Towards a content model of strategic explanation.*

Kristian J. Hammond
Department of Computer Science
The University of Chicago
1100 East 58th Street
Chicago, IL 60637

Colleen M. Seifert
Department of Psychology
University of Michigan
330 Packard Road
Ann Arbor, MI 48104

Abstract

Over the past few years there has been a growing interest in the notion of using causal explanations in both learning ([Dejong and Mooney, 1986] [Mitchell *et al.*, 1986]) and planning ([Hammond, 1989], [Hammond, 1987] and [Simmons and Davis, 1987]). The study of how complex causal explanations can be used in learning has turned into something of a cottage industry in AI; however, little attention has paid to how explanations may be constructed. In this paper, we will examine some of the current proposals concerning the process of explanation, augment them with a few ideas of our own, and suggest a new, more strategic level of knowledge about explanations that can be used to guide the explanation process. In particular, we are interested in the problems involved with integrating rule-based methods of explanation construction with memory-based approaches.

Explanation: Use vs. Construction

In recent years, explanation has been rediscovered as a theoretical issue in Cognitive Science. The stress, however, in the study of explanation has been aimed at the *use* of explanations, rather than their construction. Little work has been done on the issue of explanation as a constructive task. Our goal is to develop a model of the task that is a content theory of the *process* of explanation. In particular, we are interested in a theory that includes a strategic level of knowledge about explanations that can be used to guide the construction as well as the use of explanations.

The need for such a theory is straightforward: Explanations are being billed as the answer to the problem of credit assignment in learning ([Dejong and Mooney, 1986] and [Mitchell *et al.*, 1986]) as well as a credible approach to avoiding extensive search in planning ([Hammond, 1989], [Hammond, 1987] and [Simmons and Davis, 1987]). For these claims to be anything more than the exchange of one set of intractable problems for another, we need a theory that provides us with a method for constructing as well as for using causal explanations. As a first step towards this, we will examine the different proposals that have been made concerning explanation as a task and attempt to organize the operations required by each in into a single knowledge base. We will further propose a three-leveled organization of that knowledge base into **Task**, **Method** and **Domain** level knowledge.

Explanation in Learning: Backward Chaining

The most recent impact that causal explanation has had in Artificial Intelligence is as a mechanism for credit assignment in various theories of learning. Initially proposed as a method for learning macro operators in STRIPS ([Fikes *et al.*, 1972]) it was rediscovered by both DeJong and Mitchell and implemented in EGGs and EBG respectively. Although both researchers suggest an interesting use for explanations, neither has gone on to suggest an advancement over existing methods for the construction of the explanations that they use.

DeJong, because he has been concerned with domains involving planful action, has relied on a PAM-like ([Wilensky, 1978]) process of explanation that performs goal regression; that is, backward chaining from a perceived goal through the steps that are input to the system. The back chaining is done using a set of causal rules that include provide links between actions, states and intentions. As in PAM, an action is explained when it is linked to a plan, a plan is explained when it is linked to a goal and a goal is explained when it is either linked to an existing theme or has been provided to the system as part of its input.

Unfortunately, also like PAM, this process is nondeterministic and therefore potentially exponential. DeJong handles this problem by introducing an arbitrary length limit to the chains that the explanation

*This work was supported in part by the Defense Advanced Research Projects Agency, monitored by the Air Force Office of Scientific Research under contract F49620-88-C-0058, and the Office of Naval Research under contract N0014-85-K-010.

process is allowed to construct. As a result, any set of inputs that include gaps larger than this limit simply cannot be explained. This limit is, in some sense, essential to the overall argument that DeJong proposes for the use of explanations in learning. Without it, all inputs would fall into the class of "explainable" cases, and there would be no reason to build new scripts that are the result of the learning process. With it, the only inputs that can be explained are those which are either sufficiently detailed to allow the explanation process to link the steps, or those that are examples of previously constructed scripts.

EBL is not the only theory with this drawback; Mitchell's EBG, though concerned with more structural explanations and categories, also faces this problem. The fact that Mitchell has tended to use theorem provers as the basis for the explanation process tends to make this fact even more apparent because the techniques he uses are known to be worst-case exponential. While these techniques are known to be potentially exponential, their extensive use has minimized the problematic status of these solutions to the explanation problem.

Explanation Patterns: Reminding and Transformation

Outside of the realm of EBL, some work has been done in the area of explanation generation. Schank has suggested a memory intensive approach that uses structures called Explanations Patterns ([Schank, 1986]) that record existing explanations in a form that allows them to be reused analogically. Schank's approach combines a retrieval stage in which the features of an unexplained event are used to find an explanation pattern (or XP) in memory, followed by a transfer or "tweaking stage" in which the existing explanation is transformed to fit the current case. This approach differs from backward chaining because it involves search of a space of transformations rather than a space of operators. Given the nature of most domains (non-homogeneous with clusters of viable explanations), this approach is a potential winner in terms of efficiency of search.

It appears that the techniques that Schank suggests may be directly adaptable to the problem of building the causal *dependency structures* that the EBL methods require. A much clearer case for the reuse of packaged explanations in the construction of dependency structures has been provided by Simmons' GTD, an explanatory system that uses associational rules to generate approximate explanations and detailed causal rules to debug them ([Simmons and Davis, 1987]). While different in form, this approach depends on the same principle as Schank's XP's. That is, viable explanations for events in a domain tend to be found in clusters, making the approach of "approximate and debug" an effective one.

Both of these approaches rest on the overall coherence of most domains. They construct approximate solutions Schank through recall of existing patterns and Simmons through the use of associational rules that are then modified to fit the facts that require explanation. The interesting aspect of both is that they are alternatives to simple backward chaining, and provide a different means by which the space of possible explanations can be searched. Specifically, the space of explanations is searched via transformational rules that allow the explanation process to change the structure of an explanation without any backtracking.

Understanding as Explanation: Script Application

There is a third view of explanation that is, in some sense, the simpliest theory of how explanations are constructed. This is the view of *understanding* as an explanation process, in which new events are explained by virtue of their identification within a script. The result of this view is the notion that script application is an explanatory process in which attention is focused on relevant aspects of the situation by the script or frame ([Minsky, 1975], [Charniak, 1977], [Schank and Abelson, 1977], [Wilensky, 1978] and [DeJong, 1979]).

The basic process of script (or frame) application has two parts: script selection and script application. Script selection requires the use of *semantic indexing* of scripts in a knowledge base that is traversed in the initial phases of understanding. Once a script is found, it is applied (or instantiated) by having empty fields in the script filled in with information gleaned from a piece of text or set of conceptual structures.

Like the process of explanation transformation suggested by Schank and Simmons, this process rests of the existence of structures in memory that approximate the final form of the explanation. Like backward chaining, however, it is essentially non-deterministic in both selection and instantiation of the scripts, opening the door to the possibility of exponential search.

Rules, Memories and Scripts

Each of these three approaches to explanation makes use of a different knowledge base; specifically, domain level inference rules for backward chaining, explicit memories for reminding/transformation and scripts for selection/instantiation. But what is the relationship between these approaches? Is there a way to combine them into a single model of explanation?

An obvious approach would be to characterize scripts and episodic information as macro versions (combinations) of the inference rules. This approach is immediately problematic: The basic algorithm of transformational (memory-based) explanation is very different than that of backward chaining, in that the internal structure of the reminding is altered using domain neutral transformation rules ([Kass *et al.*, 1986]). Likewise, the process of script selection and instantiation makes use of control heuristics that have little to

do with the search control rules of backward chaining. Since the methods are incompatible, simply combining the rule sets is unworkable.

The opposite approach, that of characterizing backward chaining as a degenerate form of reminding plus transformation, seems to be equally problematic. There are dynamics and technology involved in backward chaining that cannot currently be reproduced within the confines of the reminding/transformation approach.

The only alternative, then, is to view these three approaches to explanation as exactly that, three separate and distinct methods for constructing explanations. This implies, however, that we must now think in terms of how they interact, and how we can integrate them into a single theory of explanation. In effect, it implies the need for a theory of how to control the movement between these approaches in constructing viable explanations. In some ways, this theory will be analogous to content theories of search control; however, the basic operations that make up the right-hand-sides of the control rules will differ in that they will refer to the components of reminding – transformation, selection and instantiation – as well as rule application. The intention here is to provide for explanation what Stefik did for planning ([Stefik, 1981a]); that is, suggest a content theory of the processes involved in the construction of explanations.

Step One: Representation

Our basic notion of explanation comes out of the literature on plan understanding ([Wilensky, 1978] and [Charniak, 1983]) and DeJong's work in explanation-based learning ([Dejong and Mooney, 1986]). In this work, an explanation is a set of dependencies in which states are supported by action/rule pairs, actions are supported by plan/rule pairs, plans are supported by goal/rule pairs and goals are supported by the theme/rule pairs. We will ignore for a moment the support structure of the rules themselves.

While, each support requires both a rule and a fact, either can be an assumption required to construct the explanation. This form is exactly what is produced through backward chaining, and is stored in memory for use in the construction of new explanations. Scripts take a slightly different form in that they organize multiple *part-of* relationships into a single structure that has a standard goal associated with it. The final form of the type of explanation associated with scripts, however, still consists of these dependency structures.

Although there is a single final form for each of these methods, each uses a different sort of knowledge base. Explanation via chaining requires a base of inference rules that allows the incremental addition of individual links to the final dependency chain. Explanation via reminding plus transformation requires a knowledge base of known explanations and a set of transformation rules that allows these explanations to be modified. And script based explanation requires a knowledge base of abstracted scripts that can be refined through the use of slot filling mechanisms. These different kinds of facts about a domain make up what we call the **Domain Level** of knowledge used in explanation.

Step Two: The Task

The task of explanation begins with the notion of an anomaly ([Schank, 1986]) and ends when that anomaly supported by a dependency structure that serves to link it to known or assumed facts. As we have said, this can be done through the use of any one of the three methods we have discussed. But each of these methods brings its own characteristic method and dynamic to the problem.

Backward chaining involves the use of domain rules to traverse a virtual AND/OR tree, using control rules to guide the search. Reminding/transformation requires the retrieval of explanations from memory and the subsequent transformation of those explanations. This is supported by rules that select the appropriate features from the initial problem for use in indexing, and rules that control the application of the transformation rules themselves. Script application also depends on the use of indexing techniques to find the appropriate script in a library of possibilities.

Step Three: The Actions

In this section, the central issue of the set of actions that can be applied during the construction of a explanation is addressed. What we examine here are the "right hand sides" of the control rules that guide the construction of explanations. Some of these actions, such as rule or script selection, are already part of the standard repertoire of understanding and explanation. Others, such the feature selection and *recasting of representation*, are more recent additions to this list. For presentation purposes, the more familiar of these actions are considered first, followed by more recent additions. However, the unifying notion is that these actions may be taken at any time during the construction of an explanation. While some are logically dependent on others (e.g., you have to select a set of features to use in indexing before looking for an explanation in memory), the idea is that these actions are on the right hand side of rules that theoretically can be fired at any time during the explanation process. In particular, we want to stress that the selection of an explanation method (chaining, script application or reminding/transformation) is not fixed: An explainer can (and almost always will) move between the different methods.

There really are only two actions (aside from backing up) that can be taken in constructing an explanation out of a base of domain rules; namely, rule selection and rule application. While they are not the most important rules in the library we are constructing, search

control rules are the most familiar. These are rules that guide the selection of the individual domain rules that are applied to explain an anomaly when a disjunct of possibilities is presented. For example, in trying to explain why Jack is out of orange juice, we could have attempted to find an explanation that involved some use of orange juice other than as an ingestible liquid. But this line of reasoning is cut short by control rules that give preference to the explanation that includes the "standard" use. But chaining is only a small part of the overall explanation process and is really only attempted after many other steps are taken. In fact, there are many steps involved with constructing an explanation that come well before the selection of a backward chaining rule or even the decision to apply backward chaining as a method in the construction of the explanation.

Before building an explanation, there must be something to explain; therefore, the first step is the selection of the anomaly or set of anomalies to explain. What is needed at this stage is a set of rules for deciding which anomalies are even candidates for inclusion in a single explanation. These rules are needed to initially focus the explainer's attention on the events and states that it should attempt to connect. In an earlier paper ([Hammond and Hurwitz, 1988]), we suggested a set of heuristics based on temporal, physical and systemic proximity that guided the attempts at joining separate anomalies into a single coherent explanation. These rules are particularly important when the explainer does not have a complete set of domain rules, and thus must make assumptions about the connectivity (or lack of connectivity) of a set of input features.

These rules allow a system to propose causal relationships between anomalous features that were proximate from one of many possible points of view. Schank has also proposed this type of heuristic ([Schank, 1986]); in particular, attempting to coordinate anomalies that occur at the same time as well as those that are of the same type. Likewise, Falkenhainer ([Falkenhainer, 1988]) has proposed a similar heuristic that looks for particular differences in the input (as compared to a standard model) to try to connect.

The important point here is that these heuristics focus the explainer's attention of a select sub-set of features that will be included in the explanation itself. Of course, these are heuristics, and, as such, are open to error. But it is important to understand that, like the search control rules of backward chaining, they are essential to controlling the movement through the space of possible explanations that can be constructed for any one set of inputs.

Once a set of anomalies are selected, the next step is to decide on the basic strategy that should be taken in trying to join them into a single explanation. This is just the decision as to whether to try to find a single cause for all of the anomalies, to explain them in terms of a flow of cause and effect or to find an explanation that includes multiple actors and goals that tie the anomalies together. For example, in trying to explain the correlation between a rise in ice cream sales and a rise in the occurrence of drownings, one important step is to see the two facts as the dual effects of a single cause; namely, the onset of summer. An attempt to explain these events by creating a causal chain leading from one to the other ("eating ice cream gives you stomach cramps when you try to swim" or "people who watch drowning victims being taken away tend to eat a lot of ice cream") leads the explainer away from a more reasonable explanation.

Examples of useful heuristics include attempting to build chains forward from earlier actions, building out from intentional actions to states, and attempting to construct the typical **theme->goal->plan->action** chains that are often used to explain human behavior ([Schank and Abelson, 1977]). Like the rules determining the selection of the anomalies, these heuristics are a large part of what controls the search for the explanation.

Next, somewhat dependent on the selected strategy, comes the selection of actual method to use in the construction of the explanation (or part of an explanation). This involves deciding on which of chaining, script application or reminding/transformation to apply to the problem. As we mentioned earlier, this decision is not a fixed one that will remain unchanged throughout the explanation process. Existing scripts or explanations often have to be extended using individual causal rules. Likewise, a partial explanation constructed using backward chaining will uncover a new set of features that may be used to find a script or existing explanation.

This flexibility in method application is an important point in this model: We are proposing an integrated approach, not advocating one method over others. It is clear that each of these methods has its advantages, and that each has its problems. Script and memory-based methods provide a good means for searching very regular spaces of explanations, and chaining gives us the ability to do a formal analysis of the search. However the complexity of script and memory-based approaches when applied to less regular domains has yet to be solved, while simple chaining methods do not allow for the possibility of reuse of already formed explanations. So, it is clear that the best approach is one that attempts to integrate these into a single methodology for constructing and saving explanations. It may in fact be the case that, as a model of human cognition, a variety of such approaches are necessary to replicate the flexibility of the explanation process. Depending upon how these separate processes are defined, there may be functional justifications for their existence within an arsenal of methods to be applied to problems. Finally, some psychological evidence suggests that problem solving ability, and creativity, may be linked to the individual's ability to flexibly move

between approaches to a problem.

Along with explanation retrieval and transformation, there is another sort of action that needs to be taken by memory based explanation systems that is often ignored: This is the *recasting* of initial representations into forms that might provide better feature sets for use in retrieval of existing explanations. Intuitively, this is what we do when we "see something from a different point of view" or "in a different light". In order for this to make sense, however, it must be driven by a set of heuristics that makes these changes for a reason.

Kass, Leake and Owens have done some tentative work in this area in the SWALE project ([Kass *et al.*, 1986]). In this work, they looked at the issue of transforming slots fillers of partially filled structures in order to generate more indices for search into memory. For example, viewing a race horse as an athlete in order to find explanations related to athletes that might be applied to horses. Most of this work centered on the notion of moving around in a semantic net however and was not concerned with either the notion of generating characterizations of an overall situation or in using aspects of the macro structure of the representation to generate new features. An example of the first would include representational changes such as seeing a take-over of space in an office complex as imperialism ([Schank, 1986]) or seeing the decision to go to sleep or keep working as a resource allocation problem. The point in both of these is that there is a representation that captures aspects of the situation that are not captured by a simple listing of the actions involved.

Recasting, as we are defining it, involves looking at event and goal configurations at a more abstract level. For example, in the following story:

> Bob was sitting in a hallway when a woman came out and took a drink from the near by water fountain. A few moments after she left, she returned for another drink. A few moments after this, she came back for a third drink. As she was drinking, a man came up behind her and held a knife above her, poised to strike. She turned, screamed, and then the two of them laughed and walked away together.

As with the single bug assumption, a strategy selection decision is made to try to find a single goal that both drinking water and being attacked might satisfy. How to find such an explanation? The flat representation of individual actions can be recast as the *repetition* of a single event – leading to a memory search that attempts to explain why events are repeated – followed by the threat of attack and laughter. This leads to an attempt to explain the story in terms of the possible reasons for multiple versions of the same action.

In protocol studies [Seifert, 1989] subjects seem to perform this particular recasting of representation whether or not they go on to find an explanation for the story. In particular, subjects often propose the notion of rehearsal to explain the repeated events, either as part of a play or movie, or for a real attack. In either case, the structural form of the actions is used to recognize a feature which can then be used to either find an existing explanation or help in the construction of a new one. The point is that there is a basic piece of vocabulary being used to store information in memory (and as part of backward chaining rules) that must be derived from even the deepest representations of events.

A second piece of recasting appears to be critical in the development of indices that will successfully retrieve a particularly apt explanation from memory. In representing the attack, the outcome (shared laughter) is not the expected one, and that it therefore did not achieve the desired result. Therefore, the notion of a different intention or action may be considered through a variety of strategies: the outcome that did occur can be considered the intended one, and the actions recast to fit; or the outcome can be considered a side effect of the actions rather than the intended outcome. In both cases, if the attempted attack is recast in terms of its actual outcome, the scream is viewed as a successful outcome of an attempt to scare the woman. The addition to the representation through recasting – that the attack "scared" the woman – facilitates the connection of the repeated actions with the attack. Attempting to incorporate "scaring someone" and "someone drinking water" as plans in service of the same goal are much better indices to retrieve from memory the goal of curing the hiccups, an explanation satisfying all of the information in the problem.

This process of recasting representation, although presented here as an aspect of reminding/transformation, actually belongs at the level of anomaly, strategy and method selection. This is because the recasting of representation has a global effect on the course of the explanation in much the same way that it does in problem solving. That is, it allows the introduction of new features that themselves present alternative paths for chaining and/or script and explanation retrieval. These four basic actions make up what we call the **Task Level** knowledge in explanation. That is, a level of knowledge about how to proceed in building an explanation in general. The other actions are part of what we call the **Method Level** in that they are the subparts of specific approaches. These division play the same role in explanation as the divisions suggested by both Stefik ([Stefik, 1981b]) and Hayes-Roth&Hayes-Roth ([Hayes-Roth and Hayes-Roth, 1979]) play in planning. They allow different types of knowledge to have access to the process control during the entire process, rather than just at a single stage.

The final set of actions in the method level are those associated with script application. The two most relevant are script selection and slot instantiation. We hold with the idea that script selection is an issue

open to guidance by both semantics and pragmatics ([Schank and Birnbaum, 1980]) and that slot instantiation is best guided by the semantic constraints provided by the script itself ([DeJong, 1979]). A full treatment of either of these issues is well beyond the scope of this paper, but it is important to note that these are the two most relevant actions in script application, even though each is further decomposable into subactions.

The actions involved with explanation construction fall into two main groups: **Task Level** action and **Method Level** actions. Task level actions relate to global issues of representation and strategy. Method level actions are the actions used in the control of the actual building of the explanations within a particular method being used. An important point is that method level actions tend to be used only within the confines of particular methods while task level actions are always available to guide the construction of the explanation. Knowledge of rules, individual scripts and memories of existing explanations is stored on its own **Domain Level**.

Conclusions and the next step

Our goal in this paper was to outline the process of explanation as a task. We have tried to accomplish this in three basic steps. First, we have posited a general representation for explanations that can be (and in many cases is) used for work in both EBL and plan debugging. Second, we have tried to at least partially explicate the different process models that have been proposed. And third, we have suggested an integration of those proposals through the decomposition of the primitive actions that fit into a three level organization of **Task**, **Method** and **Domain** level knowledge. This decomposition is the first step in a true integration of the different methods into a single theory of explanation construction. We suggest that the final form of this integration will be similar to the distributed systems suggested by Stefik, and Hayes-Roth and Hayes-Roth, for use in planning.

References

E. Charniak. Ms. malaprop: A language comprehension program. In *The Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, Cambridge, Mass., August 1977. IJCAI.

E. Charniak. Passing markers: A theory of contextual influence in language comprehension. *Cognitive Science*, 7:171–190, 1983.

G. Dejong and R. Mooney. Explanation-based learning: An alternative view. *Machine Learning*, 1(2):145–176, 1986.

G. DeJong. *Skimming Stories in Real Time: An Experiment in Integrated Understanding.* PhD thesis, Yale University, May 1979.

B. Falkenhainer. The utility of difference-based reasoning. In *The Proceedings of the Seventh Annual Conference on Artificial Intelligence*, St. Paul, Minnesota, August 1988. AAAI.

R.E. Fikes, P.E. Hart, and N.J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288, 1972.

K. Hammond. Learning and re-using explanations. In *Proceedings of the Fourth International Conference on Machine Learning*, Irvine, CA, June 1987. ML.

K. Hammond. *Case-Based Planning: Viewing Planning as a Memory Task.* Academic Press, 1989.

B. Hayes-Roth and F. Hayes-Roth. A cognitive model of planning. *Cognitive Science*, 3(4), 1979.

A. Kass, D. Leake, and C. Owens. Programming the theory: Swale, a program that explains. In R. Schank, editor, *Explanation Patterns: Understanding Mechanically and Creatively.* Erlbaum, 1986.

M. Minsky. A framework for representing knowledge. In P. Winston, editor, *The Psychology of Computer Vision*, chapter 6, pages 211–277. McGraw-Hill, New York, 1975.

T.M. Mitchell, R.M. Keller, and S.T. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1):47–80, 1986.

R. Schank and R. Abelson. *Scripts, Plans, Goals and Understanding.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

M. Schank, R. Lebowitz and L. Birnbaum. An integrated understander. *American Journal of Computational Linguistics*, 6(1):13–30, 1980.

R. Schank. *Explanation Patterns: Understanding Mechanically and Creatively.* Lawrence Erlbaum Associates, 1986.

C. M. Seifert. Inference in problem solving. *Unpublished manuscript*, 1989.

R.F. Simmons and R. Davis. Generate, test, and debug: combining associational rules and causal models. In *The Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, 1987. Morgan Kaufman.

M.J. Stefik. Planning and meta-planning. *Artificial Intelligence*, 16:141–169, 1981.

M.J. Stefik. Planning with constraints. *Artificial Intelligence*, 16:141–169, 1981.

R. Wilensky. *Understanding Goal-Based Stories.* PhD thesis, Yale University, 1978. Research Report #140.