Item Descriptions Add Value to Plans

Will Fitzgerald and R. James Firby

Intell/Agent Systems
1840 Oak
Evanston, Illinois 60201
w-fitzgerald@nwu.edu, firby@cs.uchicago.edu

Abstract

Item descriptions add value to plans. Item descriptions are descriptions about individual objects (or sets of objects) in the world that can be used within plans for reasoning about actions on the objects. Typical reasoning tasks include dereferencing the description to real objects, disambiguating multiple real objects based on the description, and tracking the status of desired goals. We describe the evolution of our thought on a language for item descriptions, and describe some open questions that remain.

Introduction

One of the delightful things about the AAAI Symposium Series is the chance to discuss work in progress. Because the work usually remains in progress, what one reports on for the final version of the paper is often different from what one originally submitted. In this paper, we attempt something slightly different. We present the original submission, and then annotate it with comments about what was right about it, what was wrong, and what some open questions are that are left. Thus, we provide the original submission and then the comments.

The Original Submission

Traditional AI planning systems were concerned more with the ordering of plan operators than with the objects these operators acted upon. These systems assumed (usually without stating it) that the plan execution system could easily map from an internal, symbolic representation to the object represented in the real world. For example, if a planning system created the plan step `(pickup block-a)`, a one-to-one mapping existed between the symbol `block-a` and some block in the real world. It also assumed that the plan execution system could immediately sense and act on that block.

This is clearly an oversimplification; the real world does not provide an immediate connection from objects to a planning system's internal representations. This has been recognized for some time. For example, Agre and Chapman (1990), argue for a radically situated planning system in which object representations are strictly deictic. In their terms, this means an active, functionally motivated causal relationship between an agent and an object in the world, such as `the-bee-i-am-following`. Another approach is to distinguish between internal representations and sensor names (Firby, 1989). As an agent moves around in the world, it can use the information it receives from its sensors to disambiguate objects it senses into its internal representations.

What is missing from these approaches is the value of giving planning agents access to descriptions of objects in the world in addition to representations of objects in the world. Plans typically describe how to act on objects that are currently believed to have some properties. We suggest that it is often powerful to write plans to act on objects that meet a description of those properties.

For example, an agent might have a plan to paint all red fuel drums blue. If the agent acts only on what it currently believes about objects in the world, then it may miss some fuel drums, because it does not know enough about them to justify action. If, however, an agent carries with it the description of the properties, then it can act to find out more about objects in the world. For example, an agent might know that a particular object was a fuel drum, but not its current color; or it might know that it is red, but not what kind of object it is. Without access to a description of the objects upon which an agent is to act, an agent cannot tell whether it needs to get more information, act anyway, avoid action, etc.

An item description carries with it two types of information. First, it describes the kind of object that is being described (for example, that the object is a member of the set of *fuel drums*). Second, it describes the qualities of the object being described (for example, that the *color* is *red*). The information about qualities can be missing, of course (for example, an agent could look for any fuel drum). Similarly, the information about kind can be missing (for example, an agent can look for objects that are red in color). In our current syntax, this information is defined using *describes* and *slot-value* propositions. So the description underlying "red fuel drum" is:

```
(describes object-description-n fuel-drum)
(slot-value object-description-n color red)
```

In conjunction with propositions of the sort

```
(instance-of object-m fuel-drum)
(color object-m red)
```

different plans can be written which depend on either the item description or beliefs about the object. Plans can then depend on combinations of the item descriptions and beliefs about objects. For example, it is now possible to express, "if you're asked to paint fuel drums red, and the object is already red, don't paint it." That is, in the context:

(and (describes ?description fuel-drum)
 (slot-value ?description color red)
 (color ?object red)

where `?description` and `?object` are presumably bound in this context.

Item descriptions have three more useful qualities. First, recording item descriptions allows an agent to recognize that it is being requested to act on the same type of thing as it was before (for example, "you just asked me for a red fuel drum, so I'm not going to comply with the request"). Second, they can mimic higher order propositional logical propositions within a weaker, frame-based logic, so one can assert, for example, John believes *p*, without being forced to assert *p*. Third, they map closely onto natural language descriptions of objects. When an agent is asked (for example) to paint, *in natural language*, the red fuel drums, what is being asked maps closely to the item descriptions we have described, not just propositions about objects. This is especially true when linguistic phenomena such as definite and indefinite reference or anaphora are taken into consideration.

What we are arguing for is the use of information from three sources. First, there are sensing data from the real world such as whether the vision system senses a blue object. Second, there are stored beliefs about objects such as whether the memory system has recorded this object as being blue in color. Third, there are what we believe to be a distinct information source, item descriptions, such as whether the memory system has recorded that what is wanted is a blue object.

Comments on the Original Submission

There were several things that were right with the original submission, several things that are clearly wrong, and several questions left open. We discuss these in turn.

What was right

The basic premise of the first submission, that "item descriptions add value to plans" is fundamentally right. In order for an agent to be able to judge among similar objects in the world, or to communicate to other agents in the world the differences among similar objects, or to track the state of desired end goal states, the agent will need some type of description language for those objects.

What was wrong

There were a few things wrong with the original submission. It ignored the complexities that arise when one has an item description language that essentially mimics the knowledge representation language. It overlooked the generativity would result from using the same language for item description that one uses for knowledge representation.

Unexpected complexities. By using an item description language that essentially mimicked the knowledge representation language, unexpected complexities resulted. In particular, the plans needed for agents to act in the world required special care to consider whether the conditions under which they applied were "real" (that is, reflected in the state of the knowledge of the world) or "descriptive" (that is, using item descriptions).

For example, consider a plan to take an object to a location. Essentially, this is a two-place operator: the plan takes the "object" and the "location" as arguments. What we hoped to do with item descriptions is to make it easy to write plans of the following sort:

Transport(*object, location*):
If the agent knows what the *object* is,
 move to the location of the instance of *object*;
 otherwise, ask where to find an instance *object*.
Pick up the instance of the *object*,
If the agent knows where the *location* is,
 move to the place described by the *location*,
 otherwise, ask where to find the *location*.

This pseudo-code of a plan is for example only; it's not necessarily the best way to achieve the goal of taking something to a location. This plan has two arguments: are they item descriptions or pointers to a packaged knowledge representation? It's complex to know which it should be, and we found ourselves writing plans of the sort:

If *object* is an item description,
find out *x* about the *object*;
otherwise, find out *y* about the *object*.

where x and y were typically very similar predicates—usually finding out the same information, just on different types of representation. For example, we might write:

If *object* is an item description,
then if it *item-isa* halon-drum, do *z*;
otherwise, if it *isa* halon-drum do *z*.

where *isa* describes an instance/subclass to class relationship, and *item-isa* is true if the value of a *describes* predicate describes an instance/subclass to class relationship. For example, given `(instance-of obj-47 halon-drum)` and `(describes id-3 halon-drum)`, `(isa obj-47 halon-drum)` and `(item-isa obj-47 halon-drum)` would both be true.

This makes for very complex, difficult to comprehend plans. We found even ourselves befuddled by the plans we were writing. The plans become especially complex when multiple arguments are used, of course, because plan branches have to be repeated for each clause. It was also complex to pass the item descriptions or knowledge representations to subplans, for one had to be very careful to pass the right type of representation.

Missing Generativity. The obverse to this complexity is the lack of generativity of these plans. It became clearer to us that what we were calling "knowledge representation" and "item descriptions" were essentially the same thing. To write a plan such as the **Transport** plan shown above is just to use standard methods to represent the knowledge underlying the plan.

The moral. We desired to include item descriptions in plans so that agents could do (moderate) inference on the descriptions for disambiguation and goal tracking. We ended up with a cumbersome double representation scheme, which brought us back to using a standard single representation for facts about objects the world and descriptions of objects in the world. Currently, then, we are using a single representational scheme (Fitzgerald, Wiseman, & Firby 1996).

Still, what we set out to accomplish with item descriptions remains. In a dynamically changing world, it is often important to remember what's been asked for in general terms, and not to commit prematurely to dereferencing what's being asked for. To use the previous example, an agent might be asked to paint all green drums red. It might be important to remember that one was asked to paint just those drums that were green. By the time it gets to painting a particular drum, another agent might have painted it red already, and now it doesn't have to. On the other hand, a drum whose color it didn't know might turn out to be green when the agent arrives, and so it knows to paint it red.

The moral, then, is this:

Represent the world, but treat your representations lightly.

It is important for an agent to treat its own representations of the world as likely to go out of date in a changing world.

Open Questions

There are some major questions that are raised when moving to this common representation scheme.

- Given two (sets of) descriptions about an object, when can you license believing they refer to the same object?

This is the *matching problem*. Consider the painting robot again. It senses something drum-like, that is red, and has another description of a red drum. Does this description match the other description? Is it the same?

- Given two (sets of) descriptions about an object that refer to the same object, how does one combine the descriptions without losing information?

This is the *merging problem*. The key problem here is this. You might want to combine the results of two sensors so you have a larger set of beliefs about a particular object. But, there may be plans which depend on one of the descriptions, which one does not want merged so deeply into another representation that it can't be recovered. We need to retain both the original representations as well as the merged representation.

- When should representations be dereferenced to objects?

This is the *dereferencing timing problem*. On the one hand, one doesn't want to constantly expend computational energy making inferences on representations. On the other hand, one doesn't want to lose information along the way.

- How much representation should be constructed and how should it be used?

It is important that plans be tractable, and so we don't want to allow unlimited inferencing to occur. Are there ways to write planners so they know just how much representation should be used? In our natural language work, this is driven by the "natural" ways there are to talk to an agent—if there is a natural, normal way to express something, we'd like to be able to represent that. The danger remains, though, of writing plans (or dynamically creating them) that can't be executed in a reasonable time frame.

Conclusions

We have tried to show that item descriptions add value to plans. These item descriptions allow an agent to make reasonable inferences about the objects it acts on; for example, to judge among ambiguous objects or to communicate specific goals. We currently believe that item descriptions are just representations of knowledge about objects in the world. We say this with two provisos. First, agents must take their own beliefs about objects in the world lightly, being ready to revise them as the world changes. Second, agents must be able to carry descriptions as they execute plans, and they must be able communicate these descriptions to other agents.

Acknowledgments

The authors would like to thank John Wiseman for his invaluable assistance in the preparation of this paper.

References

Agre, P. E. and Chapman, D. 1990. What are plans for? *Robotics and Autonomous Systems* 6:17–34.

Firby, R. J. 1989. Adaptive Execution in Complex Dynamic Worlds. Ph.D. diss. Dept. of Computer Science, Yale University.

Fitzgerald W.; Wiseman, J; Firby R.J. 1996. Dynamic Predictive Memory Architecture: Conceptual Memory Architecture. Technical Report #1, Intell/Agent Systems, Evanston, IL.