance of the system. The lower is that number, the better is the overall performance. The following table summarizes performances for some classical scenarios. It reports the number of logical inferences and the computing time required to compute the kernel of the completion set.

Scenario	Performance Rate
Yale Shooting Scenario	15,673 inferences in 0.26 seconds
Stanford Murder Mystery	07,733 inferences in 0.11 seconds
Russian Turkey Shooting	38,632 inferences in 0.73 seconds

that is instantaneous answers to queries, with a stack of 64k bytes. Above results are very satisfactory results, for a non optimized prototype meta-interpreter on a general purpose hardware machine, which performs backward *and* forward temporal reasoning with explicit continuous time. Above values are due to a sequential computation of the greatest lower and least upper observations. It is fairly possible to compute glo and luo as two independent and concurrent processes. To improve in performances we may also implement a cache. Caching is desirable, in fact the same queries are often computed with the same parameters, so that twin computations may (and must) be avoided, when possible. Care must be taken in using the cache; the semantics must not be compromised. Since the system is dynamic and non-monotonic, more knowledge (that is a more populated Γ) may imply a different value for those calls which answer is stored in cache. In the genuine spirit of meta-programming, the cache may be defined as a theory *cache* which elements are tuples like

$$\langle label, timepoint, information\rangle$$

where *label* is the name of the cached axiom, and *timepoint* is the *now* time-point at the time of cache update. At cache miss, the knowledge is computed and the cache is updated. At cache hit, if the corresponding *now* time-point of the cache element is smaller than the greatest time-point in Γ, then the knowledge has grown and the cache must be updated for that specific entry together with all the related entries.

The program transformation technique of *partial evaluation* originates from Kleene's S-m-n theorem [Kle52], which arises in recursive function theory. The first application to formal programming languages is due to Futamura [Fut71] and Sandewall [BHOS76], while the first application to logic programming is due to Komorowski [Kom81, LS91]. Recently, it was also proposed to avoid loss of performance of meta-interpreters [SS86a, CFL$^+$88, Gur94]. Unfortunately partial evaluation of meta-interpreters for Features and Fluents it does not provide such advantage, since the system is assumed to be dynamic, that is Γ is allowed to change over time, and Δ has to preserve its domain independence. However, we can see partial evaluation as a method to compile $\Delta \cup \Gamma$ to some variant of the Event Calculus, which is a domain dependent calculus.

Appendix A

A previous attempt to discrete time

In this appendix we describe a first attempt we made to relate Features and Fluents with Logic Programming. The Horn clause subset of the first order predicate logic is used, augmented with negation-as-failure. We first define the immersion of chronicles Υ in $\mathcal{K}gsp$-$\mathbf{I}b\mathbf{A}d\mathbf{C}i$ to finite sets of definite program clauses $\Upsilon^\star$, then we give a domain dependent frame axiom for temporal persistence which solves the frame problem. We also show a number of serious drawbacks, as some of the reasons we have been driven by, towards the results we provided in this thesis.

Assuming global features and unique names for features and actions, the Horn clause representation of scenario descriptions in $\mathcal{K}gsp$-$\mathbf{I}b\mathbf{A}d\mathbf{C}i$ is as follows: (1) for every observation $[t]f\hat{=}v$ in the OBS part of Υ, there is an atomic formula $holds(t, f, v) \leftarrow$ in $\Upsilon^\star$, where t, f are integer numbers in the Herbrand universe and v is a truth value; (2) for every action occurrence $[s,t]A$ in the SCD part of SCD, there is an atomic formula $do(s,t,A) \leftarrow$ in $\Upsilon^\star$, where s and t are integer numbers in the Herbrand universe and A is a character in the Herbrand universe; (3) for every action law like

$$[s,t]A \Rrightarrow [s]f_1 \Rightarrow [s,t]f_2 := v$$

in the LAW part of Υ, there is an Horn clause

$$holds(t, f_2, v) \leftarrow do(A, s, t), holds(s, f_1, true)$$

in $\Upsilon^\star$, where s and t are variables and reassignment statements $[s,t]f := v$ are mapped in atomic formulas $holds(t, f, v)$, expressing that certainly the value of the feature f will be v when the action terminates. Conjunctive preconditions are mapped in the conjunction of their respective atoms. A number m of conjunctive and consistent postconditions are mapped in m clauses such that the head of clause c_i has the postcondition p_i for $i = 1, \ldots, m$.

As an example, we write the corresponding Horn theory of (a ground version of) the scenario description 3.7.1 for the Yale Shooting Problem:

$holds(0, a, true)$
$holds(0, l, false)$
$do(2, 4, load)$
$do(6, 8, fire)$
$holds(T, l, true) \leftarrow do(S, T, load)$
$holds(T, a, false) \leftarrow do(S, T, fire), holds(S, l, true)$
$holds(T, l, false) \leftarrow do(S, T, fire), holds(S, l, true)$

A single *frame axiom* suffices to express what otherwise would require an *infinite* amount of *holds* axioms:

1. $holds(T, F, V) \leftarrow not\ occluded(F, T), 0 =< T - 1, holds(T - 1, F, V)$
2. $holds(T, F, unknown) \leftarrow$

The axiom asserts that for any feature f and time-point t, (1) if the feature is not occluded at time-point t, and $0 \leq t - 1$ because of the definition of time-point, then its value at time t equals its own value at the previous time-point $t - 1$, otherwise (2) the feature value is *unknown* (we assume an encapsulated semantics, as defined in [San94a]). We can refer to it as a *step-by-step* approach to temporal persistence. The *not* operator shall be interpreted as negation-as-failure [Cla78]. The axiom is in the lines of the circumscriptive frame axioms of McCarthy [McC86] and more precisely in the lines of the non-monotonic inference rule of Rychlik [Ryc90]:

$$\forall\, f, e, s\ .\ Holds(f, s), \neg Abnormal(f, e, s) \rightarrow Holds(f, result(e, s))$$

which allows to conclude a given fact being not *abnormal* unless its abnormality can be monotonically deduced from the currently available knowledge. The axiom is also close to the contraposition of the no-change axiom of Doherty [Doh94a] [1]:

$$\forall f, t\ .\ \neg Occlude(t + 1, f) \rightarrow Holds(t, f) \equiv Holds(t + 1, f)$$

which asserts that for any feature f and time-point t, if the feature f is not occluded at time-point $t + 1$ then its value at time $t + 1$ equals its own value at the previous time-point t.

Our Horn frame axiom is domain dependent because it makes use of the domain dependent axiom *occluded*. Completing the example, *occluded* is defined as follows:

$occluded(l, T) \leftarrow B < T, T =< E, do(B, E, load)$
$occluded(l, T) \leftarrow B < T, T =< E, do(B, E, fire)$
$occluded(a, T) \leftarrow B < T, T =< E, do(B, E, fire), holds(B, l, true)$

Note the test $holds(B, l, true)$ of the second clause is deliberately omitted, in order to avoid an infinite recursive call to $occluded(l, B)$. The intended model set for our ground instance of the YSP is the following:

[1] Doherty claims [Doh94a, page 31] his no-change axiom being "not Horn", and arguments about the possibility to pre-compile a query in order to remove the need for the no-change axiom in the theory.

in $[0\ldots6]$ a is true	in $[0\ldots2]$ l is false
in $(6\ldots8]$ a is occluded	in $(2\ldots4]$ l is occluded
in $[8\ldots+\infty)$ a is false	in $[4\ldots6]$ l is true
	in $(6\ldots8]$ l is occluded
	in $[8\ldots+\infty)$ l is false

Let P be the definition of *holds* and *occlude*. The SLDNF-resolution is properly driven in providing the right analysis of the particular problem specification.

PROOF. We prove that $P \vdash_{SLDNF} holds(6, l, true)$.

1. $do(2, 4, load) \in SCD$	(fact)
2. $holds(E, l, true) \leftarrow do(B, E, load) \in LAW$	(fact)
3. $holds(4, l, true)$	(unify 2 and 1)
4. $holds(6, l, true)$ $\square$	(inertia from 3)

PROOF. We prove that $P \vdash_{SLDNF} holds(8, a, false)$.

1. $do(6, 8, fire) \in SCD$	(fact)
2. $holds(E, a, false) \leftarrow do(B, E, fire), holds(B, l, true) \in LAW$	(fact)
3. $holds(6, l, true)$	(lemma)
4. $holds(8, a, false)$ $\square$	(unify 2 with 1 and 3)

In order to perform forward reasoning, that is in order to relax the s and p epistemological sub-characteristics, the control (frame axioms) should be further augmented. Abduction is a form of synthetic reasoning, which – as opposed to induction – infers explanations of observed facts, according to known and possibly general laws. Abductive consequences of an observation are such hypothesis that, when added to the theory, they should permit to derive (explain) the observation. Formally

> *Let be T a theory and O an observation (ground fact). The wff A (conjunction of ground facts) is derived from O in T per abduction, iff O is a logical consequence of $\{A\} \cup T$.*

$$T, O \vdash_{abd} A \Leftrightarrow T \cup \{A\} \models O$$

In practical applications we are interested in minimal abductive consequences, those which cannot be explained making more elementary assumptions. It is possible to restrict the range of possible hypothesis by explicitly declaring which are the abducible predicates. Referring to Eshghi [EK89], we augment P with a domain independent *abduce* axiom, which fine details can be found in [LP93].

$abduce(G, Th, []) \leftarrow syspred(G), call(G)$
$abduce(demo(G, T), Th, H) \leftarrow abduce(G, T, H)$
$abduce((G1, G2), Th, H) \leftarrow$
$\quad abduce(G1, Th, H1),$
$\quad abduce(G2, Th, H2),$
$\quad append(H1, H2, H)$
$abduce(G, Th, H) \leftarrow$
$\quad not\ syspred(G),$
$\quad not\ (G = (\_, \_)),$
$\quad not\ (G = demo(\_, \_)),$
$\quad clause\_t(G, B, Th),$
$\quad abduce(B, Th, H)$
$abduce(G, Th, [(G, Th)]) \leftarrow$
$\quad not\ syspred(G),$
$\quad not\ (G = (\_, \_)),$
$\quad not\ (G = demo(\_, \_)),$
$\quad not\ clause\_t(G, B, Th),$
$\quad demo(abducible(G), Th),$
$\quad ground(G)$

The extension involves meta-interpretation. In order to be useful in this context, the Horn theory Th shall be intended as being the theory $\Upsilon^\star$, the predicate *abduce* shall be augmented with the clause

$$\begin{array}{l} abduce(holds(T, F, V), Th, [(holds(T, F, V), Th)]) \leftarrow \\ \quad not\ (V = unknown), \\ \quad demo(abducible(holds(T, F, V)), Th), \\ \quad abduce(holds(T, F, unknown), Th, H). \end{array}$$

and there shall be a declaration like $abducible(holds(f, \_, \_))$ for every feature f of interest. The reader may easily verify the resulting control is adequate for solving the Stanford Murder Mystery and the Russian Turkey Shooting Problem [2].

Drawbacks: A number of drawbacks has been found for the step-by-step approach to temporal persistence, that discouraged further investigations. We can summarize them as follows. (1) Domain dependence. (2) Integer time only, because of the particular persistence axiom, then no continuous change. (3) It requires artful extensions in order to perform a proper backward and forward reasoning as well as satisfying all properties in the taxonomy of Sandewall. Right now, for example, given a simple scenario description with only one observation for a feature f at time-point t, the method fails at the query $[t-k]f$, for every $k \neq 0$, i.e. it fails a very simple forward analysis. The failure arise because action laws with preconditions are required in order to have benefit from the abduction. Concerning the backward analysis, it does not recognize the situation where more than one model exists. For example, given a scenario description with schedule $[s, t]load$ and the action law for loading, at every time point before t the gun it can be either loaded or unloaded because of the absence

[2] The action law $[s, t]Spin \Rrightarrow [s, t](l := true \vee l := false)$ shall be mapped to the Horn clause $holds(t, l, unknown) \leftarrow do(s, t, Spin)$.

of preconditions and further observations. (4) The method cannot be extended for actions with generic alternative results within positive logic programming. Disjunctive preconditions in fact can be simply written in the body, while disjunctive postconditions here require disjunctive logic programming, since postconditions are chosen to be head for clauses. An additional problem then arise, due to the different treatment of disjunctions in the head and those in the body of the resulting disjunctive clause. (5) The method is slow. Suppose for example a scenario description Υ with only one observation for a certain feature f and such that $f \notin Infl(a, \sigma)$ for all action a in Υ and state σ. If we ask for the value of the feature f at time $t + \{$time-points for two days$\}$, the step-by-step method will analyze each time-point backward in time for two days until t, where it finds the value. (6) The decidability is compromised, in fact:

- Suppose a scenario description Υ with no observations in OBS, no schedule's elements in SCD and no action laws in LAW for a certain feature f. If we ask for the value of the feature f at time-point t, the method performs a step-by-step backward search until the origo, where it does not find any answer. The solution may exists forward in time. If the method is properly extended in order to perform forward reasoning then, after the previous failure, it will continue going forward in time, to the infinity. It will not find any answer, because it does not exist. And there is no chance to solve the problem by fixing an arbitrarily big maximal-time-point, because the method will stop there reporting a "may be is out there, but definitely is not where I can look for".
- Suppose the method applied to a pre-post-diction problem. It has to go back and forth a number of times until it finds the value, if it exists, and diverges otherwise.

Bibliography

[Bak89] Andrew B. Baker. A simple solution to the Yale shooting problem. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 11–20, 1989.

[BDS94] Kristof Van Belleghem, Marc Denecker, and Danny De Schreye. Representing continuous change in the abductive event calculus. In *Proceedings of the International Conference on Logic Programming*, 1994.

[BHOS76] L. Beckman, Anders Haraldsson, Ö. Oskarsson, and Erik Sandewall. A partial evaluator and its use as a programming tool. *Artificial Intelligence*, 7:319–357, 1976.

[BK82] Kenneth A. Bowen and Robert Kowalski. Amalgamating language and metalanguage in logic programming. In K. L. Clark and S. A. Tärnlund, editors, *Logic Programming*, pages 153–172. Academic Press, New York, 1982.

[Bow85] Kenneth A. Bowen. Meta-Level Programming and Knowledge Representation. *New Generation Computing*, 3:359–383, 1985.

[Bru95] Maurice Bruynooghe. private conversation with. Linköping, May 1995.

[CDF+86] P. Coscia, S. Djennaoui, P. Franceschi, J. Kouloumdjian, G. Levi, L. Lei, G.H. Moll, I. de Saint Victor, G. Sardu, C. Simonelli, and L. Torre. The Epsilon Knowledge Base Management System: architecture and Data Base access optimization. In *Workshop on Logic Programming and Data Bases*, Venezia, Italy, December 1986.

[CFL+87] Patrizia Coscia, Paola Franceschi, Giorgio Levi, Giuseppe Sardu, and Luigia Torre. Object level reflection of inference rules by partial evaluation. In D. Nardi and P. Maes, editors, *Meta-level architectures and reflection*, pages 313–327. North-Holland, 1987.

[CFL+88] Patrizia Coscia, Paola Franceschi, Giorgio Levi, Giuseppe Sardu, and Luigia Torre. Meta-Level Definition and Compilation of Inference Engines in the Epsilon Logic Programming Environment. In R. A. Kowalski and K. A. Bowen, editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 359–373, Cambridge, Mass., 1988. MIT Press.

[Cla78] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York, 1978.

[Cor95] Silvia Coradeschi. Reasoning with unreliable observations in the Features and Fluents Framework. Tesi di laurea, Università di Pisa, Dipartimento di Scienze dell'Informazione, 1995.

[Cor96] Silvia Coradeschi. Reasoning with misperception in the Features and Fluents Framework. In W. Wahlster, editor, *Proceedings of the 12th European Conference on Artificial Intelligence*, pages 657–661. John Wiley & Sons, Ltd, 1996.

[Dav90] Ernest Davis. *Representation of Commonsense Knowledge.* Morgan Kauffman, 1990.

[Den93] Marc Denecker. *Knowledge Representation and Reasoning in Incomplete Logic Programming.* PhD thesis, University of Leuven, September 1993.

[DŁ94] Patrick Doherty and Witold Łukaszewicz. Circumscribing Features and Fluents. In D. Gabbay and H. J. Ohlbach, editors, *Proceedings of the 1th International Conference on Temporal Logic*, volume 827 of Lecture Notes on Artificial Intelligence, pages 82–100. Springer, 1994.

[DLS88] M. Degl'Innocenti, Giorgio Levi, and Giuseppe Sardu. A methodology for the design and the automatic composition of structured Prolog meta-interpreters. ESPRIT Project EPSILON P-530, May 1988.

[DŁS94] Patrick Doherty, Witold Łukaszewicz, and Andrzej Szałas. Computing Circumscription revisited: a reduction algorithm. Research Report LiTH-IDA-94-42, Department of Computer and Information Science, Linköping University, Sweden, December 1994.

[DMB92] M. Denecker, L. Missiaen, and M. Bruynooghe. Temporal reasoning with Abductive Event Calculus. In B. Neumann, editor, *Proceedings of the 10th European Conference on Artificial Intelligence*, pages 384–388. John Wiley & Sons, Ltd, 1992.

[Doh94a] Patrick Doherty. Notes on PMON Circumscription. Research Report LiTH-IDA-R-94-43, Department of Computer and Information Science, Linköping University, Sweden, December 1994.

[Doh94b] Patrick Doherty. Reasoning about action and change using occlusion. In *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 401–405, 1994.

[EK76] M. H. Van Emden and R. Kowalski. The Semantics of Predicate Logic as a Programming Language. *The Journal of the Association for Computing Machinery*, 23(4):733–742, October 1976.

[EK89] K. Eshghi and R. Kowalski. Abduction compared with negation by failure. In G. Levi and M. Martelli, editors, *Proceedings of the 6th International Conference and 5th Symposium on Logic Programming*, pages 234–254. MIT Press, 1989.

[Esh86] K. Eshghi. *Meta-Language in Logic Programming*. PhD thesis, Department of Computing, Imperial College, 1986.

[Esh88] K. Eshghi. Abductive planning with event calculus. In *Proceedings of the 5th International Conference on Logic Programming*, 1988.

[Fut71] Y. Futamura. Partial evaluation of computation process – an approach to a compiler-compiler. *Systems, Computers, Controls*, 2(5):721–728, 1971.

[Gär92] P. Gärdenfors. Belief Revision: A Vademecum. In A. Pettirossi, editor, *Meta-Programming in Logic. Third International Workshop, META'92*, volume 649 of *Lecture Notes in Computer Science*, pages 1–10, Uppsala, Sweden, June 10-12, 1992. Springer-Verlag.

[GL92] M. Gelfond and V. Lifschitz. Representing actions in extended logic programming. In K. Apt, editor, *Proceedings of the 9th International Joint Conference and Symposium on Logic Programming*, pages 559–573, 1992.

[GL93] M. Gelfond and V. Lifschitz. Representing Actions and Change by Logic Programs. *The Journal of Logic Programming*, 17:301–321, 1993.

[GT89] Scott D. Goodwin and André Trudel. Persistence in continuous first order temporal logics. Technical Report TR 89-24, University of Alberta, Dept. of Computer Science, October 1989.

[Gur94] Corin Alistair Gurr. A self-applicable partial evaluator for the logic programming language goedel (extended abstract). Technical Report Draft copy, Human Communication Research Centre, University of Edinburgh, 1994.

[Hau87] Brian Haugh. Simple causal minimizations for temporal persistence and projection. In *Proceedings of the 6th AAAI*, pages 218–223, 1987.

[HL88] P. M. Hill and J. W. Lloyd. Meta-Programming for Dynamic Knowledge Bases. Technical Report CS-88-18, Department of Computer Science, University of Bristol, 1988.

[HL89] P. M. Hill and J. W. Lloyd. Analysis of Meta-Programs. In H. D. Abramson and M. H. Rogers, editors, *Metaprogramming in Logic Programmig*, pages 23–51. MIT Press, 1989.

[HM86] Steve Hanks and Drew McDermott. Default Reasoning, Nonmonotonic logics, and the Frame Problem. In *Proceedings of the Fifth National (U.S.) Conference on Artificial Intelligence*, pages 328–333, Los Altos, CA, 1986. Morgan Kauffman.

[JL87] Joxan Jaffar and Jean-Louis Lassez. Constraint Logic Programming. In *Proceedings of the 14th ACM Symposium on Principles of Programming Languages*, pages 111–119, January 1987.

[JM94] Joxan Jaffar and Michael J. Maher. Constraint Logic Programming: a survey. *The Journal of Logic Programming*, 19,20:503–581, 1994.

[JMSY92] J. Jaffar, S. Michaylov, P. Stuckey, and R. Yap. The $CLP(\mathcal{R})$ Language and System. *ACM Transactions on Programming Languages*, 14(3):339–395, 1992.

[Kle52] S. K. Kleene. *Introduction to Metamathematics*. North-Holland, 1952.

[Kom81] H. J. Komorowski. A specification of an abstract PROLOG machine and its application to partial evaluation. LSST 69, Linköping University, 1981.

[Kow74] Robert Kowalski. Predicate logic as a programming language. In *Proceedings of the IFIP Congress*, pages 569–574, Stockholm, 1974. North-Holland.

[Kow79] Robert Kowalski. Algorithm = Logic + Control. *Communications of the Association for Computing Machinery*, 22(7):424–436, 1979.

[KS86] Robert Kowalski and Marek Sergot. A Logic-based Calculus of Events. *New Generation Computing*, 4(1):67–95, 1986.

[Lif88] V. Lifschitz. Pointwise Circumscription. In M. Ginsberg, editor, *Readings in Non-monotonic Reasoning*, pages 179–193. Morgan Kauffman, 1988.

[LMK87] G. Levi, M. Modesti, and J. Kouloumdjian. Status and evolution of the Epsilon prototype. In *ESPRIT '87*, pages 593–610. North-Holland, 1987.

[LP93] Giorgio Levi and Frédédic Patricelli. *PROLOG, Linguaggio, applicazioni ed implementazioni*. Scuola Superiore G. Reiss Romoli, L'Aquila, Italia, 1993.

[LR93] Giorgio Levi and Davide Ramundo. A formalization of metaprogramming for real. In D. S. Warren, editor, *Proceedings of the 10th International Conference on Logic Programming*, pages 354–373. MIT Press, 1993.

[LS91] J. W. Lloyd and J. C. Shepherdson. Partial evaluation in logic programming. *The Journal of Logic Programming*, 11:217–242, 1991.

[McC86] John McCarthy. Applications of Circumscription to Formalizing Commonsense Knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[MD80] D. McDermott and J. Doyle. Non-monotonic logic i. *Artificial Intelligence*, 13:41–72, 1980.

[Mis91] L. Missiaen. *Localized abductive planning with the Event Calculus.* PhD thesis, Department of Computer Science, University of Leuven, 1991.

[New82] Allen Newell. The Knowledge Level. *Artificial Intelligence*, 18(1):87–127, 1982.

[Rei56] H. Reichenbach. *The Direction of Time.* 1956.

[Rei78] Raymond Reiter. On Closed World Data Bases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 55–76. Plenum Press, New York, 1978.

[Rei80] Raymond Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13(1–2):81–132, 1980.

[Ryc90] Piotr Rychlik. The Generalized Theory of Model Preference (preliminary report). In *National (U.S.) Conference on Artificial Intelligence*, pages 615–620, 1990.

[San72] Erik Sandewall. An Approach to the frame problem, and its Implementation. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 7. Edinburgh University Press, Edinburgh, UK, 1972.

[San89a] Erik Sandewall. Combining logic and differential equations for describing real-world systems. In *Proceedings of the Knowledge Representation Conference*, page 412, 1989.

[San89b] Erik Sandewall. Filter preferential entailment for the logic of action in almost continuous worlds. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, Detroit, USA, 1989.

[San93] Erik Sandewall. The range of applicability of nonmonotonic logics for the inertia problem. Technical Report LiTH-IDA-R-93-08, Department of Computer and Information Science, Linköping University, Sweden, April 1993. Accepted to the International Joint Conference on Artificial Intelligence '93; Chambery, France.

[San94a] Erik Sandewall. Features and Fluents. A Systematic Approach to the Representation of Knowledge about Dynamical Systems. — final review version. Technical Report LiTH-IDA-R-94-15, Department of Computer and Information Science, Linköping University, Sweden, 1994. Published as [San94b].

[San94b] Erik Sandewall. *Features and Fluents. The Representation of Knowledge about Dynamical Systems*, volume 1 of *Oxford Logic Guides.* Oxford University Press, 1994.

[San95] Erik Sandewall. Current Research in RKLLAB, Linköping University's Laboratory for the Representation of Knowledge in Logic. Memo IDA-RKL-95-TR01, Department of Computer and Information Science, Linköping University, Sweden, 1995. ISSN-0281-4250.

[SB86] L. Sterling and R. D. Beer. Meta-Interpreters for Expert System Construction. Technical Report TR 86-122, Center for Automation and Intelligent Systems Research, Case Western Reserve University, 1986.

[Sha90] M. Shanahan. Representing continuous change in the Event Calculus. In *Proceedings of the 9th European Conference on Artificial Intelligence*, page 598, 1990.

[Sho89] Yoav Shoham. *Reasoning about Change – Time and Causation from the Standpoint of Artificial Intelligence.* MIT Press, 1989.

[SK95] Fariba Sadri and Robert Kowalski. Variants of the Event Calculus. In Leon Sterling, editor, *Proceedings of the International Conference on Logic Programming*, pages 67–81, 1995.

[SM92] D. De Schreye and B. Martens. A sensible least Herbrand semantics for untyped Vanilla meta-programming and its extension to a limited form of amalgamation. In A. Pettirossi, editor, *Meta-Programming in Logic. Third International Workshop, META'92*, volume 649 of *Lecture Notes in Computer Science*, pages 192–204, Uppsala, Sweden, June 10-12, 1992. Springer-Verlag.

[SS86a] S. Safra and E. Shapiro. Meta Interpreters for Real. In H. J. Kugler, editor, *Information Processing 86*, pages 271–278. North-Holland, 1986.

[SS86b] Leon Sterling and Ehud Shapiro. *The Art of Prolog.* MIT Press, 1986.

[SS94a] Erik Sandewall and Yoav Shoham. Nonmonotonic temporal reasoning. Technical Report IDA-RKL-94-TR11, Department of Computer and Information Science, Linköping University, Sweden, January 1994. Published as [SS94b].

[SS94b] Erik Sandewall and Yoav Shoham. Nonmonotonic temporal reasoning. In D. M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Artificial Intelligence and Logic Programming*, volume 4. Oxford University Press, 1994.

[Ste84] Leon Sterling. Expert System = Knowledge + Meta-interpreter. Technical Report CS84-17, Weizmann Institute of Science, Rehovot, Israel, 1984.

[Ste85] Leon Sterling. Meta-interpreters for expert systems. Technical Report CAISR-TR-134-85, Case Western Reserve University, 1985.

[Sub88] V. S. Subrahmanian. Foundations of metalogic programming. In J. W. Lloyd, editor, *Workshop on Meta-Programming in Logic Programming*, pages 53–66, June 1988.

[Tak86] Akikazu Takeuchi. Affinity between meta-interpreters and partial evaluation. In H. J. Kugler, editor, *Information Processing 86*, pages 279–282. North-Holland, 1986.

[Tar55] Alfred Tarski. A Lattice-theoretical Fixpoint Theorem and its Applications. *Pacific J. Math*, 5:285–309, 1955.

[Thi94] M. Thielscher. An analysis of systematic approaches to reasoning about actions and change. In P. Jorrand, editor, *Proceedings of the International Conference on AIMSA*, Sofia, Bulgaria, 1994.