# Is there any Need for Domain-Dependent Control Information?

**Matthew L. Ginsberg*** and **Donald F. Geddis†**
Computer Science Department
Stanford University
Stanford, California 94305
ginsberg@cs.stanford.edu

## Abstract

**No.**

## 1 Introduction

The split between *base-level* and *metalevel* knowledge has long been recognized by the declarative community. Roughly speaking, base-level knowledge has to do with information about some particular domain, while metalevel knowledge has to do with knowledge *about* that information. A typical base-level fact might be, "Iraq invaded Kuwait," while a typical metalevel fact might be, "To show that a country $c$ is aggressive, first try to find another country that has been invaded by $c$."

Base-level information is of necessity domain-dependent, since the facts presented will involve the particular domain about which the system is expected to reason. Metalevel information, however, can be either domain-dependent (as in the example of the previous paragraph and as typically described in [Silver,1986]), or domain-independent. Typical domain-independent metalevel rules are the cheapest-first heuristic or the results found in Smith's work on control of inference [Smith,1986, Smith and Genesereth,1985, Smith *et al.*,1986].

In this paper, we make an observation and a claim. The observation is that there are in fact two distinct types of metalevel information. On the one hand, one can have metalevel information about one's base-level knowledge itself; on the other, one can have control information about what to *do* with that knowledge.

This is a distinction that has typically been blurred by the AI community; we intend to focus on it in this paper. We will refer to the second type of metalevel knowledge as *control* knowledge, and will refer to knowledge about knowledge (and not about what to do with that knowledge) as *modal*. We choose this term because sentences expressing modal knowledge typically involve the use of predicate symbols that have other sentences as arguments. Thus a typical modal sentence might be, "I know that Iraq invaded Kuwait," or "I don't know of anyone that Kuwait has invaded." Note that the information here doesn't refer to the domain so much as it does to our knowledge *about* the domain; nor is it control information telling us what to do with this knowledge. It simply reports on the state of our information at some particular point in time. In general, we will describe as modal all information describing the structure of our declarative knowledge in some way.

The claim we are making – and we intend to prove it – is that there is no place in declarative systems for domain-dependent control information. Rather, we suggest that every bit of information of this sort is in fact a conflation of two separate facts – a domain-independent control rule and a domain-dependent modal fact telling us that the rule can be applied. A similar observation has already been made by David Smith:

> Good control decisions are not arbitrary; there is always a reason why they work. Once these reasons are uncovered and recorded, specific control decisions will follow logically from the domain-independent rationale, and simple facts about the domain. [Smith,1985, p. 12]

This is an observation with far-reaching consequences, including the following:

1. There is no need for a "metametalevel" or anything along those lines. Domain-independent control information need not be refined using higher-order information. A similar conclusion has been reached in a decision-theoretic setting by Barton Lipman [Lipman,1991].

2. Current work on learning need not focus on the development of new control rules (as in [Minton *et al.*,1989]), but should instead focus on the development of new base-level or modal information. This may make the connection to existing ideas (such as caching) somewhat easier to exploit

*This work has been supported by the Air Force Office of Scientific Research under grant number 90-0363, by NSF under grant number IRI89-12188 and by DARPA/RADC under grant number F30602-91-C-0036.

†Supported by a National Science Foundation Graduate Fellowship.

and understand.

3. Current work on the control of inference should restrict its focus to domain-independent techniques. There is much to be done here; Smith's work only scratches the surface. At a minimum, general-purpose information can be extracted from the domain-specific control rules currently incorporated into many declarative systems.
4. The recognition that the control of reasoning should proceed by applying domain-independent rules to domain-dependent modal and base-level information can enhance the power of systems that control reasoning in this way. This is a consequence of the fact that many declarative databases already contain such domain-dependent information without exploiting its implications for search control. In [Ginsberg,1991], it is suggested that the real role of nonmonotonic reasoning in AI is to focus inference, and that suggestion is typical of what we are proposing here.

The outline of this paper is as follows: We begin in the next section with an example, showing the replacement of a specific control rule by a domain-independent one using domain-dependent modal information. We then go on to show that this procedure generalizes with minimal computational cost to all domain-specific control information.

The general construction is extremely weak, and Section 3 examines a more powerful application related to the work in [Elkan,1990, Ginsberg,1991, Minton *et al.*,1989]. Concluding remarks are contained in Section 4.

## 2 The basic result

Consider the following declarative database, written in PROLOG form:

```
hostile(c) :- allied(d,c), hostile(d).
hostile(c) :- invades-neighbor(c).
allied(d,c) :- allied(c,d).
invades-neighbor(Iraq).
allied(Iraq,Jordan).
```

Informally, what we are saying here is that a country is hostile if it is allied to a hostile country, or if it invades its neighbor. Alliance is commutative, and Iraq has invaded its neighbor and is allied with Jordan.

Now suppose that we are given the query $\texttt{hostile}(\texttt{Jordan})$; is Jordan hostile? In attempting to prove that it is, it is important that we eventually expand the subgoal $\texttt{invades-neighbor}(\texttt{Iraq})$ rather than only pursue the infinite chain of subgoals of the form

$$\texttt{hostile}(\texttt{Jordan}), \texttt{hostile}(\texttt{Iraq}), \texttt{hostile}(\texttt{Jordan}), \texttt{hostile}(\texttt{Iraq}), \ldots$$

We might describe this in a domain-dependent fashion by saying that subgoals involving `invades-neighbor` should be investigated before subgoals involving `hostile`. (There are obviously many other descriptions that would have the same effect, but let us pursue this one.)

Now consider the following geometric example:

```
acute(t) :- congruent(u,t), acute(u).
acute(t) :- equilateral(t).
congruent(u,t) :- congruent(t,u).
equilateral(T1).
congruent(T1,T2).
```

A triangle is acute if it is congruent to an acute triangle or if it is equilateral. Triangles $T_1$ and $T_2$ are congruent, and $T_1$ is equilateral. Is $T_2$ acute?

This example is different from the previous one only in that the names of the predicate and object constants have been changed; from a machine's point of view, this is no difference at all. It follows from this that if the control rule delaying subgoals involving the predicate `hostile` is valid in the initial example, a rule delaying subgoals involving `acute` will be valid in the new one.

What we see from this is that the control rules are not operating on base-level information so much as they are operating on the *form* of our declarative database. To make this remark more precise, we need to discuss the structure of the proof space a bit more clearly. We propose to do this by actually axiomatizing a description of this proof space. This axiomatization will refer to the structure of the proof space only and will therefore be modal information in the sense of the introduction; since nothing will be said about how to search the proof space, no domain-dependent control information will be used.

The language we will use will label a node in the proof space by a set $\{p_1, \ldots, p_n\}$, where all of the $p_i$ need to be proved in order for the proof to be complete. A node labelled with the empty set is a goal node.

We will also introduce a predicate `child`, where $\texttt{child}(n_1, n_2)$ means that the node $n_2$ is a child of the node $n_1$. Thus in the original example, we would have

$$\texttt{child}(\{\texttt{hostile}(c)\}, \{\texttt{invades-neighbor}(c)\}) \quad (1)$$

saying that a child of the node trying to prove that $c$ is hostile is a node trying to prove the subgoal that $c$ has invaded its neighbor.

We can go further. By taking into consideration all of the facts in the database together with the inference method being used, we can delimit the extent of the `child` predicate exactly. This eventually leads to a large disjunction, one term of which is given by (1):

$$\begin{aligned}
&\texttt{child}(m, n) \equiv \\
&\quad [m = n \cup \{\texttt{invades-neighbor}(\texttt{Iraq})\}] \vee \\
&\quad [m = n \cup \{\texttt{allied}(\texttt{Iraq}, \texttt{Jordan})\}] \vee \qquad (2) \\
&\quad \exists S.[m = S \cup \{\texttt{allied}(x, y)\} \wedge \\
&\qquad n = S \cup \{\texttt{allied}(y, x)\}] \vee \ldots
\end{aligned}$$

The above expression refers to the predicate and object constants appearing in the initial database, in this case **hostile**, **invades-neighbor**, **allied**, **Iraq** and **Jordan**. We can abbreviate the large expression appearing in (2) to simply

$$\mathtt{type}_{37}(\mathtt{hostile}, \mathtt{invades\text{-}neighbor}, \mathtt{allied}, \mathtt{Iraq}, \mathtt{Jordan}) \tag{3}$$

where we are using the subscript to distinguish databases of this form from databases of some other form. Note that (3) is in fact a consequence of the form of the information in our database, so that we can either derive (3) before applying a (domain-independent) control rule in which $\mathtt{type}_{37}$ appears, or derive and stash (3) when the database is constructed. The second example is similar, allowing us to derive

$$\mathtt{type}_{37}(\mathtt{acute}, \mathtt{equilateral}, \mathtt{congruent}, T_1, T_2)$$

At this point, the hard work is done – we have formalized, via the `type` predicate, our observation that the two databases are the same. The control rule that we are using is now simply

$$\mathtt{type}_{37}(p_1, p_2, p_3, o_1, o_2) \supset \mathtt{delay}(p_1) \tag{4}$$

indicating that for databases of this type, with the $p_i$ being arbitrary predicate constants and the $o_i$ being arbitrary object constants, subgoals involving the predicate $p_1$ should be delayed. Of course, we still need to interpret `delay` in a way that will enable our theorem prover to make use of (4), but this is not the point. The point is that (4) is no longer a domain-specific control rule, but is now a domain-independent one. Any particular application of this domain-independent rule will make use of the modal information that a given database satisfies the predicate $\mathtt{type}_{37}$ for some particular choice of constants. As we have already noted, this modal information can either be derived as the theorem prover proceeds or can be cached when the database is constructed.[1]

**Proposition 2.1** *Any domain-dependent control rule can be replaced with a domain-independent control rule and a modal sentence describing the structure of the search space being expanded by the theorem prover. The computational cost incurred by this replacement is that of a single inference step, and the domain-independent control rule will be valid provided that the domain-dependent rule was.*

**Proof.** Given a specific database of a particular form, it is clearly possible to delimit in advance the nature of any single inference, thereby obtaining an expression such as (2); this expression can then be abbreviated to a `type` expression such as (3). The domain-dependent control rule can now be replaced by a domain-independent one as in (4).

The incremental cost of using the domain-independent control rule instead of the domain-dependent one will be the expense of checking the antecedent of a rule such as (4); if we choose to cache the modal information describing the structure of the database, only a single inference will be needed. In addition, since the validity of a control rule depends only on the nature of the associated search space and never on the meanings of the symbols used, it follows that the domain-independent control rule will be valid whenever the domain-dependent one is. □

## 3 A more interesting example

Although correct, the construction in the previous section is in many ways trivial, since in order to transfer a control rule from one domain to another we need to know that the two search spaces are identical. Indeed, this is the only circumstance under which we can be assured that the rule will remain valid when the domain changes.

The reason that our earlier construction is interesting, however, is that the intuitive arguments underlying domain-dependent control rules do *not* typically depend on the complete structure of the search space. In this section, we will consider a more typical example due to David Smith.

The example involves planning a trip to a distant location; let us suppose from Stanford to MIT.[2] The domain-dependent rule Smith presents is the following: When planning a long trip, plan the airplane component first. Why is this?

There are two reasons. Suppose that we form the tentative high-level plan of driving from Stanford to San Francisco airport, flying from there to Logan, and then driving to MIT. The decision to plan the airplane component of the trip first is based on the observations that:

1. The three legs of the journey are probably noninteracting. Except for scheduling concerns, the nature of our transportation to and from the airport is unrelated to our flight from San Francisco to Boston. It therefore makes sense to plan for each of the subgoals separately.
2. Airplane flights are tightly scheduled, whereas ground transportation is typically either loosely scheduled (because busses run frequently) or not scheduled at all (if we propose to drive to and from the airports involved). If we schedule the ground transportation first, we may be unable to find a flight that satisfies the induced constraints.

The observation that our three subgoals (drive to SFO, fly to Logan, drive to MIT) do not interact is

[1] For efficiency reasons, we might want to use the control rule only when the database is *known* to be of this type. This could be encoded by using a modal operator of explicit knowledge to replace (4) with a more suitable expression.

[2] Why anyone would actually want to *make* this trip escapes us.

little more than an application of the frame axiom, which says that once a subgoal has been achieved it will remain satisfied. (So that renting a car from Hertz will not magically teleport us from Boston to Miami; nor will driving to the airport cause United to go bankrupt.) What we are doing here is applying the control rule:

> When attempting to achieve a conjunctive goal, it is reasonable to attempt to achieve the conjuncts separately.

Elkan also makes this observation in [Elkan,1990].

It is a base-level fact – the frame axiom – that justifies our confidence in this domain-independent control rule.[3] The frame axiom justifies our belief that we will be able to achieve the subgoals separately, and therefore that planning for them separately is a computationally useful strategy. Thus Elkan's principle is in fact a special case of the approach that we are proposing; this observation is made in [Ginsberg,1991] as well, where it is suggested that the true role of nonmonotonic reasoning in AI generally is to enable us to simplify problems in this way.

The reason that we plan for the airplane flight before planning for the other two subgoals is similar. Here, we note that solving the subgoal involving the airplane flight is unlikely to produce a new problem for which no solution exists, while solving the subgoals involving surface transportation may. So we are invoking the domain-independent principle:

> When solving a problem, prefer inference steps that are unlikely to produce insoluble subproblems.

This domain-independent control information is applied to the domain *dependent* modal information that

$$\mathtt{fly}(\mathtt{SFO}, \mathtt{Logan}, t)$$

is unlikely to have a solution if $t$ is bound, while

$$\mathtt{drive}(\mathtt{Stanford}, \mathtt{SFO}, t)$$

is likely to have a solution whether $t$ is bound or not. Once again, we see that we are applying a domain-independent control rule to domain-dependent modal knowledge. Furthermore, the computational arguments in Proposition 2.1 remain relevant; it is no harder to cache partial information regarding the structure of the proof space (e.g., $\mathtt{fly}(\mathtt{SFO}, \mathtt{Logan}, t)$ is unlikely to have a solution if $t$ is bound) than it is to cache a complete description as in (3).

[3] In fact, this control rule is not completely domain-independent, since it applies to planning only. But it can easily be extended to the completely domain-independent rule that when attempting to solve a problem, if there is good reason to believe that the solution to a simpler problem will suffice, one should solve the simpler problem and then check to see if it is a solution to the original query.

An example due to Minton and appearing in [Minton *et al.*,1989] can be handled similarly. This example is from the blocks world, where Minton suggests that when planning to build a tower, expect to build it from the bottom up. We have chosen to discuss Smith's example in detail because only his rule is pure control information.[4] In Minton's case, the fact that you will *build* the tower from the bottom up is no reason to do the *planning* by beginning with consideration of the bottom block, although Minton appears to use the rule in this fashion. After all, it is easy to imagine domains in which the nature of the tower is constrained by the properties of the uppermost object it contains; in this case it might well be advisable to do the planning from the top down even though the *construction* will inevitably be from the bottom up. Smith's example does not share this ambiguity; it really is the case that one plans long trips by beginning with any airplane component.

We should be careful at this point to be clear about what distinguishes the examples of this section from those that appear in Section 2. The basic result of Section 2 was the following:

> By constructing control rules that appeal to the precise form of a declarative database and by caching modal information about the form of the database being used in any particular problem, control rules can always be split into domain-independent control knowledge and domain-dependent base-level or modal knowledge. Further, this split incurs minimal computational expense.

The upshot of the examples we have discussed in this section is the following:

> Although the control rules used by declarative systems might in principle be so specific as to apply to only a single domain, the domain-independent control rules in *practice* appear to be general enough to be useful across a wide variety of problems.

Whether or not this observation is valid is more an experimental question than a theoretical one; if the observation *is* valid, it reflects that fact that our world is benign enough to allow us to develop control rules of broad merit. The evidence that we have seen indicates that our world is this benign; all of the examples of control information that we have been able to identify in the AI literature (or in commonsense discourse) appeal to domain-independent control principles that have wide ranges of validity. Nowhere do we see the use of a control rule whose justification is so obscure that it is inaccessible to the system using it.

The results in [Etzioni,1990] also lend credence to this belief. Etzioni's STATIC system showed that determining whether or not a rule learned by PRODIGY

[4] As Smith himself has pointed out (personal communication).

was likely to be useful in practice could be determined by a static examination of the PRODIGY search space. Since Etzioni's system examines PRODIGY's search space only to determine whether or not it is recursive, we see once again a very general domain-independent rule (involving the need to avoid recursive problem descriptions) being used to construct apparently domain-specific control information.

## 4 Conclusion

Our aim in this paper has been to argue that reasoning systems have no need for domain-dependent control rules; rather, they apply domain-independent control techniques to domain-dependent modal and base-level information. In retrospect, the viability of this approach is obvious; what is interesting are the consequences of this view. These were mentioned in the introduction, but we would like to explore them in somewhat more detail here.

**Higher orders of meta** One of the difficulties with control reasoning thus far has been that it seems to require a "metametalevel" in order to decide how to do the reasoning about the control information itself, and so on.

Domain-independent control information eliminates this philosophical difficulty. Since the control information is completely general, there is no reason to modify it in any way. The implemented system must be faithful to this control information but need have no other constraints placed on it.

Another way to look at this is the following: Provided that no new control information information is learned, existing metalevel information can be "compiled" in a way that eliminates the need for runtime evaluation of the tradeoffs between competing control rules. The observations in this paper allow us to conclude that the new control information learned by existing systems is in reality base-level or modal information instead; viewing it in this way allows us to dispense with any reexamination of our existing control decisions. This reexamination is, of course, the "metametalevel" analysis that we are trying to avoid.

**Research on control of inference** There has been little work to date on domain-independent techniques for the control of search. Smith investigates the question in a declarative setting in [Smith,1986, Smith and Genesereth,1985, Smith *et al.*,1986], while authors concerned with constraint satisfaction have considered these issues from a somewhat different viewpoint [Dechter,1990, Dechter and Meiri,1989, Dechter and Pearl,1988, Ginsberg *et al.*,1990, and others].

The ideas in this paper suggest an obvious way in which these results can be extended. If the existing domain-dependent control information used by various systems does in fact rest on general domain-independent principles, these general principles should be extracted and made explicit. In many cases, doing so may involve extending the declarative language being used to include information that is probabilistic [Smith,1986, Smith and Genesereth,1985, Smith *et al.*,1986] or default [Elkan,1990, Ginsberg,1991] in nature; this can be expected to lead to still further interesting research questions.

**Domain-dependent information** The recognition that the domain-dependent information used by a declarative system is all base-level or modal has interesting implications in its own right.

The most immediate of these is the recognition that learning systems should not be attempting to learn control rules directly, but should instead be focusing on the base-level information the underlies them. Thus a travel-planning system should be trying to recognize that airline flights are scheduled while automobile trips are not, as opposed to the specific rule that one should plan around airline flights.

There are some immediate benefits from this approach. The first is that domain-independent control rules may be able to make use of domain-dependent information that has been included in the system for reasons having nothing to do with the control of search. In Section 3, for example, we saw that the frame axiom, through its implication that conjunctive subgoals can be achieved separately, had significant consequences in planning search. These consequences are obtained automatically in our approach.

A second benefit is that learned modal knowledge can be used by any domain-independent control rules included in the system. If domain-specific control rules are learned instead, other consequences of the modal knowledge underlying these rules will not be exploited.

Third, subtle interactions among domain-dependent control rules can now be understood in terms of simpler base-level information. Thus if my reason for going to MIT is to give a presentation of some sort, I will typically plan the time of the presentation first, only then scheduling the airplane and other parts of my travel. The reason is the base-level fact that the folks at MIT are likely to be even more tightly scheduled than are the airlines. The apparent conflict between the control rules, "Plan airplane flights first," and "Plan talks first," is recognized as tension between the base-level facts, "Airplane flights are tightly scheduled," and "Academic talks are tightly scheduled." Assuming that our declarative language is able to compare the truth values assigned to these two latter sentences, the conflict will be resolved in a straightforward fashion.

Finally, a focus on the ability to derive base-level information may allow us to address an aspect of human problem solving that has been ignored by the AI community thus far. Specifically, the results of the problem-solving effort themselves frequently impact our base-level expectations. As an example, after an hour spent fruitlessly searching for a proof of some theorem, my belief in the theorem may waver and I may decide to look for a counterexample. What I am doing here is changing my base-level beliefs based not on new information, but on the results of my theorem-proving efforts directly; the new base-level expectations then refocus my attention via their relationship to domain-independent rules (here, information about when one should try to prove a theorem and when one should attempt to prove its negation). Domain-independent control information that is capable of dealing with our new base-level expectations will duplicate behavior of this sort.[5]

In all of these cases, the advantages we have identified are an immediate consequence of the observation that reasoning systems should work with a domain *dependent* base level (including modal information) and domain *independent* control information.

## Acknowledgement

We would like to thank Alon Levy, Nils Nilsson and David Smith for many enlightening discussions. The comments of two anonymous reviewers are also gratefully acknowledged.

## References

[Dechter and Meiri,1989] Rina Dechter and Itay Meiri. Experimental evaluation of preprocessing techniques in constraint satisfaction problems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 271–277, 1989.

[Dechter and Pearl,1988] Rina Dechter and Judea Pearl. Network-based heuristics for constraint-satisfaction problems. *Artificial Intelligence*, 34:1–38, 1988.

[Dechter,1990] Rina Dechter. Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *Artificial Intelligence*, 1990. To appear.

[Elkan,1990] Charles Elkan. Incremental, approximate planning. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 145–150, 1990.

[Etzioni,1990] Oren Etzioni. Why PRODIGY/EBL works. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 916–922, 1990.

[Ginsberg and Harvey,1990] Matthew L. Ginsberg and Will D. Harvey. Iterative broadening. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 216–220, 1990.

[5] The blind search technique of iterative broadening [Ginsberg and Harvey,1990] appears to be based on a similar observation.

[Ginsberg *et al.*,1990] Matthew L. Ginsberg, Michael Frank, Michael P. Halpin, and Mark C. Torrance. Search lessons learned from crossword puzzles. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 210–215, 1990.

[Ginsberg,1991] Matthew L. Ginsberg. The computational value of nonmonotonic reasoning. In *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, Boston, MA, 1991.

[Lipman,1991] Barton L. Lipman. How to decide how to decide how to ...: Modeling limited rationality. *Econometrica*, 1991. To appear.

[Minton *et al.*,1989] Steve Minton, Jaime G. Carbonell, Craig A. Knoblock, Daniel R. Kuokka, Oren Etzioni, and Yolanda Gil. Explanation-based learning: A problem solving perspective. *Artificial Intelligence*, 40:63–118, 1989.

[Silver,1986] Bernard Silver. *Meta-Level Inference*. Elsevier, New York, 1986.

[Smith and Genesereth,1985] David E. Smith and Michael R. Genesereth. Ordering conjunctive queries. *Artificial Intelligence*, 26(2):171–215, 1985.

[Smith *et al.*,1986] David E. Smith, Michael R. Genesereth, and Matthew L. Ginsberg. Controlling recursive inference. *Artificial Intelligence*, 30:343–389, 1986.

[Smith,1985] David E. Smith. *Controlling Inference*. PhD thesis, Stanford University, August 1985.

[Smith,1986] David E. Smith. Controlling backward inference. Knowledge Systems Laboratory Report 86-68, Stanford University, June 1986.