Hierarchical Plans in a Description Logic of Time and Action*

Alessandro Artale
Ladseb-CNR
I-35020 Padova PD, Italy
artale@ladseb.pd.cnr.it

Enrico Franconi
IRST
I-38050 Povo TN, Italy
franconi@irst.itc.it

Abstract

A formal language for representing and reasoning about time, actions and plans in a uniform way is presented. We employ an action representation in the style of Allen, where an action is represented by describing what is true while the action is occurring. In this sense, an action is defined by means of temporal constraints on the world states, which pertain to the action itself, and on other possible qualifications for the action occurring over time, while a plan is seen just as a complex action. Therefore, there is no difference between actions and plans in this framework. A distinction between action types and individual actions is supported by the formalism. In this paper, we show how to extend the language with a decomposition operator in a way of distinguishing the different actions composing a plan and not just the different temporal qualifications for the plan itself. Thus, a plan is a hierarchical structure made of its distinct component subparts.

The formal representation language used in this paper is a member of the family of description logics, and it is provided with a well founded syntax, semantics and calculus. The classification and recognition tasks, together with the basic subsumption procedure, form the basis for action management. An action description can be automatically classified into a taxonomy; an action 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

The structured logic of time and action considered in this paper [Artale and Franconi,1993; 1994a; Artale *et al.*,1994; Artale and Franconi,1994b; 1995] permits dealing with time, actions and plans in a uniform way. As opposed to the most common approaches to modeling actions as state change – e.g., the formal models based on *situation calculus* [McCarthy and Hayes,1969], the STRIPS planning system [Lifschitz,1987] – where actions are instantaneous and defined as functions from one state to another, we prefer to explicitly introduce the notion of time by admitting that actions take time, like in [Allen,1991]. We argue that this view of an action is closer to naturally occurring processes. Different actions can be concurrent or may overlap in time; effects of overlapping actions can be different from the sum of their individual effects; effects may not follow the action but more complex temporal relations may hold.

Starting from a formal language able to express temporally related objects – inspired by the work of [Schmiedel,1990] – actions are represented through temporal constraints on the world states, which pertain to the action itself, like in [Allen,1991]. With respect to [Allen,1991], our formalism has a clear distinction between the language for expressing action types (the conceptual level) and the language for expressing individual actions (the assertional level). Informally, plans are built by temporally relating action types in a compositional way using the temporal constructors available in the language; similar ideas were pursued by [Weida and Litman,1992]. In this way, since the temporal relationships are proper operators of the basic language, the distinction between actions and plans disappears. As a matter of fact, we do not need distinct languages for objects and states representation, for time representation, for actions representation, and for plans representation.

While actions describe how the world is affected by their occurrence, plans are described as a collection of action types constrained by temporal relations. In this way, a plan can be graphically represented as a temporal constraint network, where nodes denote action types. At this level of representation, plans can be seen as *complex actions*: since actions composing a plan can be expanded, plans and actions are not structurally different. This distinction is further elaborated in this paper, where each action composing a plan is considered as a step referring to a different individual action, and an

*This work has been partially supported by the Italian National Research Council (CNR), projects "Sistemi Informatici e Calcolo Parallelo", "Pianificazione Automatica", "Robotica" and "Ontologic and Linguistic Tools for Conceptual Modeling".

$$
\begin{array}{rcll}
C, D & \rightarrow & A \mid & \text{(atomic concept)} \\
 & & \top \mid & \text{(top)} \\
 & & C \sqcap D \mid & \text{(conjunction)} \\
 & & C \sqcup D \mid & \text{(disjunction)} \\
 & & p \downarrow q \mid & \text{(agreement)} \\
 & & p : C & \text{(selection)} \\
 & & C@X \mid & \text{(temporal qualifier)} \\
 & & C[Y]@X \mid & \text{(temporal subst. qualifier)} \\
 & & \diamond(X\ Y \cdots)\ \mathit{Tc}_1 \mathit{Tc}_2 \cdots \bullet C & \text{(temporal quantifier)} \\
p, q & \rightarrow & f \mid & \text{(atomic feature)} \\
 & & \star g \mid & \text{(atomic parametric feature)} \\
 & & p \circ q & \text{(feature chain)} \\
\mathit{Tc} & \rightarrow & (X\ (R)\ Y) & \text{(temporal constraint)} \\
R, S & \rightarrow & R\ ,\ S \mid & \text{(temporal disjunction)} \\
 & & \mathsf{s} \mid \mathsf{mi} \mid \mathsf{f} \mid \ldots & \text{(Allen temporal relations)} \\
X, Y & \rightarrow & \sharp \mid \mathsf{x} \mid \mathsf{y} \mid \ldots & \text{(temporal variables)}
\end{array}
$$

Figure 1: Syntax rules for the temporal concept language

appropriate function relates a plan to its steps.

The distinction made by the language between the conceptual and assertional aspect of the knowledge allows us to describe actions both at an abstract level (action type) and instance level (individual action). In this environment we exploit the *subsumption* calculus as the main inference tool for managing collections of action types. Given a set of observations of individual actions in the world the system is able to recognize which type of action has taken place at a certain time interval; this task is known as *recognition*. Action types are organized in a subsumption-based plan taxonomy, which can play the role of a plan library to be used for plan retrieval and individual plan recognition tasks [Kautz,1991]. In this way, we have refined the concept of what is currrently called *plan recognition*, by splitting it into the different tasks of *plan description classification* and *specific plan recognition with respect to a plan description*. In [Artale and Franconi,1994a] we proved the decidability and proposed an actual algorithm for the subsumption and recognition reasoning tasks in the basic language.

An interesting feature of this representational framework is its extensibility to cover different standard problems from the AI literature on actions. Homogeneous concepts can be introduced, for representing states whose properties holding at an interval do also hold at all subintervals [Artale *et al.*,1994]. We propose an approach to the *frame problem*, i.e., the problem to compute what is unchanged by an action. By introducing an *inertial* operator in the language we express the persistence of the truth of a proposition if there is no evidence of its falsity; with this approach the famous *Yale Shooting Problem* can be easily solved [Artale and Franconi,1994b]. We consider here the possibility to relate an action with more elementary actions, by *decomposing* it in partially ordered steps; this kind of representation is found in hierarchical planners.

2 The Formal Language

We briefly describe the basic description logic able to represent classes of individuals and their temporal relations – for a full discussion of the language please refer to [Artale and Franconi,1994a; 1995]. Basic types of the language are *concepts*, *individuals*, *temporal variables* and *intervals*. A concept is a description gathering the common properties among a collection of individuals. Concepts can describe entities of the world, states, 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 interval variable (resp. value) defined by the constraint network. In this way, *action types* (resp. *individual actions*) can be represented in a uniform way by temporally related concepts (resp. individuals).

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 $\star g$) and constrained *interval variables* (denoted by X, Y) according to the abstract syntax rules of figure 1. Temporal variables are introduced by the temporal existential quantifier "$\diamond$". Variables appearing in temporal constraints should be declared within the same temporal quantifier, with the exception of the special variable $\sharp$. Explicit temporal variables allow for complex temporal patterns to be specified. [Bettini,1993] proposes a variable-free temporal extension of a description logic, which can not express temporal patterns like $(y\ \mathsf{starts}\ x)(z\ \mathsf{finishes}\ x)(y\ \mathsf{meets}\ z)$. We believe that the introduction of variables in the language does not affect the principle of description logics which are considered as variable-free languages; in fact, variables are limited in the temporal realm.

We are going now to informally present the *meaning* of the terms of the language. Concept expressions are interpreted in our logic 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 denote an action, then its interpretation can be seen as the set of individual actions of that type occurring at some interval.

Within a concept, the special "$\sharp$" variable denotes the generic interval at which the concept itself *holds*; in the case of actions, it refers to the temporal interval at which the action itself *occurs*. A concept holds at an interval X if it is temporally qualified at X – written $C@X$; in this way, every occurrence of $\sharp$ embedded within the concept expression C is interpreted as the X variable. Since any concept is implicitly temporally qualified at the special $\sharp$ variable, it is not necessary to explicitly qualify concepts at $\sharp$. The temporal existential quantifier introduces interval variables, related each other and possibly to the $\sharp$ variable in a way defined by the set of *temporal constraints*. The informal meaning of a concept with a temporal existential quantification can be understood with the following example in the action domain.

$$
\begin{array}{l}
\texttt{Basic-Stack} \doteq \\
\quad \diamond(x\ y)(x\ \mathsf{m}\ \sharp)(\sharp\ \mathsf{m}\ y). \\
\qquad (\star\texttt{BLOCK} : (\texttt{OnTable}@x \sqcap \texttt{OnBlock}@y))
\end{array}
$$

`Basic-Stack` denotes, according to the definition (a terminological axiom), any action occurring at some interval involving a $\star$`BLOCK` that was once `OnTable` and then

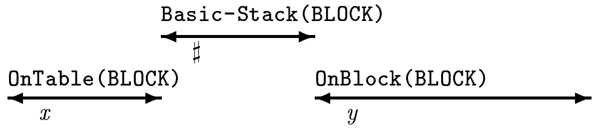

Figure 2: Temporal dependencies in the definition of the `Basic-Stack` action.

`OnBlock`. 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 m $\sharp$) and ($\sharp$ m y) state that the interval denoted by x should meet the interval denoted by $\sharp$ – the occurrence interval of the action type `Basic-Stack` – and that $\sharp$ should meet y. Figure 2 shows the temporal dependencies of the intervals in which the concept `Basic-Stack` holds. The parametric feature $\star$`BLOCK` plays the role of *formal* parameter of the action, mapping any individual action of type `Basic-Stack` to the block to be stacked, independently from time. Please note that, whereas the existence and identity of the $\star$`BLOCK` of the action is time invariant, it can be qualified differently in different intervals of time, e.g. the $\star$`BLOCK` is necessarily `OnTable` only during the interval denoted by x. On the other hand, the (non-parametric) feature `OnTable` may change its value over time – thus, representing a changing state.

The assertion `Basic-Stack`(i,a) says that a is an individual action of type `Basic-Stack` occurred at the interval i. Moreover, the same assertion implies that a is related to a $\star$`BLOCK` , say b, which is of type `OnTable` at some interval j, meeting i, and of type `OnBlock` at another interval l, met by i.

$$\begin{aligned}&\texttt{Basic-Stack}(i,a) \Longrightarrow\\ &\quad \exists b.\ \star\texttt{BLOCK}(a,b) \wedge\\ &\qquad \exists j,l.\ \big(\texttt{OnTable}(j,b) \wedge \texttt{OnBlock}(l,b) \wedge\\ &\qquad\qquad \mathsf{m}(j,i) \wedge \mathsf{m}(i,l)\big)\end{aligned}$$

Now we show how from a series of observations in the world we can make action recognition – i.e. the task we have already called *specific plan recognition with respect to a plan description.* This is an inference service which computes if an *individual action* is an instance of an *action type* at a certain interval. Given the following state of affairs, describing a world where blocks can be on the table and/or on each other and where a generic individual action a is taking place at time interval i_a having the block *block-a* as its parameter:

$$\begin{aligned}&\star\texttt{BLOCK}(a, \textit{block-a}),\\ &\mathsf{m}(i_1,i_a),\ \texttt{OnTable}(i_1, \textit{block-a}),\\ &\mathsf{mi}(i_2,i_a),\ \texttt{OnBlock}(i_2, \textit{block-a})\end{aligned}$$

then, the system recognizes that in the context of a knowledge base Σ, composed by the above theory and the definition of the `Basic-Stack` concept, the individual action a is an instance of the concept `Basic-Stack` at the time interval i_a, i.e. $\Sigma \models \texttt{Basic-Stack}(i_a,a)$.

3 Decomposition of Actions

In this section we consider plans as hierarchical structures whose constituent actions could be seen as its decomposing steps. Functional relations decompose an individual plan to its steps, allowing us to take into account plans where the component actions do not share the same actual parameters. Suppose, for example, that we want represent the plan composed by a sequence of two Stack actions. This plan could be represented as:

$$\begin{aligned}&\texttt{Double-Stack}\ \doteq\\ &\quad \Diamond(x\ y)(x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ \mathsf{m}\ y).\\ &\qquad (\texttt{Basic-Stack}@x \sqcap \texttt{Basic-Stack}@y)\end{aligned}$$

According to this plan description, each individual plan – say for example a – of type `Double-Stack` has to be an instance of two `Basic-Stack` actions occurring at two "meeting" intervals, too.

$$\begin{aligned}&\texttt{Double-Stack}(i,a) \Longrightarrow\\ &\quad \exists j,l.\ \big(\texttt{Basic-Stack}(j,a) \wedge \texttt{Basic-Stack}(l,a) \wedge\\ &\qquad\quad \mathsf{d}(j,i) \wedge \mathsf{d}(l,i) \wedge \mathsf{m}(j,l)\big)\end{aligned}$$

This representation does not take into account the case where the actual parameters involved in the two `Basic-Stack` actions are different. In fact, there is only one action involved in the former definition: a. The individual action a is qualified as a `Double-Stack` plan at the current interval, it is a `Basic-Stack` action at the interval j, and it is a `Basic-Stack` action again at the interval l. The crucial point is that the parameters involved in the `Stack` actions are described by means of *parametric features* – functions that remain invariant during time – which in this case apply to the *same* individual `Basic-Stack` action – a. So, the parameter of the action a must be the same at the intervals j and l, and the case of sequential actions of the same type but with different parameters can not be expressed. In the case of the `Basic-Stack` actions, the actual $\star$`BLOCK` parameter value should be the same at both j and l intervals, disallowing the possibility of expressing the plan composed by the sequence of two `Basic-Stack` actions manipulating two different blocks.

The described behavior is the consequence of having a fully substitutional framework for plan definitions. Since plans are seen just as complex actions, the *parts* composing a plan are just different qualifications on the same individual entity. A solution to this problem is to associate distinct individual actions for each action step occuring in a given plan, in such a way that parametric features, being applied on distinct action instances, can introduce different parameter values. This is realized by relating the plan to the component action types by means of different functions, using the *decomposition* operator. We add the following syntactic rules, to be used for plan definitions:

$$
\begin{array}{rcll}
C, D & \rightarrow & C@@X & \text{(decomposition)} \\
p, q & \rightarrow & p@@X \downarrow q@@Y & \text{(temporal agreement)}
\end{array}
$$

where $C@@X$ means that C is a step in a plan and the temporal agreement specifies equality constraints between paths of features defined in different steps of a plan (see, e.g. [Weida and Litman,1994]).

More formally, we add the following semantic equations to the semantics of the basic description logic [Artale and Franconi,1994a]:

$$
\begin{aligned}
&(C@@X)^{\mathcal{I},\mathcal{S}}_{\mathcal{V},t,\mathcal{H}} = \\
&\quad \{a \in dom\,(X^{\mathcal{S}}) \mid X^{\mathcal{S}}(a) \in C^{\mathcal{I},\mathcal{S}}_{\mathcal{V},\mathcal{V}(X),\mathcal{H}}\}
\end{aligned}
$$

$$
\begin{aligned}
&(p@@X \downarrow q@@Y)^{\mathcal{I},\mathcal{S}}_{\mathcal{V},t,\mathcal{H}} = \\
&\quad \{a \in dom\,(X^{\mathcal{S}} \circ p^{\mathcal{I},\mathcal{S}}_{\mathcal{V},\mathcal{V}(X),\mathcal{H}}) \cap dom\,(Y^{\mathcal{S}} \circ q^{\mathcal{I},\mathcal{S}}_{\mathcal{V},\mathcal{V}(Y),\mathcal{H}}) \mid \\
&\quad\ X^{\mathcal{S}} \circ p^{\mathcal{I},\mathcal{S}}_{\mathcal{V},\mathcal{V}(X),\mathcal{H}}(a) = Y^{\mathcal{S}} \circ q^{\mathcal{I},\mathcal{S}}_{\mathcal{V},\mathcal{V}(Y),\mathcal{H}}(a)\}
\end{aligned}
$$

where $\cdot^{\mathcal{S}}$ – called the *Step Interpretation* – is a function associating a newly generated function to each interval introduced by a decomposition operator.

The representation of the Double-Stack plan in the extended language is:

$$
\begin{aligned}
&\texttt{Double-Stack} \ \doteq \\
&\quad \Diamond(x\ y)\ (x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ \mathsf{m}\ y). \\
&\quad\quad \big(\texttt{Basic-Stack}@@x \sqcap \texttt{Basic-Stack}@@y\big)
\end{aligned}
$$

Since the nodes x and y of the conceptual temporal constraint network introduce distinct individual actions, we can now distinguish the different instances of the two Basic-Stack actions occurring in distinct intervals:

$$
\begin{aligned}
&\texttt{Double-Stack}(i, a) \Longrightarrow \\
&\quad \exists b, c, j, l.\ \big(\texttt{X}(a, b) \land \texttt{Basic-Stack}(j, b) \land \\
&\quad\quad\quad \texttt{Y}(a, c) \land \texttt{Basic-Stack}(l, c) \land \\
&\quad\quad\quad \mathsf{d}(j, i) \land \mathsf{d}(l, i) \land \mathsf{m}(j, l)\big)
\end{aligned}
$$

where X and Y are newly generated functions, mapping the individual plan a to its individual steps b and c. It is now possible the accomplishment of a plan moving two different blocks during two Basic-Stack events in sequence.

The case of *simultaneous* occurrences of the same action type with different parameters can be expressed in the extended language. The plan where two possibly different blocks are moved by two Basic-Stack actions at the same time is:

$$
\begin{aligned}
&\texttt{Sim-Double-Stack} \ \doteq \\
&\quad \Diamond(x\ y)(x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ =\ y). \\
&\quad\quad (\texttt{Basic-Stack}@@x \sqcap \texttt{Basic-Stack}@@y)
\end{aligned}
$$

We want now to address the case where different individual actions share common parameters or where a plan is made of the repetition, in different intervals, of the same individual action. We exploit the *temporal agreement* construct – $p@@X \downarrow q@@Y$, which specifies equality constraints between features introduced in different intervals. In the following example, the Same-Block-Double-Stack plan is constituted of two Basic-Stack actions moving the same ⋆BLOCK:

$$
\begin{aligned}
&\texttt{Same-Block-Double-Stack} \ \doteq \\
&\quad \Diamond(x\ y)\ (x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ \mathsf{m}\ y). \\
&\quad\quad (\texttt{Basic-Stack}@@x \sqcap \texttt{Basic-Stack}@@y \sqcap \\
&\quad\quad\ \star\texttt{BLOCK}@@x \downarrow \star\texttt{BLOCK}@@y)
\end{aligned}
$$

Same-Double-Stack is a plan where a Stack action occurs twice in distinct intervals, with the same fillers for the actual parameters:

$$
\begin{aligned}
&\texttt{Same-Double-Stack} \ \doteq \\
&\quad \Diamond(x\ y)\ (x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ \mathsf{m}\ y). \\
&\quad\quad (\texttt{Basic-Stack}@@x \sqcap \texttt{Basic-Stack}@@y \sqcap \\
&\quad\quad\ @@x \downarrow @@y)
\end{aligned}
$$

The construct $@@X \downarrow @@Y$ – a particular case of temporal agreement – imposes an equality constraint between the functions associated to the nodes X and Y , so that the two Basic-Stack actions are instantiated by the same individual object; thus, the actual parameters associated to the Basic-Stack actions are equal.

Finally we propose an example where the actions composing the plan have different types. Consider, in the cooking domain, the following example:

$$
\begin{aligned}
&\texttt{Heat-Noodles} \ \doteq \\
&\quad \Diamond(x\ y)\ (x\ \mathsf{d}\ \sharp)(y\ \mathsf{d}\ \sharp)(x\ \mathsf{b}, \mathsf{m}\ y). \\
&\quad\quad (\texttt{Make-Noodles}@@x \sqcap \texttt{Heat}@@y \sqcap \\
&\quad\quad\ \star\texttt{Agent}@@x \downarrow \star\texttt{Agent}@@y)
\end{aligned}
$$

The definition specifies that the plan has two steps, Make-Noodles and Heat, instantiated by distinct individual actions, while the temporal agreement forces the ⋆Agents of the two steps to be the same.

References

[Allen, 1991] James F. Allen. Temporal reasoning and planning. In James F. Allen, Henry A. Kautz, Richard N. Pelavin, and Josh D. Tenenberg, editors, *Reasoning about Plans*, chapter 1, pages 2–68. Morgan Kaufmann, 1991.

[Artale and Franconi, 1993] Alessandro Artale and Enrico Franconi. A unified framework for representing time, actions and plans. In F. D. Anger, H. W. Guesgen, and J. van Benthem, editors, *Workshop Notes of the IJCAI-93 Workshop on Temporal and Spatial Reasoning*, pages 193–217, Chambery, France, August 1993. Also in the Workshop Notes of the IJCAI-93 Workshop on Object-Based Representation System, pages 32–44; a shorter version appears in the Proceedings of the Italian Planning Workshop 1993 (IPW'93), pages 167–172, Roma Italy, September 1993.

[Artale and Franconi, 1994a] Alessandro Artale and Enrico Franconi. A computational account for a description logic of time and action. In *Proc. of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR-94)*, pages 3–14, Bonn, Germany, May 1994.

[Artale and Franconi, 1994b] Alessandro Artale and Enrico Franconi. Persistent properties in a description logic of time and action. In *Working Notes of the*

*AI*IA Temporal Reasoning Workshop 1994*, pages 9–12, Parma, Italy, September 1994.

[Artale and Franconi, 1995] Alessandro Artale and Enrico Franconi. Time, actions and plans representation in a description logic. *International Journal of Intelligent Systems*, 1995. To appear.

[Artale *et al.*, 1994] Alessandro Artale, Claudio Bettini, and Enrico Franconi. Homogeneous concepts in a temporal description logic. In *Proceedings of the International Workshop on Description Logics*, pages 36–41. DFKI, Saarbrücken, Bonn, Germany, May 1994.

[Bettini, 1993] Claudio Bettini. A family of temporal terminological logics. In *Proc. of the third Congress of the Italian Association for Artificial Intelligence, AIIA-93*, LNAI. Springer-Verlag, 1993.

[Kautz, 1991] Henry A. Kautz. A formal theory of plan recognition and its implementation. In James F. Allen, Henry A. Kautz, Richard N. Pelavin, and Josh D. Tenenberg, editors, *Reasoning about Plans*, chapter 2, pages 69–126. Morgan Kaufmann, 1991.

[Lifschitz, 1987] Vladimir Lifschitz. On the semantics of STRIPS. In *The 1986 Workshop on Reasoning about Actions and Plans*, pages 1–10. Morgan Kaufman, 1987.

[McCarthy and Hayes, 1969] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of Artificial Intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 4, pages 463–502, Edinburgh, UK, 1969. Edinburgh University Press.

[Schmiedel, 1990] A. Schmiedel. A temporal terminological logic. In *Proc. of AAAI-90*, pages 640–645, Boston, MA, 1990.

[Weida and Litman, 1992] Robert Weida and Diane Litman. 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 (KR-92)*, pages 282–293, Cambridge, MA, October 1992.

[Weida and Litman, 1994] Robert Weida and Diane Litman. Subsumption and recognition of heterogeneous constraint networks. In *Proceedings of CAIA-94*, 1994.