Knowledge About Planning: On the Meaning and Representation of Plan Decomposition

Diane Horton and **Graeme Hirst**
Department of Computer Science,
University of Toronto
Toronto, CANADA M5S 1A4
dianeh@ai.toronto.edu and *gh@ai.toronto.edu*

Abstract

Plan-related inference has been one of the most-studied problems in Artificial Intelligence. Pollack (1990) has argued that a plan should be seen as a set of mental attitudes towards a structured object. Although the objects of these attitudes have received far more attention to date than the attitudes themselves, little has been said about the exact meaning of one of their key components — the decomposition relation. In developing a plan representation for our work on plan misinference in dialogue, we have explored two of the possible meanings, their implications, and the relationship between them. These issues underly the literature, and in this paper, we step back and discuss them explicitly.

1 Introduction

Early representations of planning knowledge adopted the frame-like structure of STRIPS (Fikes and Nilsson 1971). A decomposition, like any other attribute of a plan, was simply a slot whose filler was a list of actions or goals. Examples of this approach to representation include (Sacerdoti 1974) and in natural language research, (Allen and Perrault 1980). The meaning of the slots and fillers was generally unstated; one could only infer it by examining the algorithms that operated on these frames[1]. This sort of representation is still quite common. See, for example, (Weida and Litman 1992), (Löwgren 1992) and (Eller and Carberry 1992).

The present work is motivated by our long-term goal to develop an account of a plan-related phenomenon in natural language: plan "misinference" in dialogue (see section 8). It is out desire that it be a "knowledge-level" account (Newell 1982; Levesque and Brachman 1986). Thus, we require a representation that has a clear, explicit meaning, divorced from the details of the algorithms that will operate on it. In the process of developing such a representation, we have examined issues bearing on the choice of a representation for plan decomposition.

2 Downwards and upwards decomposition

If actions (or more precisely, the occurrences of actions) are treated as objects with truth values, i.e., propositions or predicates, then the meaning of the decomposition relation can be expressed clearly and explicitly using logical implication. But which way should the implication go? Does an action imply its substeps, or do the substeps imply the action? Kautz (1991) takes the approach that an action implies its substeps. The decomposition of the action of getting a PhD, for example, could be represented in his framework as follows[2]:

$$\begin{aligned}\forall x, \mathrm{GetPhd}(x) \supset\ & \mathrm{TakeCourses}(\mathrm{step1}(x)) \wedge \\ & \mathrm{DoQual}(\mathrm{step2}(x)) \wedge \\ & \mathrm{DoThesis}(\mathrm{step3}(x)) \wedge \kappa.\end{aligned} \tag{1}$$

A decomposition rule such as this gives the *necessary* substeps of an action. Such a rule can be represented pictorially by a tree in which each node is an action and the children of a node are its substeps:

$$\begin{tikzcd}
 & \text{GetPhd} \arrow[dl] \arrow[d] \arrow[dr] & \\
\text{TakeCourses} & \text{DoQual} & \text{DoThesis}
\end{tikzcd}$$

Since arrows to denote the decomposition rules point downwards in this tree, we call this **downwards decomposition**.

Helft and Konolige (1992) adopt the opposite form of decomposition, in which the substeps imply the action. Our PhD rule, in their (propositional) framework, might look like this:

$$\mathrm{TakeCourses} \wedge \mathrm{DoQual} \wedge \mathrm{DoThesis} \supset \mathrm{GetPhd}. \tag{2}$$

[1]Systems for plan construction that are based on this sort of representation frequently use a backchaining algorithm that implicitly assumes the slot fillers are sufficient for the plan (*i.e.*, performing the substeps or achieving the subgoals in the decomposition slot is sufficient to perform the plan.)

[2]The capitalized tokens are event types, the 'step' tokens are role functions that take an event instance and return another event type, and κ consists of constraints on the events, including temporal constraints, preconditions, and effects.

A rule of this form gives the *sufficient* substeps of an action. We use the term **upwards decomposition** for this sort of rule, to reflect the fact that the implication arrows point upwards:

$$\begin{tikzcd} & \text{GetPhd} & \\ \text{TakeCourses} \arrow[ur] & \text{DoQual} \arrow[u] & \text{DoThesis} \arrow[ul] \end{tikzcd}$$

Allen's recent work (1991) also uses upwards decomposition. However, his formalism, in contrast to Kautz's, and Helft and Konolige's, does not keep the details of the inference strategy separate from the specification of those inferences that are supported[3].

A third option is to use only bidirectional decomposition, and thus to represent all and only the sub-actions whose occurrences are both necessary and sufficient for the occurrence of the parent action. As the next section will demonstrate, there are actions whose decomposition cannot be expressed in this manner.

3 Do we require this distinction?

Some actions have one or more sub-actions that are sufficient but not necessary, and can thus only be expressed using upwards decomposition. For instance, it may be that passing an exemption exam is one way to get credit in a course, but it is not necessary[4]:

$$\text{PassExemptionExam} \supset \text{GetCredit} \tag{3}$$

This cannot be represented with downwards decomposition. If, on the other hand, a set of sub-actions is necessary but not sufficient for an action, this fact can be expressed using downwards decomposition but not upwards. For example, satisfying breadth requirements may be necessary in order to earn a B.Sc. degree, but not sufficient:

$$\text{EarnBSc} \supset \text{SatisfyBreadth}. \tag{4}$$

Thus, the semantic distinctions between upwards, downwards, and bi-directional decomposition are necessary in order to properly express decomposition for some actions.

4 Relevance of the inference strategy

Although, as we have just argued, there are domain facts that fit the model of upwards or downwards decomposition only, we need not represent such facts directly. For instance, with only upwards decomposition, we can still achieve the effect of a (downwards) fact of the form $\alpha \supset \beta$, by including β in all of the upwards decomposition rules and making a closed-world assumption (Reiter 1981). However, if $\alpha \supset \beta$ is something that we are sure is true, it would seem preferable to represent it as a rule, rather than to have it be inferred with the assistance of a fallible closure assumption.

An important general relationship between the two kinds of decomposition can be expressed in terms of closure: Looking back at (5) upwards and (6) downwards decomposition, the downwards form is what we would get if we made a closed-world assumption about the knowledge represented in the upwards form[5].

Of course, we can also change the inferences that are sanctioned by using an abductive, rather than deductive, inference mechanism. Abduction, intuitively, reverses the direction of inference; however, for both upwards and downwards decomposition, the inferences supported by abduction and deduction are not simply inverses of each other. With downwards decomposition we can either:

- from an observed action, deduce all of its substeps, or
- from the observation of *any* of the substeps, abduce the action.

On the other hand, with upwards decomposition, we can either:

- from the observation of *all* of the substeps, deduce the action, or
- from an observed action, abduce all of its substeps.

Notice that with upwards decomposition, we cannot, using either straightforward deduction or abduction, infer an action from the observation of some of its substeps. Yet this ability is usually desirable. Although it uses upwards decomposition only, Helft and Konolige's theory does not suffer from this problem because it defines plan recognition using, effectively, a combination of deduction *and* abduction[6].

Thus the choice of a form of decomposition is not independent of the sort of inference mechanism to be adopted.

Kautz's work provides another demonstration of this fact. With his choice of downward decomposition and a deductive model of plan inference, inference from a substep of an action to the action is not supported. Kautz introduces a notion of entailment in minimal models (and corresponding proof-theoretic notions) to achieve this effect.

[3]In Allen's theory, three levels of rule are used to infer an action from its substeps, the first of which takes us from the substeps to the conclusion that the decomposition happened. The rules at this level involve the operator "<<<", which is a system-level construct — it bears on the actual inference strategy, not merely on which inferences are sanctioned.

[4]For simplicity, here and throughout this paper, we use a very simple, propositional representation for actions. Other representational issues are orthogonal.

[5]Technically, (6) is the Clark completion (Clark 1978) of (5). In the first-order case, this relationship is still strong, but whether or not it holds exactly depends on the form of the particular first-order theory.

[6]Very roughly (ignoring elements to do with minimization), Helft and Konolige define a solution to a plan recognition problem to be: a plan A, a subset of the plan A_0, and a goal g such that $T_w \cup \{A_0\} \vdash O$, and $T_a \cup \{A\} \vdash g$, where T_w is the planning theory, T_a is the *agent's* planning theory, and O are the observed actions. Since the solution involves elements on both sides of the '$\vdash$', the effects of both deduction and abduction are achieved.

5 Allowing alternative decompositions

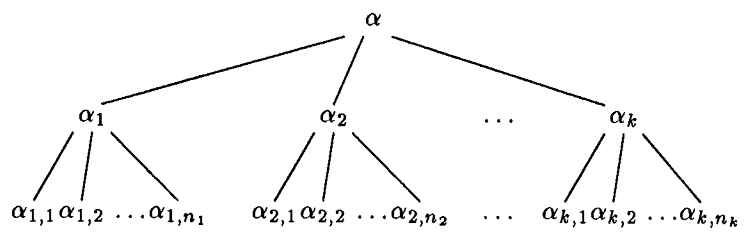
So far, we have focused on actions with a single decomposition. More generally, we must allow for alternate decompositions. With upwards decomposition, this can be done simply by having a series of rules (implicitly conjoined), one per alternative:

$$\begin{array}{l} \alpha_{1,1} \wedge \alpha_{1,2} \wedge \ldots \wedge \alpha_{1,n_1} \supset \alpha \\ \alpha_{2,1} \wedge \alpha_{2,2} \wedge \ldots \wedge \alpha_{2,n_2} \supset \alpha \\ \quad\ldots \\ \alpha_{k,1} \wedge \alpha_{k,2} \wedge \ldots \wedge \alpha_{k,n_k} \supset \alpha \end{array} \tag{5}$$

With downwards decomposition, we can simply disjoin the alternatives:

$$\begin{array}{rl} \alpha \supset & (\alpha_{1,1} \wedge \alpha_{1,2} \wedge \ldots \wedge \alpha_{1,n_1}) \vee \\ & (\alpha_{2,1} \wedge \alpha_{2,2} \wedge \ldots \wedge \alpha_{2,n_2}) \vee \\ & \quad\ldots \\ & (\alpha_{k,1} \wedge \alpha_{k,2} \wedge \ldots \wedge \alpha_{k,n_k}) \end{array} \tag{6}$$

Since each alternative decomposition may have different preconditions and effects, with either of these forms they must be attached to the conjunctions that represent each individual decomposition. This is perhaps awkward. An alternative is to introduce another level of representation, with names for each alternative decomposition, as follows (in the diagram, we deliberately avoid committing to a direction for the implications):


α
α_1
α_2
$\ldots$
α_k
$\alpha_{1,1}\alpha_{1,2}\ldots\alpha_{1,n_1}$
$\alpha_{2,1}\alpha_{2,2}\ldots\alpha_{2,n_2}$
$\ldots$
$\alpha_{k,1}\alpha_{k,2}\ldots\alpha_{k,n_k}$

With this form of decomposition, preconditions and effects for each alternative decomposition can be associated with the new actions $\alpha_1, \alpha_2, \ldots, \alpha_n$. However, now the question of which way the implication should go must be answered for two types of decomposition layer[7].

The relationships between α and $\alpha_1, \alpha_2, \ldots \alpha_n$ are perhaps better stated using specialization than decomposition[8]. Specialization automatically provides inheritance from action α to its specializations $\alpha_1, \alpha_2, \ldots \alpha_n$ of, for instance, preconditions and effects. Furthermore, the newly-introduced specializations are meaningful additions to the ontology; they share among themselves those properties that they inherit from α, yet they are distinct at least in that they have different substeps.

[7]Calistri-Yeh (1991) uses this sort of representation (with bidirectional implications between all layers).

[8]This is, in fact, Kautz's approach (with downward implication for the decompositions from each $\alpha_i (1 \leq i \leq k)$ to its substeps $\alpha_{i,1}$ to α_{i,n_i}.).

6 Causality

Helft and Konolige cite causality as their reason for using upwards decomposition, arguing that the subactions cause the action, and not vice versa. More precisely, one might say that the occurrence of the subactions causes the occurrence of the action. Inference from an action down to its substeps can follow as a result of the backwards reasoning that takes place during plan construction. In Helft and Konolige's theory then, plan construction is taken to be an abductive process that finds a plan satisfying the following:

$$\text{planning-theory} + \text{plan} \vdash \text{goal}.$$

Given that the decomposition axioms are upwards (going from the subactions to the action), abductive plan construction will allow us to derive the subactions from the action. But to actually write that the action *implies* the subactions is another matter, and Helft and Konolige argue that this is not the direction in which causality flows. However, there are other links than causality that an implication might represent. For instance, if substeps are necessary (in the technical sense), a downwards 'action implies substeps' rule is not incorrect.

7 Tolerance for incomplete information

Downwards and upwards decomposition tolerate different sorts of incomplete information:

- With upwards decomposition, as in (5), we need not know all the decompositions of the action, but we must know all of the [sufficient] substeps in each in order to write rules that are "correct", *i.e.*, that accurately reflect the domain.
- With downwards decomposition, as in (6), we must know all of the decompositions, but need not know all [necessary] substeps in each in order to write rules that are correct.

Although most would agree that "incorrect" rules are unacceptable, it is quite another matter to write down correct rules but then use them in a way that makes assumptions that we know are defeasible — for instance, closed-world assumptions. This sort of approach is quite standard, and its merits are especially clear in the context of databases. We certainly do not want our databases to explicitly represent false information, but we must make some sort of closed-world assumption in order to answer even a simple question such as "how many computer science courses are there?" (Reiter 1984).

8 Our work on plan misinference in dialogue

Our interest is in modelling the detection of, and recovery from, plan **misinference** — the attribution of the wrong plan to the agent[9]. We are particularly concerned with plan misinference and its consequences in

[9]Others have addressed the problem of plan inference where the plan itself may be faulty, *e.g.*, (Pollack 1990). Such work could be said to concern faulty-plan inference, whereas we are currently interested in faulty plan-inference.

the context of natural language dialogue. We would like to model the reasoning behind dialogues such as the following:

(1)	▷ 1	Ann:	Where's Computer Science 104 taught?
	2	Bob:	Sidney Smith Building, room 2118, but the class is full.
	3	Ann:	No, I teach it.

In this example, Bob is correct in thinking that Ann plans to go to the classroom where 104 is taught, but he also thinks, wrongly, that this is part of a plan to take the course.

When dealing with conversation, we have to consider not only domain actions but also linguistic actions, and these linguistic actions can themselves be misinferred:

(2)	1	Ann:	Add the sugar, and then beat that with the butter.
	2	Bob:	Okay.
	3	Ann:	Now sift in the flour.
	▷ 4	Bob:	What should we do with the eggs?
	5	Ann:	You don't add them until the end.
	6	Bob:	I meant shouldn't we put them in the fridge.

The problem here arises at utterance (2.4). Ann wrongly thinks that Bob's plan is to continue to track the cake-making process by mentioning the step involving eggs. In fact, his plan is to find out what to do with the eggs, independent of the cake-making.

8.1 Our approach

We are working on a formal, knowledge-level account of the reasoning underlying plan misinference, in contrast to an account expressed in terms of algorithms and data structures. The only other work we are aware of on plan misinference, that of Eller and Carberry (1992), is of the latter sort. We view the two as complementary; algorithmic work typically focuses more on the domain, while knowledge-level work focuses on pinning down the foundations. Eventually, one hopes, the two will meet!

The primary components of our model are (1) a representation for planning knowledge; (2) a definition of what constitutes a plan inference problem and a solution to such a problem; and (3) an analysis of how the solutions change as further observations are available. It is the last of these that will model the process of detecting and recovering from a plan misinference.

8.2 Our representation for plans

The representation language is a sorted, first-order language with equality. Following Davidson (1967) and others since (*e.g.*, Kautz (1991) and Allen (1991), we treat actions as objects, and use role predicates on them to state their role fillers.

The representation has two levels: one contains domain-specific planning knowledge (specific preconditions, decompositions, etc.), and the other, domain-independent facts, including what it means for an action to occur. Domain specific planning knowledge is represented by a tuple $\Sigma = \langle \Sigma_a, \Sigma_p, \Sigma_e, \Sigma_d, \Sigma_s \rangle$ where Σ_a is a set of atoms, each representing an action type, Σ_p is a set of precondition axioms, Σ_e is a set of effect axioms, Σ_d is a set of decomposition axioms, and Σ_s is a set of specialization axioms. We have developed a representation, based on event calculus (Kowlaski and Sergot 1986) that fills in the details of each of these components. We use specialization to represent the relationship between an action and its alternative decompositions, for the reasons discussed in section (5). For the relationship between a particular decomposition and the substeps in involves, for the time being we use bidirectional decomposition, in effect making the simplifying assumption that our knowledge about decompositions is complete. This assumption is appropriate in situations where the the system is expert in the domain, such as in intelligent tutoring systems. Of course, the *user's* plan knowledge may still be incomplete or even incorrect. In fact, this is highly likely in tutoring systems. See (Pollack 1990) and (Calistri-Yeh 1991) for approaches to that problem.

8.3 Current work

Recent work by Pinto and Reiter (1992) has made it possible to express the occurrence of actions in the situation calculus (McCarthy and Hayes 1969). This ability is necessary for any action theory that is to underly plan inference. Because situation calculus has a first-order semantics (whereas event calculus relies on Prolog's negation-as-failure mechanism), we are reworking our representation to be based upon this new version of situation calculus.

We have been investigating the viability of combining alternative forms of inference (deduction, abduction, and various combinations of both) with alternative directions of decomposition rule. Our next step will be to select one and explore its behavior as new actions are continuously observed.

9 Summary

We have argued that a theory should be clear about what decomposition means, and that the following bear on a choice:

- there are domain facts which can be directly represented only with upwards decomposition, and others that can be represented only with downwards decomposition.
- For a given theory, it may be possible to achieve the effect of downward decomposition by performing completion on upwards rules.
- upwards and downwards decomposition tolerate different kinds of incomplete knowledge.
- upwards and downwards decomposition support different inferences that are not simply inverses of each other.

Acknowledgments

The authors acknowledge the support of the Natural Sciences and Engineering Research Council of Canada. We thank Sheila McIlraith, Susan McRoy, Javier Pinto, Ray Reiter, and Jan Wiebe for helpful discussions.

References

Allen, J. F. (1991). Temporal reasoning and planning. In Allen, J. F., Kautz, H. A., Pelavin, R. N., and Tenenberg, J. D., editors, *Reasoning About Plans*, chapter 1, pages 1–68. Morgan Kaufmann, San Mateo, CA.

Allen, J. F. and Perrault, C. R. (1980). Analyzing intention in utterances. *Artificial Intelligence*, 15:143–178. Reprinted in *Readings in Natural Language Processing*, Barbara J. Grosz, Karen Sparck Jones and Bonnie Lynn Webber, editors, pages 441–458, 1986.

Calistri-Yeh, R. J. (1991). Utilizing user models to handle ambiguity and misconceptions in robust plan recognition. *User Modelling and User-Adapted Interaction.*

Clark, K. L. (1978). Negation as failure. In Gallaire, H. and Minker, J., editors, *Logic and Data Bases*, pages 293–322. Plenum Press.

Davidson, D. (1967). The logical form of action sequences. In Rescher, N., editor, *The Logic of Decision and Action.* University of Pittsburgh Press.

Eller, R. and Carberry, S. (1992). A meta-rule approach to flexible plan recognition in dialogue. *User Modelling and User-Adapted Interaction*, 2(1–2):27–53.

Fikes, R. E. and Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208.

Goldman, A. I. (1970). *A theory of human action.* Princeton University Press, Princeton, New Jersey.

Helft, N. and Konolige, K. (1991). Plan recognition as abduction and relevance. Unpublished MS.

Kautz, H. (1991). A formal theory of plan recognition and its implementation. In Allen, J. F., Kautz, H. A., Pelavin, R. N., and Tenenberg, J. D., editors, *Reasoning About Plans*, chapter 2, pages 69–125. Morgan Kaufmann, San Mateo, CA.

Kowalski, R. and Sergot, M. (1986). A logic-based calculus of events. *New Generation Computing*, 4:67–95.

Levesque, H. J. and Brachman, R. J. (1986). Knowledge level interfaces to information systems. In Brodie, M. and Mylopoulos, J., editors, *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies*, chapter 2, pages 13–34. Springer-Verlag.

Löwgren, J. (1992). The Ignatius environment: Supporting the design and development of expert-system user interfaces. *IEEE Expert*, 7(4):49–57.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, pages 463–502. Reprinted in *Readings in Artificial Intelligence*, Webber, B.L. and Nilsson, N.J., editors, pp. 431–450, 1981.

Newell, A. (1982). The knowledge level. *Artificial Intelligence*, 18(1):87–127.

Pinto, J. and Reiter, R. (1992). Adding a time line to the situation calculus: Extended abstract. Submitted for publication.

Pollack, M. E. (1990). Plans as complex mental attitudes. In Cohen, P., Morgan, J., and Pollack, M., editors, *Intentions in Communication*, System Development Foundation Benchmark Series, pages 77–104. MIT Press.

Reiter, R. (1981). On closed world data bases. In Webber, B. L. and Nilsson, N., editors, *Readings in Artificial Intelligence*, pages 119–140. Tioga Publishing Company, Los Altos, California. Reprinted from Gallaire and Minker 1978, Logic and Data Bases.

Reiter, R. (1984). Towards a logical reconstruction of relational database theory. In Brodie, M. L., Mylopoulos, J., and Schmidt, J., editors, *On Conceptual Modelling: Perspectives fromArtificial Intelligence, Databases and Programming Languages*, chapter 8, pages 191–233. Springer.

Sacerdoti, E. D. (1974). Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5(2):115–135.

Weida, R. and Litman, D. (1992). Terminological reasoning with constraint networks and an application to plan recognition. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR92)*, Cambridge, MA.