# Representing a Robotic Domain Using Temporal Description Logics

**Alessandro Artale** ARTALE@IRST.ITC.IT
*IRST, I-38050 Povo TN, Italy*
**Enrico Franconi** FRANCONI@CS.MAN.AC.UK
*Computer Science Dept., Univ. of Manchester, UK*

## Abstract

A temporal logic for representing and reasoning on a robotic domain is presented. Actions are represented by describing what is true while the action itself is occurring, and plans are constructed by temporally relating actions and world states. The temporal language is a member of the family of Description Logics, which are characterized by high expressivity combined with good computational properties. The logic is used to organize the domain actions and plans in a taxonomy. The *classification* and *recognition* tasks, together with the *subsumption* task form the basis for action management. An action/plan description can be automatically *classified* into a taxonomy; an action/plan instance can be *recognized* to take place at a certain moment from the observation of what is happening in the world during a time interval.

## 1. Introduction

In the Description Logic[1] literature, several approaches for representing and reasoning with time dependent concepts have been proposed. Here the usefulness of a temporal description logic to represent actions and plans in a robotic domain is illustrated.

The most common approaches to model actions are based on the notion of *state change* – e.g., the formal models based on the original *situation calculus* (McCarthy & Hayes, 1969; Sandewall & Shoham, 1994) or the STRIPS-like planning systems (Fikes & Nilsson, 1971; Lifschitz, 1987) – in which actions are generally considered instantaneous and defined as functions from one state to another by means of pre- and post-conditions. In the logic presented here, an explicit notion of time is introduced in the modeling language and actions are defined as *occurring over time intervals*, following the Allen proposal (Allen, 1991). In this formalism an action is represented by describing the time course of the world while the action occurs. Concurrent or overlapping actions are allowed: effects of overlapping actions can be different from the sum of their individual effects; effects may not directly follow the action but more complex temporal relations may hold.

An interval temporal logics is presented based on description logics and inspired by the works of Schmiedel (Schmiedel, 1990) and of Weida and Litman (Weida & Litman, 1992). Advantages of using description logics are their high expressivity combined with desirable computational properties – such as decidability, soundness and completeness of reasoning procedures (Buchheit, Donini, & Schaerf, 1993; Schaerf, 1994; Donini & Era, 1992; Donini, Lenzerini, Nardi, & Schaerf, 1994; Donini, Lenzerini, Nardi, & Nutt, 1995). To represent the temporal dimension classical description logics are extended with temporal constructors;

1. Description Logics have been also called *Frame-Based Description Languages*, *Term Subsumption Languages*, *Terminological Logics*, *Taxonomic Logics*, *Concept Languages* or KL-ONE-*like languages*.

thus a uniform representation for states, actions and plans is provided. In this framework a *state* describes a collection of properties of the world holding at a certain time. *Actions* are represented through temporal constraints on world states, which pertain to the action itself. *Plans* are built by temporally relating actions and states. Furthermore, the distinction made by description logics between the terminological and assertional aspects of the knowledge allows us to describe actions and plans both at an abstract level (action/plan types) and at an instance level (individual actions and plans).

This work is motivated by the research and development of a hybrid knowledge representation system which should be able to represent states, actions and plans in the robotic domain of the integrated system MAIA (*Advanced Model of Artificial Intellingence*). MAIA is a project iniitially developed at IRST that concerns a robot which moves around the IRST offices to transport objects, essentially papers and books. In our framework the robot has a box which contains the transported objects. It can distinguish between rooms and corridors while moving around, and between doors along a corridor and studio doors. All these distinctions are recognized by a continuous sensing. This domain is similar to the one studied by Kabanza, Barbeu and St-Denis(1997).

The robot has *actuators* to perform the elementary actions. The elementary actions are to ask for a permission in order to open a studio door, open a door, close a door, open its own holding box, and to move along corridors and inside rooms. There are also exogenous actions which are under the human control. These last actions model the interactions between the robot and the human beings and are of two kinds. The first is consequent to the request made by the robot to open a studio door: the human acknowledge the robot request by allowing or negating the door opening. The second has to occur during the robot release action to take away the transported object from the robot box: the accomplishment of this action is monitored by the human closing the robot box.

In this environment, the *subsumption* calculus – i.e., the superconcept/subconcept relation – is the main reasoning tool offered by the representation system for managing collections of action and plan types. Action and plan types can be organized in a subsumption-based taxonomy, which plays the role of an action/plan library to be used for the tasks known in the literature as plan retrieval and individual plan recognition (Kautz, 1991; Devanbu & Litman, 1996). A refinement of the *plan recognition* notion is proposed, by splitting it into the different tasks of *plan description classification* – involving a plan type – and *specific plan recognition with respect to a plan description* – involving an individual plan. According to the latter reasoning task, the system is able to recognize which type of action/plan has taken place at a certain time interval, given a set of observations of the world.

The paper is organized as follows. After introducing the main features of description logics in Section 2, the temporal language $\mathcal{TL}$-$\mathcal{ALCF}$ is introduced in Section 3. The language syntax is first described, together with a worked out example illustrating the informal meaning of temporal expressions. Section 4 shows that the temporal language is suitable for action and plan representation and reasoning in the robotic domain. Section 5 surveys the literature of extensions of description logics for representing and reasoning with time and actions. This Section is concluded by a comparison with *State Change* based approaches by briefly illustrating the effort made in the situation calculus area to temporally extend

$$\begin{array}{lcll}
C, D & \rightarrow & A \mid & \text{(atomic concept)} \\
 & & \top \mid & \text{(top)} \\
 & & \bot \mid & \text{(bottom)} \\
 & & \neg C \mid & \text{(complement)} \\
 & & C \sqcap D \mid & \text{(conjunction)} \\
 & & C \sqcup D \mid & \text{(disjunction)} \\
 & & \forall P.C \mid & \text{(universal quantifier)} \\
 & & \exists P.C \mid & \text{(existential quantifier)} \\
 & & p : C \mid & \text{(selection)} \\
 & & p \downarrow q \mid & \text{(agreement)} \\
 & & p \uparrow q \mid & \text{(disagreement)} \\
 & & p \uparrow \mid & \text{(undefinedness)} \\
p, q & \rightarrow & f \mid & \text{(atomic feature)} \\
 & & p \circ q & \text{(path)}
\end{array}$$

Figure 1: Syntax rules for the $\mathcal{ALCF}$ Description Logic.

this class of formalisms. Section 6 makes some conclusions and shows future directions. Appendix A reveals the model theoretic semantics of $\mathcal{TL}$-$\mathcal{ALCF}$, together with a formal definition of the subsumption and instance recognition problems. Results about the sound and complete calculus for the temporal language is summarized in Appendix B.

## 2. An Introduction to Description Logics

In this section we give a brief introduction to the formal framework of Description Logics. The presentation of the formal apparatus will strictly follow the formalism introduced by (Schmidt-Schauß & Smolka, 1991) and further elaborated by (Hollunder & Nutt, 1990; Donini, Lenzerini, Nardi, & Nutt, 1991; Donini, Hollunder, Lenzerini, Spaccamela, Nardi, & Nutt, 1992; Donini et al., 1994, 1995; Buchheit et al., 1993; De Giacomo & Lenzerini, 1995, 1996): in this perspective, Description Logics are considered as a *structured* fragment of predicate logic. $\mathcal{ALC}$ (Schmidt-Schauß & Smolka, 1991) is the minimal Description Logic including full negation and disjunction – i.e., propositional calculus.

The basic types of a description language are *concepts*, *roles*, and *features*. A concept is a description gathering the common properties among a collection of individuals; from a logical point of view it is a unary predicate ranging over the domain of individuals. Properties are represented either by means of roles – which are interpreted as binary relations associating to individuals of a given class values for that property – or by means of features – which are interpreted as functions associating to individuals of a given class a *single* value for that property. In the following, we will consider the language $\mathcal{ALCF}$ (Hollunder & Nutt, 1990), extending $\mathcal{ALC}$ with features (i.e., functional roles).

According to the syntax rules of Figure 1, $\mathcal{ALCF}$ *concepts* (denoted by the letters $C$ and $D$) are built out of *atomic concepts* (denoted by the letter $A$), *atomic roles* (denoted by the letter $P$), and *atomic features* (denoted by the letter $f$). The syntax rules are expressed following the tradition of description logics: they can be read as, e.g., if $C$ and $D$ are concept expressions then $C \sqcap D$ is a concept expression, too.

Let us now consider the formal semantics of the description logic. We define the *meaning* of concept expressions as sets of individuals – as for unary predicates – and the meaning of roles as sets of pairs of individuals – as for binary predicates. Formally, an *interpretation* is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consisting of a set $\Delta^{\mathcal{I}}$ of individuals (the *domain* of $\mathcal{I}$) and a function $\cdot^{\mathcal{I}}$ (the *interpretation function* of $\mathcal{I}$) mapping every concept to a subset of $\Delta^{\mathcal{I}}$, every role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and every feature to a partial function from $\Delta^{\mathcal{I}}$ to $\Delta^{\mathcal{I}}$, such that the equations of the left column in Figure 2 are satisfied. The semantics of the language can also be given by stating equivalences among expressions of the language and First Order Logic formulae. An atomic concept $A$, an atomic role $P$, and an atomic feature $f$, are mapped respectively to the open formulæ$A(\gamma)$, $P(\alpha, \beta)$, and $f(\alpha, \beta)$ – with $f$ a functional relation, also written $f(\alpha) = \beta$. The rightmost column of figure 2 gives the transformational semantics of $\mathcal{ALCF}$ expressions in terms of equivalent FOL well-formed formulæ. A concept $C$, a role $P$ and a path $p$ correspond to the FOL open formulae $F_C(\gamma)$, $F_P(\alpha, \beta)$, and $F_p(\alpha, \beta)$, respectively. It is worth noting that, using the standard model-theoretic semantics, the extensional semantics of the left column can be derived from the transformational semantics of the right column.

A *terminology* or *TBox* is a finite set of *terminological axioms.* For an atomic concept $A$ – called *defined concept* – and a (possibly complex) concept $C$, a terminological axiom is of the form $A \doteq C$[2]. An atomic concept not appearing on the left-hand side of any terminological axiom is called a *primitive concept.* We consider *acyclic simple* TBoxes only: a defined concept may appear at most once as the left-hand side of an axiom, and no terminological cycles are allowed, i.e., no defined concept may occur – neither directly nor indirectly – within its own definition (Nebel, 1991). An interpretation $\mathcal{I}$ *satisfies* $A \doteq C$ if and only if $A^{\mathcal{I}} = C^{\mathcal{I}}$.

As an example of the expressive power of $\mathcal{ALCF}$, we can consider the unary relation (i.e., a concept) denoting the class of `Researchers`, defined using the atomic predicates `Employee, Professor, Article, Journal, Conference` (concepts), `WRITTEN-PAPER` (role) and `HAS-HEAD, PUBLISHED` (features):

$$\begin{aligned}\texttt{Researcher} \doteq\ & \texttt{Employee} \sqcap (\texttt{HAS-HEAD} : \texttt{Professor}) \sqcap (\exists \texttt{WRITTEN-PAPER}.\top) \sqcap \\ & \forall \texttt{WRITTEN-PAPER}.(\texttt{Article} \sqcap \texttt{PUBLISHED} :(\texttt{Journal} \sqcup \texttt{Conference}))\end{aligned}$$

i.e., a researcher is an employee with a head (exactly one!), who wrote at least one paper, and all of this papers are articles published in some journal or conference.

An *ABox* is a finite set of *assertional axioms*, i.e. predications on individual objects. Let $\mathcal{O}$ be the alphabet of symbols denoting *individuals*; an assertion is an axiom of the form $C(a)$, $R(a, b)$ or $p(a, b)$, where $a$ and $b$ denote individuals in $\mathcal{O}$. The interpretation $\mathcal{I}$ is extended over individuals in such a way that $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ for each individual $a \in \mathcal{O}$, and $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ if $a \neq b$ (unique name assumption). $C(a)$ is satisfied by an interpretation $\mathcal{I}$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$, $P(a, b)$ is satisfied by $\mathcal{I}$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$, and $p(a, b)$ is satisfied by $\mathcal{I}$ iff $p^{\mathcal{I}}(a^{\mathcal{I}}) = b^{\mathcal{I}}$.

2. Terminological axioms of the form $A \dot{\leq} C$ imposing only necessary conditions on the interpretation of the introduced concepts can be rewrited as $A =: A* \sqcap C$ where the newly introduced concept name $A*$ stands for the absent part of the definition of $A$.

| | | |
|---|---|---|
| $\top^{\mathcal{I}} =$ | $\Delta^{\mathcal{I}}$ | true |
| $\bot^{\mathcal{I}} =$ | $\emptyset$ | false |
| $(\neg C)^{\mathcal{I}} =$ | $\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$ | $\neg F_C(\gamma)$ |
| $(C \sqcap D)^{\mathcal{I}} =$ | $C^{\mathcal{I}} \cap D^{\mathcal{I}}$ | $F_C(\gamma) \wedge F_D(\gamma)$ |
| $(C \sqcup D)^{\mathcal{I}} =$ | $C^{\mathcal{I}} \cup D^{\mathcal{I}}$ | $F_C(\gamma) \vee F_D(\gamma)$ |
| $(\exists P.C)^{\mathcal{I}} =$ | $\{a \in \Delta^{\mathcal{I}} \mid \exists b.(a,b) \in P^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$ | $\exists x.F_P(\gamma,x) \wedge F_C(x)$ |
| $(\forall P.C)^{\mathcal{I}} =$ | $\{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in P^{\mathcal{I}} \Rightarrow b \in C^{\mathcal{I}}\}$ | $\forall x.F_P(\gamma,x) \Rightarrow F_C(x)$ |
| $(p : C)^{\mathcal{I}} =$ | $\{a \in dom\, p^{\mathcal{I}} \mid p^{\mathcal{I}}(a) \in C^{\mathcal{I}}\}$ | $\exists x.F_p(\gamma,x) \wedge F_C(x)$ |
| $p \downarrow q^{\mathcal{I}} =$ | $\{a \in dom\, p^{\mathcal{I}} \cap dom\, q^{\mathcal{I}} \mid p^{\mathcal{I}}(a) = q^{\mathcal{I}}(a)\}$ | $(\exists x.F_p(\gamma,x) \wedge F_q(\gamma,x)) \wedge$ $(\forall x,y.F_p(\gamma,x) \wedge F_q(\gamma,y) \to x = y)$ |
| $p \uparrow q^{\mathcal{I}} =$ | $\{a \in dom\, p^{\mathcal{I}} \cap dom\, q^{\mathcal{I}} \mid p^{\mathcal{I}}(a) \neq q^{\mathcal{I}}(a)\}$ | $(\exists x,y.F_p(\gamma,x) \wedge F_q(\gamma,y)) \wedge$ $(\forall x,y.F_p(\gamma,x) \wedge F_q(\gamma,y) \to x \neq y)$ |
| $(p\uparrow)^{\mathcal{I}} =$ | $\Delta^{\mathcal{I}} \setminus dom\, p^{\mathcal{I}}$ | $\neg\exists x.F_p(\gamma,x)$ |
| $(p \circ q)^{\mathcal{I}} =$ | $p^{\mathcal{I}} \circ q^{\mathcal{I}}$ | $\exists x.F_p(\alpha,x) \wedge F_q(x,\beta)$ |

Figure 2: The extensional and transformational semantics in $\mathcal{ALCF}$.

A *knowledge base* is a finite set $\Sigma$ of terminological and assertional axioms (i.e., $\Sigma = \langle TBox, ABox \rangle$). An interpretation $\mathcal{I}$ is a *model* of a knowledge base $\Sigma$ iff every axiom of $\Sigma$ is satisfied by $\mathcal{I}$.

Let us describe now the basic reasoning services provided by a DL-system. From $\Sigma$ does not *logically follows* that $C \equiv \bot$ (written $\Sigma \not\models C \equiv \bot$) if there exists a model $\mathcal{I}$ of $\Sigma$ such that $C^{\mathcal{I}} \neq \emptyset$: we say that $C$ is *satisfiable* and we indicate this reasoning problem as *concept satisfiability.* $\Sigma$ *logically implies* $D \sqsubseteq C$ (written $\Sigma \models D \sqsubseteq C$) if $D^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ for every model of $\Sigma$: we say that $D$ is *subsumed* by $C$ in $\Sigma$. The reasoning problem of checking whether $D$ is *subsumed* by $C$ in $\Sigma$ is called *subsumption checking.* We write $\Sigma \not\models$ to indicate the problem of checking whether $\Sigma$ has a model, a problem called *knowledge base consistency.* $\Sigma$ *logically implies* $C(a)$ (written $\Sigma \models C(a)$) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for every model of $\Sigma$: we say that $a$ is an *instance* of $C$ in $\Sigma$. The reasoning problem of checking whether $a$ is an *instance* of $C$ in $\Sigma$ is called *instance checking.* Notice that for propositionally complete languages we have that $\Sigma \models D \sqsubseteq C$ if and only if $\Sigma \models D \sqcap \neg C \equiv \bot$, and $\Sigma \models C(a)$ if and only if $\Sigma \cup \{\neg C(a)\} \not\models$. In other words, subsumption can be reduced to satisfiability and instance checking to knowlege base consistency.

An acyclic simple TBox can be transformed into an *expanded* TBox having the same models, where no defined concept makes use in its definition of any other defined concept. In this way, the interpretation of a defined concept in an expanded TBox does not depend from any other defined concept. It is easy to see that $D$ is subsumed by $C$ in $\Sigma$ with an acyclic simple TBox if and only if the expansion of $D$ (w.r.t. $\Sigma$) is subsumed by the expansion of $C$ (w.r.t. $\Sigma$) in $\Sigma' = \langle \emptyset, ABox \rangle$ (i.e., the knowledge base with an the empty TBox). The expansion procedure recursively substitutes every defined concept occurring in a definition with its defining expression; such a procedure may generate a TBox exponential in size, but it was proven (Nebel, 1990) that it works in polynomial time under reasonable restrictions. In the following we will interchangeably refer either to reasoning with respect to a TBox or to reasoning involving expanded concepts with respect to an empty TBox.

| Relation | Abbr. | Inverse |
|---|---|---|
| before$(i,j)$ | b | a |
| meets$(i,j)$ | m | mi |
| overlaps$(i,j)$ | o | oi |
| starts$(i,j)$ | s | si |
| during$(i,j)$ | d | di |
| finishes$(i,j)$ | f | fi |

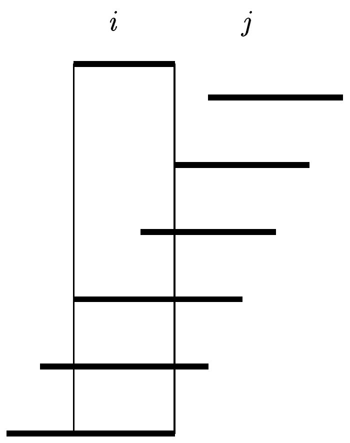

Figure 3: The Allen's interval relationships.

## 3. The Temporal Description Logic

The temporal language $\mathcal{TL}$-$\mathcal{ALCF}$ is the logic considered here (Artale & Franconi, 1994, 1998). This language is composed of the temporal Logic $\mathcal{TL}$ – able to express interval temporal networks – and the non-temporal description logic $\mathcal{ALCF}$.

### 3.1 Syntax

Basic types of the language are *concepts*, *individuals*, *temporal variables* and *intervals*. Concepts can describe entities of the world, states and events. Temporal variables denote intervals bound by temporal constraints, by means of which abstract temporal patterns in the form of constraint networks are expressed. Concepts (resp. individuals) can be specified to hold at a certain temporal variable (resp. interval). In this way, *action types* (resp. *individual actions*) can be represented in a uniform way by temporally related concepts (resp. individuals).

For the basic temporal interval relations the Allen notation (Allen, 1991) (figure 3) is used: before (b), meets (m), during (d), overlaps (o), starts (s), finishes (f), equal (=), after (a), met-by (mi), contains (di), overlapped-by (oi), started-by (si), finished-by (fi). *Concept expressions* (denoted by $C, D$) are syntactically built out of *atomic concepts* (denoted by $A$), *atomic features* (denoted by $f$), *atomic parametric features* (denoted by $\star g$), *atomic roles* (denoted by $P$) and *temporal variables* (denoted by $X, Y$). Temporal concepts $(C, D)$ are distinguished from non-temporal concepts $(E, F)$, following the syntax rules of Figure 4. Names for atomic features and atomic parametric features are from the same alphabet of symbols; the $\star$ symbol is not intended as operator, but only as differentiating the two syntactic types.

Temporal variables are introduced by the temporal existential quantifier "$\diamond$" – excluding the special temporal variable $\sharp$, usually called NOW, and intended as the reference interval. Variables appearing in temporal constraints $(\overline{Tc})$ must be declared within the same temporal quantifier, with the exception of the special variable $\sharp$. Temporal variables appearing in the right hand side of an "@" operator are called *bindable*. Concepts must not include *unbound* (a.k.a. *free*) bindable variables. Informally, a bindable variable is said to be *bound*

| | | | | |
|---|---|---|---|---|
| $\mathcal{TL}$ | $C, D$ | $\rightarrow$ | $E \mid$ | (non-temporal concept) |
| | | | $C \sqcap D \mid$ | (conjunction) |
| | | | $C@X \mid$ | (qualifier) |
| | | | $C[Y]@X \mid$ | (substitutive qualifier) |
| | | | $\diamond(\overline{X})\ \overline{Tc}.C$ | (existential quantifier) |
| | $Tc$ | $\rightarrow$ | $(X\ (R)\ Y) \mid$ | (temporal constraint) |
| | | | $(X\ (R)\ \sharp) \mid$ | |
| | | | $(\sharp\ (R)\ Y)$ | |
| | $\overline{Tc}$ | $\rightarrow$ | $Tc \mid Tc\ \overline{Tc}$ | |
| | $R, S$ | $\rightarrow$ | $R\ ,\ S \mid$ | (disjunction) |
| | | | $\mathsf{s} \mid \mathsf{mi} \mid \mathsf{f} \mid \ldots$ | (Allen's relations) |
| | $X, Y$ | $\rightarrow$ | $\mathrm{x} \mid \mathrm{y} \mid \mathrm{z} \mid \ldots$ | (temporal variables) |
| | $\overline{X}$ | $\rightarrow$ | $X \mid X\ \overline{X}$ | |
| $\mathcal{ALCF}$ | $E, F$ | $\rightarrow$ | $A \mid$ | (atomic concept) |
| | | | $\top \mid$ | (top) |
| | | | $\bot \mid$ | (bottom) |
| | | | $\neg E \mid$ | (complement) |
| | | | $E \sqcap F \mid$ | (conjunction) |
| | | | $E \sqcup F \mid$ | (disjunction) |
| | | | $\forall P.E \mid$ | (universal quantifier) |
| | | | $\exists P.E \mid$ | (existential quantifier) |
| | | | $p : E \mid$ | (selection) |
| | | | $p \downarrow q \mid$ | (agreement) |
| | | | $p \uparrow q \mid$ | (disagreement) |
| | | | $p \uparrow$ | (undefinedness) |
| | $p, q$ | $\rightarrow$ | $f \mid$ | (atomic feature) |
| | | | $\star g \mid$ | (atomic parametric feature) |
| | | | $p \circ q$ | (path) |

Figure 4: Syntax rules for the interval Description Logic $\mathcal{TL}$-$\mathcal{ALCF}$

in a concept if it is declared at the nearest temporal quantifier in the body of which it occurs; this avoid the usual formal inductive definition of a bound variable. Moreover, in chained constructs of the form $((C[Y_1]@X_1)[Y_2]@X_2\ldots)$ non bindable variables – i.e., the ones on the left hand side of an "@" operator – cannot appear more than once. Note that, since description logics are a fragment of FOL with one free variable, the above mentioned restrictions force the temporal side of the language to have only one free temporal variable, i.e., the reference time $\sharp$.

As usual, terminological axioms for building simple acyclic $\mathcal{TL}$-$\mathcal{ALCF}$ TBoxes are allowed. While using in a concept expression a name referring to a defined concept, it is possible to use the substitutive qualifier construct, to impose a coreference with a variable appearing in the definition associated to the defined concept. The statement $C[Y]@X$ constrains the variable $Y$, which should appear in the definition of the defined concept $C$, to corefer with $X$ (see Section 4.2 for an example). A drawback in the use of this operator is the requirement to know the internal syntactical form of the defined concept, namely, the names of its temporal variables.

Let $\mathcal{O}$ and $\mathcal{OT}$ be two alphabets of symbols denoting *individuals* and *temporal intervals*, respectively. An assertion – i.e., a predication on temporally qualified individual entities – is a statement of one of the forms $C(i,a)$, $P(i,a,b)$, $p(i,a,b)$, $\star g(a,b)$, $R(i,j)$, where $C$ is a concept, $P$ is a role, $p$ is a feature, $\star$g is a parametric feature, $R$ is a temporal relation, $a$ and $b$ denote individuals in $\mathcal{O}$, $i$ and $j$ denote temporal intervals in $\mathcal{OT}$.

## 3.2 Informal Semantics

Let us now informally see the intended *meaning* of the terms of the language $\mathcal{TL}$-$\mathcal{ALCF}$ (for the formal details see the appendix A). Concept expressions are interpreted over pairs of *temporal intervals* and *individuals* $\langle i, a\rangle$, meaning that the individual $a$ is in the extension of the concept at the interval $i$. If a concept is intended to describe an action, then its interpretation can be seen as the set of individual actions of that type occurring at some interval.

Within a concept expression, the special "$\sharp$" variable denotes the current interval of evaluation; in the case of actions, it is thought that it refers to the temporal interval at which the action itself *occurs*. The temporal existential quantifier introduces interval variables, related to each other and possibly to the $\sharp$ variable in a way defined by the set of *temporal constraints*. To evaluate a concept at an interval $X$, different from the current one, it is necessary to temporally qualify it at $X$ – written $C@X$; in this way, every occurrence of $\sharp$ embedded within the concept expression $C$ is interpreted as the $X$ variable[3]. The informal meaning of a concept with a temporal existential quantification can be understood with the following examples, defining the `Basic-Open` elementary action:

$$\begin{aligned}\texttt{Basic-Open} \doteq\ & \texttt{Open-Door} \sqcap \\ & \Diamond(x\ y)\ (x\ \mathsf{o}\ \sharp)(x\ \mathsf{b}\ y)(\sharp\ \mathsf{o}\ y).\ \big((\star\texttt{DOOR} : \neg\texttt{Opened})@x \sqcap (\star\texttt{DOOR} : \texttt{Opened})@y\big)\end{aligned}$$

Figure 5 shows the temporal dependencies of the intervals in which the concept `Basic-Open` holds. `Basic-Open` denotes, according to the definition (a terminological axiom), any action occurring at some interval involving a $\star$`DOOR` that was once *closed* (i.e., $\neg$`Opened`) and then `Opened` as an effect of the actuation of the elementary action `Open-Door`. The $\sharp$ interval could be understood as the occurring time of the action type being defined: referring to it within the definition is an explicit way to temporally relate states and actions occurring in the world with respect to the occurrence of the action itself. The temporal constraints $(x\ \mathsf{o}\ \sharp)$, $(x\ \mathsf{b}\ y)$ and $(\sharp\ \mathsf{o}\ y)$ state that the interval denoted by $x$ *overlaps* the interval

3. Since any concept is implicitly temporally qualified at the special $\sharp$ variable, it is not necessary to explicitly qualify concepts at $\sharp$.

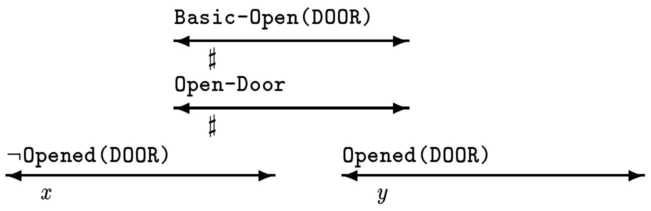

Figure 5: Temporal dependencies in the definition of the `Basic-Open` action.

denoted by ♯ – the occurrence interval of the action type `Basic-Open` – and is *before* $y$, while ♯ *overlaps* $y$. The parametric feature $\star$`DOOR` plays the role of *formal* parameter of the action, mapping any individual action of type `Basic-Open` to the actual door to be opened, independently from time. Please note that, whereas the existence and identity of the $\star$`DOOR` is time invariant, it can be qualified differently in different intervals of time, e.g., the $\star$`DOOR` is necessarily `Opened` only during the interval denoted by $y$.

An assertion of the type `Basic-Open`$(i, a)$ states that the individual $a$ is an action of the type `Basic-Open` occurred at the interval $i$. Moreover, the same assertion implies that $a$ is also of type `Open-Door` at the occurrence time $i$, and it is related to a $\star$`DOOR` , say $d$, which is ¬`Opened` at some interval $j$, overlapping $i$, and `Opened` at another interval $l$, overlapped by $i$ and after $j$.

$$\texttt{Basic-Open}(i,a) \Rightarrow \texttt{Open-Door} \wedge \exists d.\ \star\texttt{DOOR}(a,d) \wedge \exists j,l.\ (\neg\texttt{Opened}(j,d) \wedge \texttt{Opened}(l,d) \wedge \mathsf{o}(j,i) \wedge \mathsf{o}(i,l) \wedge \mathsf{a}(l,j))$$

An individual action is an object in the conceptual domain associated with the relevant properties – or states – of the world affected by the individual action itself via a collection of *features*; moreover, temporal relations constrain time intervals imposing an ordering in the change of the states of the world.

## 4. Representing the Robotic Domain

In the following Section examples of action, plan and state representations are introduced with the aim of showing the applicability of our framework.

A state is a collection of properties of world objects holding at a certain time. An action description represents how the world state may evolve in relation with the possible occurrence of the action itself. We assume that elementary actions directly correspond to robot executable actions, and that states may be sensed by the robot thru time. A plan is a complex action: it is described by means of temporally related world states and simpler actions. Plans are always grounded on a set of elementary executable actions.

### 4.1 Representing the domain plans

Let us introduce the plan for crossing a studio door:

$$\texttt{Cross-Studio} \doteq \Diamond x\ (x\ \mathsf{o}\ \sharp).\ (\texttt{Open-Studio}@x \sqcap \texttt{Move})$$

Figure 6 shows the temporal dependencies of the intervals in which the concept `Cross-Studio` holds. The definition employs the ♯ interval to denote the occurrence time of the plan itself; in this way, it is possible to describe how different actions or states of the world appear-

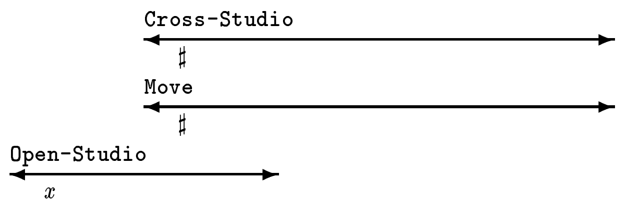

Figure 6: Temporal dependencies in the definition of the `Cross-Studio` plan.

ing in the definition of the plan are related to it. In the `Cross-Studio` example the `Move` action should take place at the same time of plan itself, while the action which opens the studio door must overlap it. To move inside the room the robot must hold open the door while it moves inside, so the two actions `Open-Studio` and `Move` must overlap. We assume that the `Open-Studio` action is a specific kind of `Basic-Open` (i.e., `Open-Studio` is subsumed by `Basic-Open`, written `Open-Studio`$\sqsubseteq$`Basic-Open`), since additionally the robot must check for a permission prior to execute the `Basic-Open` elementary action in order to open the door of a studio. According to the given definition, since $\sharp$ is simultaneous to the occurring time of `Move`, every occurrence of `Cross-Studio` is also a `Move` occurence, i.e., it can be deduced that crossing a studio is a more specific plan than the one for moving (`Cross-Studio`$\sqsubseteq$`Move`).

The definition of a plan can be reused within the definition of other plans by exploiting the full compositionality of the language. The `Cross-Studio` plan defined above is used in the definition of `Deliver-In-Studio`:

$$\begin{aligned}\texttt{Deliver-In-Studio} \doteq\ & \Diamond(v\ w\ z)\ (v\ \mathsf{s}\ \sharp)(v\ \mathsf{m}\ w)(w\ \mathsf{b}\ z)(z\ \mathsf{f}\ \sharp).\\ & (\texttt{Open-Request}@v \sqcap\\ & \ \texttt{Cross-Studio}@w \sqcap\\ & \ \texttt{Release}@z)\end{aligned}$$

In this case, precise temporal relations between the nodes of two corresponding temporal constraint networks are asserted: e.g., the action `Release` takes place strictly after the `Move` action of the `Cross-Studio` plan (see figure 7, where the plan `Cross-Studio` is expanded).

A plan subsuming `Deliver-In-Studio` is the more general plan defined below, `Deliver`, supposing that the action `Cross` subsumes `Cross-Studio`. This means that among all the individual actions of the type `Deliver` there are all the individual actions of type `Deliver-In-Studio`:

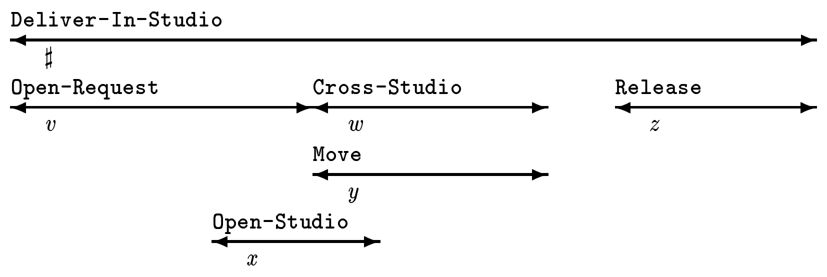

Figure 7: Temporal dependencies in the definition of `Deliver-In-Studio`.

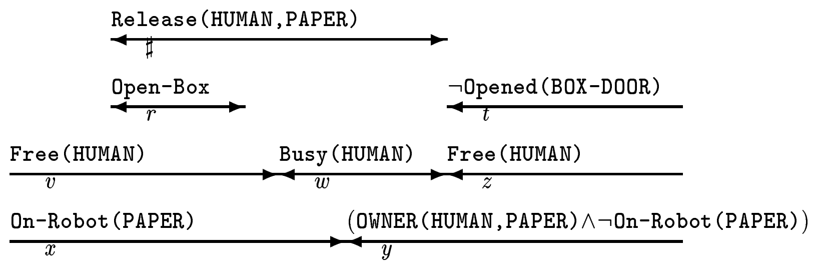

Figure 8: Temporal dependencies in the definition of the `Release` action.

$\texttt{Deliver} \doteq \Diamond\ (w\ z)\ (w\ (\mathsf{s},\mathsf{d})\ \sharp)(w\ \mathsf{b}\ z)(z\ \mathsf{f}\ \sharp).\ (\texttt{Cross}@w \sqcap \texttt{Release}@z)$

Let us now consider a different definition for the `Deliver` plan:

$\texttt{Deliver1} \doteq \Diamond\ (w\ z)\ (w\ (\mathsf{s})\ \sharp)(w\ \mathsf{b}\ z)(z\ \mathsf{f}\ \sharp).\ (\texttt{Cross}@w \sqcap \texttt{Release}@z)$

then `Deliver1` does not subsume anymore `Deliver-In-Studio` because the temporal networks have incomparable temporal relations with respect to the temporal variable $w$. In fact, from the above definitions we can derive:

$\texttt{Deliver1} \sqsubseteq \texttt{Deliver}$
$\texttt{Deliver-In-Studio} \sqsubseteq \texttt{Deliver}$

bu not:

$\texttt{Deliver-In-Studio} \sqsubseteq \texttt{Deliver1}$

## 4.2 Representing the domain actions

As a further example of the expressive power of the temporal language, it is now shown how to represent the `Release` action. In our domain a releasing action involves a *paper* and a *human being* which should own the paper. The action is successfull if the human being is not busy, the paper is inside the robot box and the robot actuates the elementary action that opens the box; at the end, the paper is owned by the human being (Figure 8).

Action parameters are represented by means of partial functions mapping from the action itself to the involved action parameter. In the language, these functions are called *parametric features*. For example, the action `Release` has the parameters ⋆`HUMAN` and ⋆`PAPER`, representing in some sense the objects that are involved in the action independently from time. So, in the assertion "⋆PAPER($a$,*paper*1)", *paper*1 denotes the requested paper involved in the action $a$ at any interval. On the other hand, an assertion involving a (non-parametric) feature, e.g., "OWNER($i$,*paper*1,*john*)", does not imply anything about the truth value at intervals other than $i$.

The concept expression, which defines the `Release` action, makes use of temporal qualified concept expressions, including feature *selections* and *agreements*: The expression (⋆PAPER : On-Robot)@$x$ means that the paper should be inside the robot box at the interval denoted by $x$; while (⋆PAPER∘OWNER ↓ ⋆HUMAN)@$y$ indicates that at the interval $y$ the ⋆`HUMAN` possesses the ⋆`PAPER`. The formal definition of the action `Release` is:

$$\begin{aligned}\texttt{Release} \doteq\ & \Diamond(x\ y\ r\ t\ v\ w\ z)\ (x\ \mathsf{m}\ y)(x\ \mathsf{o}\ w)(r\ \mathsf{s}\ \sharp)(r\ \mathsf{b}\ w)(t\ \mathsf{mi}\ \sharp)\\ & (v\ \mathsf{o}\ \sharp)(v\ \mathsf{m}\ w)(w\ \mathsf{f}\ \sharp)(z\ \mathsf{mi}\ \sharp).\\ & ((\star\texttt{PAPER} : \texttt{On-Robot})@x \sqcap\\ & \ (\star\texttt{PAPER} : \neg\texttt{On-Robot} \sqcap \star\texttt{PAPER}\circ\texttt{OWNER} \downarrow \star\texttt{HUMAN})@y \sqcap\\ & \ \texttt{Open-Box}@r \sqcap (\star\texttt{BOX-DOOR} : \neg\texttt{Opened})@t \sqcap\\ & \ (\star\texttt{HUMAN} : \texttt{Free})@v \sqcap (\star\texttt{HUMAN} : \texttt{Busy})@w \sqcap (\star\texttt{HUMAN} : \texttt{Free})@z)\end{aligned}$$

The above defined concept does not state which properties are the prerequisites for the releasing action or which properties must be true whenever the action succeeds. The definition states that whenever the human is free and the paper is into the robot box the robot can carry out the opening of the box which starts the release action itself. After some human activity the paper is no more into the robot box but it is owned by the human. The end of the release action is set by the closure of the robot box (an action executed by the human during its activity). Note that the world states described at the intervals denoted by $v, w, z$ are the result of an action which is not under the robot controls that has been generically called `Human-Activity`:

$$\begin{aligned}\texttt{Human-Activity} \doteq\ & \Diamond(x\ w\ z)\ \ (x\ \mathsf{o}\ \sharp)(x\ \mathsf{m}\ w)(w\ \mathsf{f}\ \sharp)(z\ \mathsf{mi}\ \sharp).\\ & ((\star\texttt{HUMAN} : \texttt{Free})@x \sqcap\ (\star\texttt{HUMAN} : \texttt{Busy})@w \sqcap\\ & \ (\star\texttt{HUMAN} : \texttt{Free})@z)\end{aligned}$$

The `Release` action can be redefined by making use of the `Human-Activity` action definition:

$$\begin{aligned}\texttt{Release} \doteq\ & \Diamond(x\ y\ r\ t\ v\ u)\ (x\ \mathsf{m}\ y)(x\ \mathsf{o}\ u)(r\ \mathsf{s}\ \sharp)(r\ \mathsf{b}\ u)(t\ \mathsf{mi}\ \sharp)(v\ \mathsf{o}\ \sharp)(v\ (\mathsf{s},\mathsf{d},\mathsf{o})\ x)(u\ \mathsf{f}\ \sharp).\\ & ((\star\texttt{PAPER} : \texttt{On-Robot})@x \sqcap\\ & \ (\star\texttt{PAPER} : \neg\texttt{On-Robot} \sqcap \star\texttt{PAPER}\circ\texttt{OWNER} \downarrow \star\texttt{HUMAN})@y \sqcap\\ & \ \texttt{Open-Box}@r \sqcap (\star\texttt{BOX-DOOR} : \neg\texttt{Opened})@t \sqcap\\ & \ (\texttt{Human-Activity}[x]@v)@u)\end{aligned}$$

The temporal substitutive qualifier (`Human-Activity`$[x]@v$) *renames* the variable $x$ to $v$ within the defined `Human-Activity` action. It has to be understood as a way of making coreference between two temporal variables introduced by different temporal networks, while the renamed variable ($x$ in this example) inherites the temporal constraints of the substituting interval ($v$ in this case). Furthermore, the effect of temporally qualifying the human activity at $u$ is that the $\sharp$ variable associated to the human activity – referring to the occurrence time of the action itself – is bound to the interval denoted by $u$. Because of this binding on the occurrence time of the human activity, the $\sharp$ variable in the `Release` action and the $\sharp$ variable in the `Human-Activity` action denote different time intervals, so that the human activity occurs at an interval finishing the occurrence time of the release action.

Now it is shown how action recognition can be performed from a series of observations of the world state sensed by the robot in the environment. This task is called *specific plan recognition with respect to a plan description.* The following ABox describes a situation in which papers can be into the robot box, or owned by a person, while a person can be free or he can be doing some activity, and in which some unknown generic individual action $a$ is taking place at time interval $i_a$ having *john* and *paper1* as its actual parameters:

$\star\texttt{HUMAN}(a, john),\ \star\texttt{PAPER}(a, paper1),\ \star\texttt{BOX-DOOR}(a, bd)$

$\mathsf{o}(i_1, i_a),\ \texttt{On-Robot}(i_1, paper1),$

$\mathsf{mi}(i_2, i_1),\ \neg\texttt{On-Robot}(i_2, paper1),\ \texttt{OWNER}(i_2, paper1, john)$

$\mathsf{s}(i_3, i_a),\ \mathsf{b}(i_3, i_6),\ \texttt{Open-Box}(i_3, a),$

$\mathsf{mi}(i_4, i_a),\ \neg\texttt{Opened}(i_4, bd),$

$\mathsf{o}(i_5, i_a),\ \mathsf{m}(i_5, i_6),\ \texttt{Free}(i_5, john),$

$\mathsf{f}(i_6, i_a),\ \texttt{Busy}(i_6, john),$

$\mathsf{mi}(i_7, i_a),\ \texttt{Free}(i_7, john).$

In the context of a knowledge base $\Sigma$ composed by the above ABox and the definitions of the `Release` and `Human-Activity` concepts in the TBox as given above, the system deduces that, the individual action $a$ is of type `Release` at the time interval $i_a$, i.e.,

$\Sigma \models \texttt{Release}(i_a, a)$,

and that

$\Sigma \models \texttt{Human-Activity}(i_b, a)$

for an $i_b$ such that $\mathsf{f}(i_6, i_b)$. Thus, the robot observations of the world are explained by means of understanding the kind of actions which are occurring.

### 4.3 Representing the domain world states

States of the world in $\mathcal{TL}$-$\mathcal{ALCF}$ are described by using the expressive but still decidable description language $\mathcal{ALCF}$.

In our domain, the fact that every location is a room or a corridor which are in turn disjoint concepts is captured by the following definitions:

$\texttt{Location} \doteq (\texttt{Room} \sqcup \texttt{Corridor})$
$\texttt{Room} \mathrel{\dot{\leq}} \neg\texttt{Corridor}$

A door is described as something that connects two different locations:

$\texttt{Door} \doteq \star\texttt{INSIDE} : \texttt{Location} \sqcap \star\texttt{OUTSIDE} : \texttt{Location} \sqcap \star\texttt{INSIDE} \uparrow \star\texttt{OUTSIDE}$

Note that we used parametric features in the above definition to enforce that each of the connecting locations of a door remains the same along time. Finally, a studio door is a kind of door that connects a room with a corridor:

$\texttt{Studio-Door} \doteq \texttt{Door} \sqcap \star\texttt{INSIDE} : \texttt{Room} \sqcap \star\texttt{OUTSIDE} : \texttt{Corridor}$

## 5. Related Works

The original formalism devised by Allen (Allen, 1991) forms, in its very basis, the foundation for our work. It is a predicate logic in which interval temporal networks can be introduced, properties can be asserted to *hold* over intervals, and events can be said to *occur* at intervals. His approach is very general, but it suffers from problems related to the semantic formalization of the predicates `hold` and `occur` (Blackburn, 1992). Moreover, computational properties of the formalism are not analyzed. The study of this latter aspect was, on the contrary, our main concern.

In the Description Logic literature, other approaches for representing and reasoning with time and action were proposed. In the beginning the approaches based on an explicit notion of time are surveyed, and then the Strips-like approaches. This Section ends by illustrating some of the approaches devoted to temporally extend the situation calculus.

Weida and Litman (1992, 1994) propose T-Rex, a loose hybrid integration between Description Logics and constraint networks. Plans are defined as collections of steps together with temporal constraints between their duration. Each step is associated with an action type, represented by a generic concept in K-Rep – a non-temporal Description Logics. Thus a plan is seen as a *plan network*, a temporal constraint network whose nodes, corresponding to time intervals, are labeled with action types and are associated with the steps of the plan itself. A structural plan subsumption algorithm is defined, characterized in terms of graph matching, and based on two separate notions of subsumption: pure terminological subsumption between action types labeling the nodes, and pure temporal subsumption between interval relationships labeling the arcs. The plan library is used to guide plan recognition (Weida, 1995, 1996) in a way similar to that proposed in (Kautz, 1991). Even if this work has strong motivations, no formal semantics is provided for the language and the reasoning problems.

There are Description Logics intended to represent and reasoning about actions following the Strips tradition. Heinsohn, Kudenko, Nebel and Profitlich (Heinsohn, Kudenko, Nebel, & Profitlich, 1992) describe the Rat system, used in the Wip project at the German Research Center for AI (DFKI). They use a Description Logic to represent both the world states and atomic actions. A second formalism is added to compose actions in plans and to reason about simple temporal relationships. No explicit temporal constraints can be expressed in the language. Rat actions are defined by the change of the world state they cause, and they are instantaneous as in the Strips-like systems, while plans are linear sequences of actions. The most important service offered by Rat is the *simulated execution* of part of a plan, checking if a given plan is *feasible* and, if so, computing the global pre- and post-conditions. The feasibility test is similar to the usual consistency check for a concept description: they *temporally project* the pre- and post-conditions of individual actions composing the plan, respectively backward and forward. If this does not lead to an inconsistent initial, final or intermediate state, the plan is feasible and the global pre- and post-conditions are determined as a side effect.

Devambu and Litman (Devanbu & Litman, 1991, 1996) describe the Clasp system, a *plan-based* knowledge representation system extending the notion of subsumption and classification to plans, to build an efficient information retrieval system. In particular, Clasp was used to represent plan-like knowledge in the domain of telephone switching software by extending the use of the software information system LASSIE (Devanbu, Brachman, Selfridge, & Ballard, 1991). Clasp is designed for representing and reasoning about large collections of plan descriptions, using a language able to express temporal, conditional and looping operators. Following the Strips tradition, plan descriptions are built starting from states and actions, both represented by using the Classic (Brachman, McGuiness, Patel-Schneider, Resnick, & Borgida, 1991) terminological language. Since plans constructing operators correspond to regular expressions, algorithms for subsumption integrate work in automata theory with work in concept subsumption. The temporal expressive power of this system is limited to sequences, disjunction and iterations of actions and each action is instantaneous. Furthermore, state descriptions are restricted to a simple conjunction of primitive CLASSIC concepts. Like RAT, CLASP checks if an instantiated plan is well formed, i.e., the specified sequence of individual actions are able to transform the given initial state into the goal state by using the STRIPS rules.

We briefly report now on the efforts made by researchers in the situation calculus field to overcome the strict sequential perspective inherent to this framework. Recent works enrich the original framework to represent properties and actions having different truth values depending not only on the situation but also on time. The work of Reiter (Reiter, 1996), moving from the results showed by Pinto (Pinto, 1994) and by Ternovskaia (Ternovskaia, 1994), provides a new axiomatization of the situation calculus able to capture concurrent actions, properties with continuous changes, and natural exogenous actions – those under nature's control. The notion of *fluent* – which models properties of the world – and *situation* are maintained. Each action is instantaneous and responsible for changing the actual situation to the subsequent one. Concurrent actions are simply sets of instantaneous actions that must be coherent, i.e., the action's collection must be non empty and all the actions occur at the same time. Pinto (Pinto, 1994) and Reiter (Reiter, 1996) introduce the time dimension essentially to capture both the occurrence of the natural actions, due to known laws of physics – i.e., the ball bouncing at times prescribed by motion's equations – and the dynamic behavior of physical objects – i.e., the position of a falling ball. This is realized by introducing a time argument for each action function, while properties of the world are divided into two different classes: classical fluents that hold or do not hold throughout situations, and *continuous parameters* that may change their value during the time spanned by the given situation.

More devoted to have a situation calculus with a time interval ontology is the work of Ternovskaia (Ternovskaia, 1994). In order to describe *processes* – i.e., actions extended in time – she introduces durationless actions that initiate and terminate those processes. As a matter of fact, processes become fluents, with instantaneous events – *Start(Fluent)* and *Finish(Fluent)* – which respectively make true or false the corresponding fluent, and with persistence assumptions that make the fluent true during the interval. For example, in a blocks world the *picking-up* process is treated as a fluent with *Start(picking-up(x))* and *Finish(picking-up(x))* instantaneous actions that enable or falsify the *picking-up* fluent.

## 6. Conclusions

The main objective of this paper was showing the usefulness of a logical formalism for representing actions and plans in a robotic domain. According to this framework, an action has a duration in time, it can have parameters, which are the ties with the temporal evolution of the world, and it is possibly associated over time with other actions. The peculiar computational properties of this logic make it an effective representation and reasoning tool for plan recognition purposes. An action/plan taxonomy based on subsumption can be set up, and it can play the role of a action/plan library for action/plan retrieval tasks.

We are studying several extensions to the basic temporal language. With the possibility to specify homogeneous predicates the temporal behavior of world states can be described in a more natural way, while the introduction of the non-monotonic inertial operator gives rise to some forms of temporal prediction (Artale & Franconi, 1998). Another extension (Artale

$$
\begin{array}{rcl}
(\mathsf{s})^{\mathcal{E}} & = & \{\langle [u,v],[u_1,v_1]\rangle \in \mathcal{T}^\star_< \times \mathcal{T}^\star_< \mid u = u_1 \wedge v < v_1\} \\
(\mathsf{f})^{\mathcal{E}} & = & \{\langle [u,v],[u_1,v_1]\rangle \in \mathcal{T}^\star_< \times \mathcal{T}^\star_< \mid v = v_1 \wedge u_1 < u\} \\
(\mathsf{mi})^{\mathcal{E}} & = & \{\langle [u,v],[u_1,v_1]\rangle \in \mathcal{T}^\star_< \times \mathcal{T}^\star_< \mid u = v_1\} \\
 & \ldots & \textit{(meaning of the other Allen temporal relations)} \\
(R\ ,\ S)^{\mathcal{E}} & = & R^{\mathcal{E}} \cup S^{\mathcal{E}} \\
\langle \overline{X}, \overline{\mathit{Tc}}\rangle^{\mathcal{E}} & = & \{\mathcal{V} : \overline{X} \mapsto \mathcal{T}^\star_< \mid\ \forall (X\ (R)\ Y) \in \overline{\mathit{Tc}}.\ \langle \mathcal{V}(X), \mathcal{V}(Y)\rangle \in (R)^{\mathcal{E}}\}
\end{array}
$$

Figure 9: The temporal interpretation function.

& Franconi, 1995) deals with the possibility of relating an action to more elementary actions, *decomposing* it in partially ordered steps; this kind of reasoning is found in hierarchical planners like NONLIN (Tate, 1977), SIPE (Wilkins, 1988) and FORBIN (Dean, Firby, & Miller, 1990).

## Acknowledgments

This work was partially supported by the Italian National Research Council (CNR) project "Ontologic and Linguistic Tools for Conceptual Modeling", and by the "Foundations of Data Warehouse Quality" (*DWQ*) European ESPRIT IV Long Term Research (LTR) Project 22469. The second author wishes to thank **ITC-irst** for having supported him during the preparation of this paper.

## Appendix A. Semantics

In this Section, a Tarski-style extensional semantics for the $\mathcal{TL}$-$\mathcal{ALCF}$ language is given, and a formal definition of the subsumption and recognition reasoning tasks is devised.

Assume a linear, unbounded, and dense temporal structure $\mathcal{T} = (\mathcal{P}, <)$, where $\mathcal{P}$ is a set of time points and $<$ is a strict partial order on $\mathcal{P}$. In such a structure, given an interval $X$ and a temporal relation $R$, it is always possible to find an interval $Y$ such that $(X\ (R)\ Y)$. The assumption of linear time – which means that for any two points $t_1$ and $t_2$ such that $t_1 \leq t_2$ the set of points $\{t \mid t_1 \leq t \leq t_2\}$ is totally ordered – fits the intuition about the nature of time, so that the pair $[t_1, t_2]$ can be thought as the closed interval of points between $t_1$ and $t_2$. The *interval set* of a structure $\mathcal{T}$ is defined as the set $\mathcal{T}^\star_<$ of all closed intervals $[u, v] \doteq \{x \in \mathcal{P} \mid u \leq x \leq v, u \neq v\}$ in $\mathcal{T}$.

A *primitive interpretation* $\mathcal{I} \doteq \langle \mathcal{T}^\star_<, \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ consists of a set $\mathcal{T}^\star_<$ (the *interval set* of the selected temporal structure $\mathcal{T}$), a set $\Delta^{\mathcal{I}}$ (the *domain* of $\mathcal{I}$), and a function $\cdot^{\mathcal{I}}$ (the *primitive interpretation function* of $\mathcal{I}$) which gives a meaning to atomic concepts, roles, features and parametric features:

$$A^{\mathcal{I}} \subseteq \mathcal{T}^\star_< \times \Delta^{\mathcal{I}} \qquad P^{\mathcal{I}} \subseteq \mathcal{T}^\star_< \times \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \qquad f^{\mathcal{I}} : (\mathcal{T}^\star_< \times \Delta^{\mathcal{I}}) \overset{\textit{partial}}{\longmapsto} \Delta^{\mathcal{I}} \qquad \star g^{\mathcal{I}} : \Delta^{\mathcal{I}} \overset{\textit{partial}}{\longmapsto} \Delta^{\mathcal{I}}$$

Atomic parametric features are interpreted as partial functions; they differ from atomic features for being independent from time.

In order to give a meaning to temporal expressions present in generic concept expressions, figure 9 defines the *temporal interpretation function.* The *temporal interpretation function* $\cdot^{\mathcal{E}}$ depends only on the temporal structure $\mathcal{T}$. The labeled directed graph $\langle \overline{X}, \overline{Tc} \rangle$ – where $\overline{X}$ is the set of variables representing the nodes, and $\overline{Tc}$ is the set of temporal constraints representing the arcs – is called *temporal constraint network.* The interpretation of a temporal constraint network is a set of variable assignments that satisfy the temporal constraints. A *variable assignment* is a function $\mathcal{V} : \overline{X} \mapsto \mathcal{T}^{\star}_{<}$ associating an interval value to a temporal variable. A temporal constraint network is *consistent* if it admits a non empty interpretation. The notation, $\langle \overline{X}, \overline{Tc} \rangle^{\mathcal{E}}_{\{x_1 \mapsto t_1, x_2 \mapsto t_2, \ldots\}}$, used to interpret concept expressions, denotes the subset of $\langle \overline{X}, \overline{Tc} \rangle^{\mathcal{E}}$ where the variable $x_i$ is mapped to the interval value $t_i$.

It is now possible to interpret generic concept expressions. Consider the equations introduced in Figure 10. An *interpretation function* $\cdot^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$, based on a variable assignment $\mathcal{V}$, an interval $t$ and a set of constraints $\mathcal{H} = \{x_1 \mapsto t_1, \ldots\}$ over the assignments of inner variables, extends the primitive interpretation function in such a way that the equations of Figure 10 are satisfied – we do not report the constructors that can be obtaind by complementation. Intuitively, the interpretation of a concept $C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$ is the set of entities of the domain that are of type $C$ at the time interval $t$, with the assignment for the free temporal variables in $C$ given by $\mathcal{V}$ – see $(C@X)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$ – and with the constraints for the assignment of variables in the scope of the outermost temporal quantifiers given by $\mathcal{H}$. Note that, $\mathcal{H}$ interprets the variable renaming due to the temporal substitutive qualifier – see $(C[Y]@X)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$ – and it takes effect during the choice of a variable assignment, as the equation $(\Diamond(\overline{X})\ \overline{Tc}.\ C)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$ shows.

In absence of free variables in the concept expression – with the exception of $\sharp$– for notational simplification the *natural* interpretation function $C^{\mathcal{I}}_{t}$, being equivalent to the interpretation function $C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}$ with any $\mathcal{V}$ such that $\mathcal{V}(\sharp) = t$ and $\mathcal{H} = \emptyset$ is introduced. The set of interpretations $\{C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}\}$ obtained by varying $\mathcal{I}, \mathcal{V}, t$ with a fixed $\mathcal{H}$ is maximal wrt set inclusion if $\mathcal{H} = \emptyset$, i.e., the set of natural interpretations includes any set of interpretations with a fixed $\mathcal{H}$. In fact, since $\mathcal{H}$ represents a constraint in the assignment of variables, the unconstrained set is the larger one. Note that, for feature interpretation only the natural one is used since it is not admitted to temporally qualify them.

An interpretation $\mathcal{I}$ satisfies the terminological (TBox) axiom $A \doteq C$ iff $A^{\mathcal{I}}_{t} = C^{\mathcal{I}}_{t}$, for every $t$. A concept $C$ is *subsumed* by a concept $D$ ($C \sqsubseteq D$) if $C^{\mathcal{I}}_{t} \subseteq D^{\mathcal{I}}_{t}$ for every interpretation $\mathcal{I}$ and every interval $t$. An interpretation $\mathcal{I}$ is a *model* for a concept $C$ if $C^{\mathcal{I}}_{t} \neq \emptyset$ for some $t$. If a concept has a model, then it is *satisfiable*, otherwise it is *unsatisfiable.*

It is interesting to note that only the relations s, f, mi are really necessary, because it is possible to express any temporal relationship between two distinct intervals using only these three relations and their transpositions si, fi, m (Halpern & Shoham, 1991). The following equivalences hold:

$$\begin{aligned}
\Diamond x\ (x\ \mathsf{a}\ \sharp).\ C@x &\equiv \Diamond xy\ (y\ \mathsf{mi}\ \sharp)(x\ \mathsf{mi}\ y).\ C@x \\
\Diamond x\ (x\ \mathsf{d}\ \sharp).\ C@x &\equiv \Diamond xy\ (y\ \mathsf{s}\ \sharp)(x\ \mathsf{f}\ y).\ C@x \\
\Diamond x\ (x\ \mathsf{o}\ \sharp).\ C@x &\equiv \Diamond xy\ (y\ \mathsf{s}\ \sharp)(x\ \mathsf{fi}\ y).\ C@x
\end{aligned}$$

$$
\begin{array}{rcl}
A^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \{a \in \Delta^{\mathcal{I}} \mid \langle t, a\rangle \in A^{\mathcal{I}}\} \quad = \quad A^{\mathcal{I}}_t \\
\top^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \Delta^{\mathcal{I}} \quad = \quad \top^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} \\
(C \sqcap D)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} \cap D^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} \\
(\forall P.C)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \{a \in \Delta^{\mathcal{I}} \mid \forall b.(a,b) \in P^{\mathcal{I}}_t \Rightarrow b \in C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}\} \\
(p \downarrow q)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \{a \in dom\, p^{\mathcal{I}}_t \cap dom\, q^{\mathcal{I}}_t \mid p^{\mathcal{I}}_t(a) = q^{\mathcal{I}}_t(a)\} \quad = \quad (p \downarrow q)^{\mathcal{I}}_t \\
(p : C)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \{a \in dom\, p^{\mathcal{I}}_t \mid p^{\mathcal{I}}_t(a) \in C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}}\} \\
(C@X)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & C^{\mathcal{I}}_{\mathcal{V},\mathcal{V}(X),\mathcal{H}} \\
(C[Y]@X)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & C^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}\cup\{Y \mapsto \mathcal{V}(X)\}} \\
(\Diamond(\overline{X})\ \overline{Tc}.\ C)^{\mathcal{I}}_{\mathcal{V},t,\mathcal{H}} & = & \{a \in \Delta^{\mathcal{I}} \mid \exists \mathcal{W}.\ \mathcal{W} \in \langle \overline{X}, \overline{Tc}\rangle^{\mathcal{E}}_{\mathcal{H}\cup\{\sharp \mapsto t\}} \wedge a \in C^{\mathcal{I}}_{\mathcal{W},t,\emptyset}\} \\
P^{\mathcal{I}}_t & = & \hat{P}_t \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid\ \forall a, b.\ \langle a, b\rangle \in \hat{P}_t \leftrightarrow \langle t, a, b\rangle \in P^{\mathcal{I}} \\
f^{\mathcal{I}}_t & = & \hat{f}_t : \Delta^{\mathcal{I}} \overset{partial}{\longmapsto} \Delta^{\mathcal{I}} \mid\ \forall a.\ (a \in dom\, \hat{f}_t \leftrightarrow \langle t, a\rangle \in dom\, f^{\mathcal{I}}) \wedge \\
 & & \qquad\qquad\qquad\qquad\qquad \hat{f}_t(a) = f^{\mathcal{I}}(t, a) \\
(p \circ q)^{\mathcal{I}}_t & = & p^{\mathcal{I}}_t \circ q^{\mathcal{I}}_t \\
\star g^{\mathcal{I}}_t & = & \star g^{\mathcal{I}}
\end{array}
$$

Figure 10: The interpretation function.

To assign a meaning to ABox axioms, the temporal interpretation function $\cdot^{\mathcal{E}}$ is extended to temporal intervals so that $i^{\mathcal{E}}$ is an element of $\mathcal{T}^{\star}_{<}$ for each $i \in \mathcal{OT}$. The semantics of assertions is the following: $C(i, a)$ is satisfied by an interpretation $\mathcal{I}$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}_{i^{\mathcal{E}}}$; $P(i, a, b)$ is satisfied by $\mathcal{I}$ iff $\langle i^{\mathcal{E}}, a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in P^{\mathcal{I}}$; $p(i, a, b)$ is satisfied by $\mathcal{I}$ iff $p^{\mathcal{I}}_{i^{\mathcal{E}}}(a^{\mathcal{I}}) = b^{\mathcal{I}}$; $\star g(a, b)$ is satisfied by $\mathcal{I}$ iff $\star g^{\mathcal{I}}(a^{\mathcal{I}}) = b^{\mathcal{I}}$; and $R(i, j)$ is satisfied by $\mathcal{I}$ iff $\langle i^{\mathcal{E}}, j^{\mathcal{E}}\rangle \in R^{\mathcal{E}}$. Given a knowledge base $\Sigma$, an individual $a$ in $\mathcal{O}$ is said to be an *instance* of a concept $C$ *at the interval* $i$ if $\Sigma \models C(i, a)$.

Now we are able to give a semantic definition for the reasoning task we called *specific plan recognition with respect to a plan description*. This is an inference service that computes if an *individual action/plan* is an instance of an *action/plan type* at a certain interval, i.e., the task known as instance recognition in the description logic community. Given a knowledge base $\Sigma$, an interval $i$, an individual $a$ and a concept $C$, the *instance recognition problem* is to test whether $\Sigma \models C(i, a)$.

## Appendix B. The Calculus for $\mathcal{TL}$-$\mathcal{ALCF}$

Artale and Franconi (1998) contribute to explore the decidable realm of interval-based temporal description logics by presenting sound, complete and terminating procedures for subsumption reasoning. The key idea in order to obtain decidable languages is the restriction posed on the temporal expressivity by eliminating the universal quantification on temporal variables. The main results are proven starting with the simplest language, $\mathcal{TL}$-$\mathcal{F}$, where $\mathcal{F}$ is feature language with neither negation nor disjunction and lacking roles, too.

Then, the authors show how to reason with more expressive languages such as $\mathcal{TLU}$-$\mathcal{FU}$, which adds disjunction both at the temporal and non-temporal sides of the language, and $\mathcal{TL}$-$\mathcal{ALCF}$. The subsumption algorithms are based on a *normalization procedure*, i.e., an interpretation preserving transformation which operates a separation between the temporal and the non-temporal part of the formalism. A concept in normal form can be seen as a *conceptual temporal constraint network* , i.e., a labeled directed graph $\langle \overline{X}, \overline{\mathcal{T}c}, \overline{Q@X} \rangle$ (in $\mathcal{TL}$-$\mathcal{ALCF}$ syntax: $\diamond(\overline{X})\ \overline{\mathcal{T}c}.\ (Q^0 \sqcap Q^1@X^1 \sqcap \ldots \sqcap Q^n@X^n))$ where arcs are labeled with a set of arbitrary temporal relationships – representing their disjunction, and temporal nodes are labeled with non-temporal concepts (i.e., each $Q^j$ is an $\mathcal{ALCF}$ concept expression). The subsumption procedure checks whether there is a mapping function between conceptual temporal constraint network (i.e., a form of subgraph isomorphism, which was called *s-mapping*) such that a subsumption relation holds both among the non-temporal concepts labeling the corresponding nodes in the mapping function, and among the temporal relations of the corresponding arcs. Then the calculus can adopt standard procedures developed both in the description logics community and in the temporal constraints community. Algorithms to compute subsumption between non-temporal concepts are well known and based on a notational variant of the first-order tableaux calculus (Schmidt-Schauß & Smolka, 1991; Hollunder & Nutt, 1990; Donini et al., 1995). A standard technique for computing subsumption between temporal networks relies on methods for checking the satisfiability of such networks (van Beek & Manchak, 1996). The following theorem summarizes the main results proved by Artale and Franconi:

**Theorem B.1** *Let $C_1$ and $C_2$ be $\mathcal{TL}$-$\mathcal{F}$ or $\mathcal{TL}$-$\mathcal{ALCF}$ concepts in normal form, then $C_1$ subsumes $C_2$ ($C_2 \sqsubseteq C_1$) if and only if there exists an s-mapping from $C_1$ to $C_2$.*

*Let $C = C_1 \sqcup \cdots \sqcup C_m$ and $D = D_1 \sqcup \cdots \sqcup D_n$ be $\mathcal{TLU}$-$\mathcal{FU}$ concepts in normal form; then $D$ subsumes $C$ ($C \sqsubseteq D$) if and only if $\forall i \exists j.\ C_i \sqsubseteq D_j$.*

*Concept subsumption between $\mathcal{TL}$-$\mathcal{F}$ or $\mathcal{TLU}$-$\mathcal{FU}$ concept expressions in normal form is an NP-complete problem; $\mathcal{TL}$-$\mathcal{ALCF}$ subsumption is a PSPACE-hard problem.*

## References

Allen, J. F. (1991). Temporal reasoning and planning. In Allen, J. F., Kautz, H. A., Pelavin, R. N., & Tenenberg, J. D. (Eds.), *Reasoning about Plans*, chap. 1, pp. 2–68. Morgan Kaufmann.

Artale, A., & Franconi, E. (1994). A computational account for a description logic of time and action. In J.Doyle, E.Sandewall, & P.Torasso (Eds.), *Proc. of the 4 th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 3–14 Bonn, Germany. Morgan Kaufmann.

Artale, A., & Franconi, E. (1995). Hierarchical plans in a description logic of time and action. In A.Borgida, M.Lenzerini, D.Nardi, & B.Nebel (Eds.), *Workshop Notes of the Int. Workshop on Description Logics. DL-95*, pp. 1–5 Roma, Italy. Tech. Rep. 07.95. Also in the Workshop Notes of the IJCAI-95 Workshop on "The Next Generation of Plan Recognition Systems: Challanges for and Insight from Related Areas of AI", Montréal, 1995.

Artale, A., & Franconi, E. (1998). A temporal description logic for reasoning about actions and plans. *Journal of Artificial Intelligence Research*, *9*.

Blackburn, P. (1992). Fine grained theories of time. In *Working Papers of the* 4th *Intl. Workshop on* Semantics of Time, Space, Movement, and Spatio-Temporal Reasoning, pp. 299–320.

Brachman, R. J., McGuiness, D. L., Patel-Schneider, P. F., Resnick, L. A., & Borgida, A. (1991). Living with CLASSIC: When and how to use a KL-ONE-like language. In Sowa, J. (Ed.), *Principles of Semantic Networks*. Morgan Kaufmann.

Buchheit, M., Donini, F. M., & Schaerf, A. (1993). Decidable reasoning in terminological knowledge representation systems. *Information Systems*, *1*, 109–138.

De Giacomo, G., & Lenzerini, M. (1996). Tbox and abox reasoning in expressive description logics. In *Proc. of the 5 th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 316–327 Boston, MA. Morgan Kaufmann.

De Giacomo, G., & Lenzerini, M. (1995). What's in an aggregate: Foundations for description logics with tuples and sets. In *Proc. of the 13 th IJCAI* Montreal, Canada.

Dean, T., Firby, J., & Miller, D. (1990). Hierarchical planning involving deadlines, travel time and resources. *Computational Intelligence*, *6*(1).

Devanbu, P. T., & Litman, D. J. (1991). Plan-based terminological reasoning. In *Proc. of the 2 nd International Conference on Principles of Knowledge Representation and Reasoning*, pp. 128–138 Cambridge, MA.

Devanbu, P. T., & Litman, D. J. (1996). Taxonomic plan reasoning. *Artificial Intelligence*, *84*, 1–35.

Devanbu, P., Brachman, R., Selfridge, P., & Ballard, B. (1991). Taxonomic plan reasoning. *Communication of the ACM*, *34*(5).

Donini, F. M., Hollunder, B., Lenzerini, M., Spaccamela, A. M., Nardi, D., & Nutt, W. (1992). The complexity of existential quantification in concept languages. *Artificial Intelligence*, *53*, 309–327.

Donini, F. M., Lenzerini, M., Nardi, D., & Nutt, W. (1991). Tractable concept languages. In *Proc. of the 12 th IJCAI*, pp. 458–465 Sidney, Australia.

Donini, F. M., Lenzerini, M., Nardi, D., & Nutt, W. (1995). The complexity of concept languages. Tech. rep. RR-95-07, DFKI, Germany. A preliminary version appears in Proc. of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR-91).

Donini, F. M., Lenzerini, M., Nardi, D., & Schaerf, A. (1994). Deduction in concept languages: from subsumption to instance checking. *Journal of Logic and Computation*, *4*(4), 423–452.

Donini, F. M., & Era, A. (1992). Most specific concepts for knowledge bases with incomplete information. In *Proc. of CIKM-92*, pp. 545–551.

Fikes, R. E., & Nilsson, N. (1971). STRIPS: a new approach to the application of theorem proving as problem solving. *Artificial Intelligence*, *2*, 198–208.

Halpern, J. Y., & Shoham, Y. (1991). A propositional modal logic of time intervals. *Journal of ACM*, *38*(4), 935–962.

Heinsohn, J., Kudenko, D., Nebel, B., & Profitlich, H. (1992). RAT: representation of actions using terminological logics. Tech. rep., DFKI, Saarbrücken, Germany.

Hollunder, B., & Nutt, W. (1990). Subsumption algorithms for concept languages. Tech. rep. RR-90-04, DFKI, Germany.

Kabanza, F., Barbeau, M., & St-Denis, R. (1997). Planning control rules for reactive agents. *Artificial Intelligence*, *95*(1).

Kautz, H. A. (1991). A formal theory of plan recognition and its implementation. In Allen, J. F., Kautz, H. A., Pelavin, R. N., & Tenenberg, J. D. (Eds.), *Reasoning about Plans*, chap. 2, pp. 69–126. Morgan Kaufmann.

Lifschitz, V. (1987). On the semantics of STRIPS. In *The 1986 Workshop on Reasoning about Actions and Plans*, pp. 1–10. Morgan Kaufman.

McCarthy, J., & Hayes, P. J. (1969). Some philosophical problems from the standpoint of Artificial Intelligence. In Meltzer, B., & Michie, D. (Eds.), *Machine Intelligence*, Vol. 4, pp. 463–502 Edinburgh, UK. Edinburgh University Press.

Nebel, B. (1990). Terminological reasoning is inherently intractable. *Artificial Intelligence*, *43*, 235–249.

Nebel, B. (1991). Terminological cycles: Semantics and computational properties. In Sowa, J. F. (Ed.), *Principles of Semantic Networks*, chap. 11, pp. 331–362. Morgan Kaufmann.

Pinto, J. A. (1994). *Temporal Reasoning in the Situation Calculus.* Ph.D. thesis, Department of Computer Science, University of Toronto.

Reiter, R. (1996). Natural actions, concurrency and continuous time in the situation calculs. In *Proc. of the 5 th International Conference on Principles of Knowledge Representation and Reasoning* Boston, MA.

Sandewall, E., & Shoham, Y. (1994). Non-monotonic temporal reasoning. In Gabbay, D. (Ed.), *Handbook of Artificial Intelligence and Logic programming.* Oxford University Press.

Schaerf, A. (1994). Reasoning with individuals in concept languages. *Data & Knowledge Engineering*, *13*(2), 141–176.

Schmidt-Schauß, M., & Smolka, G. (1991). Attributive concept descriptions with complements. *Artificial Intelligence*, *48*(1), 1–26.

Schmiedel, A. (1990). A temporal terminological logic. In *Proc. of AAAI-90*, pp. 640–645 Boston, MA.

Tate, A. (1977). Generating project networks. In *Proc. of the 5 th IJCAI*, pp. 888–893 Cambridge, MA.

Ternovskaia, E. (1994). Interval situation calculus. In *Workshop Notes of the ECAI-94 Workshop "Logic and Change"*, pp. 153–164 Amsterdam.

van Beek, P., & Manchak, D. W. (1996). The design and experimental analysis of algorithms for temporal reasoning. *Journal of Artificial Intelligence Research*, *4*, 1–18.

Weida, R. (1995). Knowledge representation for plan recognition. In *Working Notes of the IJCAI-95 workshop "The Next Generation of Plan Recognition Systems"* Montreal, Canada.

Weida, R. (1996). *Closed Terminologies and Temporal reasoning in Descriptions for Plan Recognition.* Ph.D. thesis, Department of Computer Science, Columbia University, New York, NY.

Weida, R., & Litman, D. (1992). Terminological reasoning with constraint networks and an application to plan recognition. In *Proc. of the 3 rd International Conference on Principles of Knowledge Representation and Reasoning*, pp. 282–293 Cambridge, MA.

Weida, R., & Litman, D. (1994). Subsumption and recognition of heterogeneous constraint networks. In *Proceedings of CAIA-94*.

Wilkins, D. (1988). *Practical planning.* Morgan Kaufmann, San Mateo CA.