Universal Planning: An (Almost) Universally Bad Idea

Matthew L. Ginsberg

Computer Science Department
Stanford University
Stanford, California 94305

Universal Planning: An (Almost) Universally Bad Idea

Abstract

Several authors have recently suggested that a possible approach to planning in uncertain domains is to analyze all possible situations beforehand, and to then store information about what to do in each. The result of this is that a system can simply use its sensors to examine its domain, and then decide what to do by finding its current situation in some sort of a table.

The purpose of this note is to argue that even if the compile-time costs of the analysis are ignored, the size of the table must in general grow exponentially with the complexity of the domain. This makes it unlikely that this approach will be able to deal with problems of an interesting size; one really needs the ability to do some amount of inference at run-time. Alternatively put, an effective approach to acting in uncertain domains cannot be to look and then leap, but must always be to look, to *think*, and only then to leap.

1 There are too many universal plans

In order for us to present a sharp criticism of the approach that we are describing as "universal planning," we need to begin by giving a sharp definition of it.

The key idea in all of this work is that an agent is working to achieve some goal, and that in order to determine what to do next in the pursuit of this goal, the agent finds its current situation in a large table that informs him of the correct action to take.

Of course, the action suggested by the table might be simply, "Think about your current situation and decide what to do next." This is in many ways representative of the *conventional* approach to planning – what distinguishes universal plans from conventional plans is that the action suggested by a universal plan is always a primitive one that the agent can execute immediately [1, 2, 3, 5, 6, 7].

Definition 1.1 *A* universal plan *is an arbitrary function from the set of possible situations S into the set of primitive actions A.*

We will assume throughout this paper that there is some fixed universal plan u that describes the *intended* behavior of the agent being constructed; the question that we wish to investigate is whether or not it is practical to provide the agent with an explicit description of u in its entirety, as opposed to some method (such as "invoke the planner") by which $u(s)$ can be computed if the agent finds himself in the situation s.

Given Definition 1.1, we can draw distinctions between the intentions of the various authors working with universal plans in one form another. Schoppers, who coined the term "universal plan," seems to feel that it will be possible to equip an agent with a completely accurate description of a universal plan for acting in its particular environment, while other authors (Agre/Chapman [1], Drummond [2] and Nilsson [5]) concede that only some approximation to the universal plan u will be available. Rosenschein and Kaelbling [3, 6] appear to agree with Schoppers, although we will argue shortly that this would be a misrepresentation of their beliefs.

There is another choice to be made if an approximate universal plan is used. Agre/Chapman and Drummond assume that the agent using the approximate universal plan will continue to use it blindly in all situations, so that the agent's actions will in some instances be unrelated to its overall goals, while Nilsson appears to assume that the agent will recognize these situations and do some additional planning when they are encountered. Of course, the only effective way to recognize situations in which the approximate plan differs from the optimal one is to label them explicitly; in other words, the "action" specified by the approximate plan must be to invoke the planner.

Here then, are the three possible approaches to universal planning:

1. Assume that it will be possible to explicitly represent the universal plan giving the agent's optimal action as a function of situation [7]. We will argue that this assumption is unlikely to be valid in practice.
2. Concede that a universal plan that always suggests a primitive action to be taken will occasionally suggest an action that is not related to the achievement of the agent's goals [1, 2]. We will argue that this approach is only viable for precognitive activities such as locomotion.
3. Concede that a universal plan that is used to describe cognitive behavior must occasionally suggest that a planner be invoked in order to decide what to do next [5]. In this case, universal planning is being used only as a method for storing precompiled plans that the agent has available at runtime.

All of the arguments that we will make rest on a careful examination of the techniques that have been suggested to actually represent a universal plan as data. Schoppers, for example, suggests that a decision tree be constructed [7], while Nilsson and Rosenschein each suggest that some sort of Boolean circuit be used [5, 6]. We will restrict our attention to this latter approach, as it is somewhat the more general of the two.[1]

By a *gate*, we will mean an element of the Boolean circuit being used to describe some particular universal plan. We will assume that every gate has two inputs and a single output; note that since each gate has four possible input combinations, there are 16 possible such gates.

[1] An expression of the form, "If p then q else r," can be encoded by having the signal corresponding to p drive the output corresponding to q, and passing this signal though an inverter before having it drive the output corresponding to r.

Now suppose that our domain includes n binary sensors that are used to examine the current situation, and that there are a primitive actions that the planner might take. By a *circuit of size* g, we will mean an arbitrary collection of g gates, connected in arbitrary ways to each other, to the outputs of the n sensors, and to a actuators that correspond to decisions to take each of the primitive actions. The only restriction we place on the circuit is that none of the outputs be connected to each other.[2]

Finally, we will make two assumptions about the sensors. The first is that they are independent, so that every combination of sensor readings corresponds to a possible situation; the second is that the information given by the sensors is sufficient to determine the action that the robot should take. These two assumptions in combination allow us to assume that there are 2^n distinct situations for which the universal plan must specify an action.

Proposition 1.2 *There are* $(2^a)^{(2^n)}$ *distinct universal plans. There are at most* $16^g(g+n)^{2g+a}$ *distinct circuits of size* g.

Proof. This is just a matter of counting. The number of functions (i.e., universal plans) from a set of size 2^n (the set of situations) into a set of size 2^a (the set of possible combinations of actions to be taken) is $(2^a)^{(2^n)}$.

To describe a circuit consisting of g gates, we must specify, for each gate, which of the 16 possible types it is and, for each of the $2g+a$ input signals (one for each of the two inputs to each gate and one for each of the actuators), which of the $g+n$ signals drives it (one for each gate's output, and one for each sensor signal). $\square$

Proposition 1.3 *The number of gates that can be expected to be needed in order to describe a universal plan in a domain with* n *sensors is on the order of* 2^n.

Proof. This is simply the result of solving

$$\sum_{i=1}^{g} 16^i(i+n)^{2i+a} = (2^a)^{(2^n)}$$

for g. $\square$

In other words, the number of gates needed in order to describe a universal plan will in general grow exponentially with the size of the domain. Similar arguments can be made for Schoppers's decision-tree formulation.

Note that Proposition 1.3 holds independent of the value of a, the number of primitive actions that our agent can take. It might appear that even though it is impractical to construct a circuit that exactly captures our original universal plan u, one might construct one that describes some alternative universal plan u' that takes an action that is approximately the same as the one described by u (perhaps the robot moves 1.5 inches instead of 1.6 in some situation). But we see that this, too, is in general impractical; if we lump all of our actions into a single one, so that $a = 1$ and the circuit is simply trying to tell us whether we need to act or not to act, we still need 2^n gates to make the decision.

[2]Note that we are allowing feedback loops, but we will nevertheless assume that the circuit responds effectively instantly to its inputs. This is reasonable, since the circuit does not contain any explicitly time-dependent elements such as latches.

2 Perhaps most of the universal plans don't matter

If Schoppers (and Rosenschein) are to realize their intention of capturing the entire universal plan needed to act in some domain, they must argue that this universal plan can for some reason be described using far fewer gates than one would expect on average. This is certainly possible *in principle*, since the various domain sensors could be sensing things like, "You should take action 1 now," and so on.

Rosenschein and Kaelbling [3, 6] do just this. Although the action component of their proposed architecture for intelligent agents is indeed expected to contain an exact description of the universal plan u, the perception component explicitly allows for arbitrary manipulations of the agent's primitive sensory inputs. These manipulations might well involve symbolic computations that do the planning themselves, turning the primitive inputs into things like, "Take action 1 now."

Schoppers, on the other hand, assumes that it will be possible to encode a universal plan without changing the sensory inputs. In simple domains such as the blocks-world examples considered in [5, 7], this assumption turns out to be valid – the available sensors provide information that is very nearly that which is needed to determine what action to take. (It is not surprising, for example, that a sensor telling you whether or not a particular block is clear is providing information that can be turned into an action decision with a minimum of effort.)

In general, however, things are not so simple. Consider the blocks-world problem of stacking nine blocks so that they spell "fruitcake." The reason that this problem is difficult for universal planners is that in a blocks world with nine blocks, there will be in excess of 9! possible situations for which we need to specify actions; this is impractical unless the available sensors somehow give us more useful information than simply, "The block labelled k is clear."

Nilsson has managed to construct a universal plan for achieving this goal – but only by introducing new sensors that were designed specifically for this problem [4].[3] Specifically, Nilsson assumes the existence of sensors that determine as primitives how much of "fruitcake" has been constructed so far and which block must be moved first in order to eventually clear the top of a given block. Given these sensors, it is hardly surprising that Nilsson can construct a universal plan for the goal in question!

More complex examples are worse still. The plan for using a tactical combination to win a chess game might be something like, "I'm going to attack his queen, and it'll have to go here, and then I can do *this* and he'll have to respond like *that*, and then I'll be able to checkmate him."

Well, maybe the opponent won't move his queen. A human player need not consider this possibility, since he can simply assume that he will be able to win the game after the queen is captured. But a universal planner can take no such luxuries – it will need to plan, in the most minute detail, how it will deal with *every* combination of moves (good and bad) by the opponent. In addition, since the move that should be selected in one chess position is

[3]Note how similar this is to the perceptual modifications allowed by Rosenschein and Kaelbling.

likely to be very different from one that would be selected in a "nearby" position (perhaps where one of the pieces has been displaced by a square or changed color), there is no reason to believe that it will be possible to express the universal plan for this problem using a reasonable number of gates, decision nodes, or what have you.

What about other planning situations? In general, I am extremely pessimistic, for reasons that I will describe shortly. But let me point out that pessimism or otherwise, Schoppers is tacitly making an extremely strong claim – that the planning problems encountered by autonomous agents *will* be of the type for which universal plans can be represented. This claim needs to be defended, and he has yet to do so.

Why am I pessimistic? Consider for a moment Schoppers's original argument *in favor* of universal plans. He described a blocks-world domain in which there was a baby wandering around, occasionally knocking down the blocks or relocating them. In an unpredictable domain, he argued, it was important to have planned for unexpected contingencies in advance.

Suppose, then, that we are trying to get two of Schoppers's blocks on top of one another in a particular location. We get the first block and set it down, and while we are getting the second block, the baby moves the first one. So we put the second block down, get the first block back where it belongs, and go back to the second block.

Unfortunately, the baby now moves the first block again. Sigh. So we put the second block down, go back to the first block ...

Or do we? In actuality, we are likely to do something fairly novel. We might try and carry the second block with us, so that we can bring both blocks to the target location simultaneously. If we can't do that, we might take the *second* block to the target location first, hoping that the baby wouldn't move *it*. We might find the baby and lock him in the bathroom while we moved the blocks around.

The point is that the decision is a complex one. What we decide to do depends in subtle ways on the values of *many* sensors, not just the few indicating where the blocks are and whether they are clear. Where is the baby? Whose baby is it? After all, we can discipline our own baby more effectively than someone else's. So even in this very simple example, we see that deciding what to do may be a very sensitive function of the sensory information available. What reason is there to believe that the goal is one for which a universal plan can be described in non-exponential form?

Agre, Chapman and Drummond all appear to be in general agreement with the view that it is in general impossible to describe a universal plan in exact detail; we will discuss the consequences of this in the next section. Before doing so, however, we should discuss the implications of Proposition 1.2 on computational planning in general.

The reason that the proposition is of interest is that a computer running a planning program is, after all, a collection of gates that have been wired together in some fashion. Is there any reason to believe that the arguments we have made cannot be extended to imply that planning is in general an impossible task?

I would argue that there is, since we have an intuitive sense that it is possible to solve planning problems by "figuring out what to do." It is not unreasonable to hope that it will be possible to write programs that also "figure out what to do," although one would expect that such programs will suffer from the same difficulties we do when they encounter complex and unexpected situations.

Note, however, how delicate this argument is – the *only* reason we have to believe that a computer will be capable of solving planning problems is because we believe it will be possible for a machine to emulate our own solution methods. This assumption depends, in part, on our underlying belief that our environment behaves in predictable ways; the argument would obviously break down if there were no rhyme or reason to things.

3 Approximate universal plans

Instead of the "exact" universal plans discussed in the last section, Agre/Chapman and Drummond assume that their agents will work with "approximate" universal plans that sometimes suggest suboptimal actions. Agre and Chapman argue in [1] that this is a suitable approach if an autonomous agent is to perform tasks such as playing a video game.

This seems entirely reasonable. The advantage of approximate universal plans is that they allow agents to react extremely quickly to their surroundings; the disadvantage is that their inflexibility makes it extremely difficult for an agent using such a plan to expend limited resources in order to improve its performance.

For noncognitive tasks, this seems a fair choice. But for tasks where one expects that an agent *will* be using its cognitive abilities to improve its performance, an approximate universal plan will in general be overly restrictive. (Indeed, the ability to enhance performance by expending additional mental resources seems to be a reasonable *definition* of the distinction between cognitive and noncognitive behavior.)

In [5], Nilsson suggests that this difficulty can be overcome by having one of the actions suggested by a particular universal plan be "plan." This is clearly a way around the problem, but what Nilsson is doing is reducing the universal plan (in his case, a particular data structure he calls an "action net") to a representation technique for conditional plans that are already known to the agent at runtime.

If this is done, then the contribution made by Nilsson's work must be evaluated in this light. The work on action nets is addressing a data compaction problem, as opposed to previously recognized issues in planning or reasoning about action. Nilsson needs to present sharp arguments to the effect that:

1. Action nets do provide compact representations of the precompiled plans available to an autonomous agent, and
2. This is an important issue. In other words, Nilsson must convince us that the agent will have so many precompiled plans that they *need* to be compacted, and that addressing this issue is of comparable importance to solving existing problems in planning, such as developing computationally viable descriptions of action in a particular domain or inverting those descriptions to construct a plan for acting when no precompiled plan can be applied.

Unfortunately, the only result in this direction is due to Schoppers, and indicates that a decision-tree description can reduce by some 30% the amount of time needed to find the precompiled plan assigned to the situation in which an agent finds himself [8]. This is not that large a savings; in addition, it seems likely that the principal cost of deciding what to do in an arbitrary situation is not finding that situation in a table of precompiled plans, but is rather deciding what to do if no such plan is available.

4 Summary

The principal point that this paper makes is that it is impractical for an agent to precompute its response to every situation in which it might find itself. We have not argued, and would not attempt to argue that it is impractical to plan for all of the reasonably likely contingencies, or even for all of the vaguely likely ones. In this situation, one must either expect to replan if a sufficiently unlikely or unexpected situation arises, or expect the agent's abilities to be permanently limited by design decisions that were made when its approximate universal plan was constructed.

Acknowledgement

This work has been supported by General Dynamics and by NSF under grant number IRI89-12188. I would like to thank Andrew Baker, Ken Fertig, Leslie Kaelbling, Nils Nilsson, Stan Rosenschein, Marcel Schoppers and David Smith for many enlightening discussions.

References

[1] P. Agre and D. Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272, 1987.

[2] M. Drummond. Situated control rules. Technical report, NASA Ames Research Center, Moffett Field, CA, 1988.

[3] L. P. Kaelbling. Goals as parallel program specifications. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 60–65, 1988.

[4] N. J. Nilsson, 1988. Unpublished Stanford memo.

[5] N. J. Nilsson. Action networks. Technical report, Stanford University, 1988.

[6] S. J. Rosenschein and L. P. Kaelbling. The synthesis of machines with provable epistemic properties. In J. Y. Halpern, editor, *Proceedings of the 1986 Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 83–98, Los Altos, CA, 1986. Morgan Kaufmann.

[7] M. J. Schoppers. Universal plans for reactive robots in unpredictable domains. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 1039–1046, 1987.

[8] M. J. Schoppers. *Representation and Automatic Synthesis of Reaction Plans.* PhD thesis, University of Illinois at Urbana-Champaign, Urbana-Champaign, Illinois, 1989.