Plan Recognition and Revision in Support of Guideline-Based Care

Yuval Shahar and Mark A. Musen

Section on Medical Informatics, Departments of Medicine and Computer Science,
Stanford University, Stanford, CA 94305
shahar@camis.stanford.edu

Abstract

We consider the problem of providing automated support for guideline-based clinical care. Clinical guidelines are a common format in medical domains for prescribing a set of rules and policies that an attending physician should follow. In terms of an AI planning task, clinical guidelines can be viewed as a shared library of highly reusable skeletal reactive plans, whose details need to be refined by the executing planner over significant periods of time. The application of clinical guidelines involves the collection and interpretation of patient-related data, the application of prespecified plans, and a revision of the plans when necessary. Over the past decade, several research groups have implemented reactive-planning architectures specific for the task of refining skeletal plans over time. The importance of such systems is increasing as more clinical data are being captured and represented in an electronic format, and as quality control of medical care grows in importance. Conducting a flexible, intelligent dialogue with the physician user of these systems requires an ability to reason about the user's *goals* and *plans* and about possible *modifications* to these plans. We point out that automated support for guideline-based care can be viewed as a collaborative effort of two planning agents: the physician and an automated ("assistant") planner, whose knowledge might be limited, or who might not have access to all the data. We demonstrate that achieving even a modicum of collaboration between the two planners and of highly desirable flexibility in the automated support to the physician involves a form of reasoning about mental states: a recognition of each planner's (in particular, the physician's) *intentions* and *plans* to achieve them, and a consideration of the available *plan-revision strategies* during execution time. In particular, automated support for clinical guidelines could be enhanced considerably by a sharable, explicit, formal representation of (1) *therapy-planning–operators' effects*, (2) *plan-revision strategies*, and (3) the underlying *goals and policies* of the guideline, in the form of *temporal-abstraction patterns to be maintained, achieved, or avoided.* Finally, relying on our analysis, we can list in a structured manner several possible sources of disagreement between the two planners, thus supporting the maintenance of the automated planner's knowledge base as well as improving the appropriateness, and thus acceptability (by the physician) of its future recommendations.

1. Clinical Guidelines and Protocol-Based Care

It has become increasingly clear to the medical community that common standards of care need to be specified. Two formats for representing such standards are clinical guidelines and clinical protocols. **Clinical guidelines** are a set of general rules and policies for treatment of patients who have a particular clinical condition. **Clinical protocols** are a more detailed version of clinical guidelines, used when the guidelines (possibly experimental) need to be applied in as similar a fashion as possible, so as to enable statistical analysis of outcomes for comparison among a set of guidelines.

During the past 15 years, there have been several efforts to create automated reactive planners to support the process of protocol-based care over significant periods of time, such as ONCOCIN [Tu et al., 1989] in the oncology domain and T-HELPER [Musen et al., 1993] in the AIDS domain. The task-specific architecture underlying both of these systems has been abstracted into the **episodic skeletal-plan refinement (ESPR)** problem-solving method [Tu et al., 1992]. The ESPR method decomposes the task of guideline-based therapy into three subtasks: *instantiation* and *decomposition* of a specific plan, *identification* of problems, and *revision* of the current plan.

A different approach to automated support of guideline-based therapy is the **critiquing method**: the program critiques the physician's plan rather than recommending a complete one of its own, thus focusing on the user's needs and exploiting the assumption that the user has considerable domain-specific knowledge [Miller, 1986]. A task-specific architecture implementing the critiquing process has been generalized in the HyperCritic system [van der Lei and Musen, 1991]. Task-specific architectures assign well-defined *roles* to domain knowledge and thus facilitate its acquisition and maintenance.

The application of clinical guidelines and protocols involves collecting and interpreting considerable amounts of data over time, applying standard treatment plans, and

revising those plans when necessary. The guidelines often involve implicit assumptions about the knowledge of the planner executing the plans, both in the data-interpretation and in the treatment-planning phases. Thus, clinical guidelines can be viewed as a *shared library* of *highly reusable skeletal reactive plans*, whose details need to be refined and executed by a *reactive planner* over *significant periods of time.*

Automated support for the application of a clinical guideline involves a dialogue between two planning agents—the human and the automated one (who can be considered as an "assistant planner"). Each agent has relative advantages. For instance, the physician might have access to additional data, unavailable through the narrow channel of the electronic medical record, or to a more comprehensive knowledge base; the automated planner might more easily detect patterns in the data over long periods of time, and has direct, fast, and accurate access to a protocol's complex set of prescribed rules.

In this paper we argue that each planning agent (in particular, the automated one) should be trying to understand at least a small part of the reasoning processes of the other, to optimize the results of their collaboration. This understanding involves reasoning about a knowledge base of clinical guidelines (in principle, common to both planners) that should be annotated explicitly with the intentions of each guideline's designer. One such annotation that we discuss includes, for each segment of the guideline, lists of desirable and undesirable temporal patterns of concepts that can be abstracted from the patient's data. In addition, the automated planner needs to have access to a set of plan-revision strategies and to the domain-specific effects of planning operators. Thus, the physician's plans can be recognized, leading to a more flexible dialogue and to a better acceptance of automated systems for support of guideline-based care.

1.1 The Paper's Structure

We start by explaining, in Section 2, some of the special problems and requirements involved in the application of clinical guidelines and protocols. We then demonstrate (in Section 3) the need for plan recognition by an automated guideline-based planner, and (in Section 4) the need for reasoning about plan revision. In Section 5 we emphasize the special importance of temporal reasoning in representing and of reasoning about goals and policies in time oriented domains. Time-oriented domains are the subject matter of practically any clinical guideline that needs to be instantiated episodically for the same group of patients. Using the language introduced in Section 5, we present in Section 6 several generic plan-revision strategies that need to be considered when attempting to recognize a physician's modification to the skeletal guideline or protocol. In Section 7 we discuss briefly the implications of this reasoning framework for a structured listing of further sources of discrepancies between the physician and the automated planner; noting such discrepancies systematically supports both the automated planner's operation and the maintenance of its knowledge base. We summarize our views in Section 8.

2. Problems in Application of Clinical Guidelines

Several problems, common to clinical protocols, arise due to the skeletal nature of clinical guidelines and protocols and the fact that guidelines require additional task-specific and domain-specific knowledge for instantiation in any particular case.

- Protocols are often *ambiguous* or *incomplete* [Musen et al., 1987]. The protocol might suggest a particular drug, but leave the precise dose to the attending physician. The protocol might specify what to do when a renal toxicity due to the drug is detected, or when suppression of the bone marrow occurs, but not what the physician should do when both occur at the same time. Furthermore, physicians often do not adhere to written protocols, believing their own therapy to be closer to the "intentions" of the protocol designers [Hickam et al., 1985]. Thus, additional clinical knowledge is implied and is necessary for a refinement of the protocol.

- Certain *violations* of the dictated protocol necessitate withdrawing the patient from a controlled study. Such extreme measures often seem arbitrary, particularly in view of the inherent ambiguity of many protocols, and might not be part of the designer's intentions. A better representation of a guideline plan might include several *levels* of violation, so that the patient is still considered to be treated by a protocol until a certain violation level has been identified. Such flexibility would enable the automated planner to continue offering useful advice even if the default plan is not adhered to completely (e.g., there are alternatives to prescribed therapies which are part of the implied domain knowledge).

- As mentioned above, a physician might ignore the automated planner's recommendation. We might want her to explain why she has done so in a structured manner, thus facilitating the continuation of collaboration and the maintenance of the automated planner's knowledge base. Therefore, a predefined set of categories should be available to capture the physician's justification for making different decisions (e.g., "there is a particular type of knowledge missing from the knowledge base"). It would also help to be able to represent explicitly whether a change in therapy is due to the program's recommendation, the physician's decision, or the patient's behavior.

- Since modifications to the protocol are common, an automated planner should adjust its future recommendations to reflect changes introduced by the physician. The automated planner should recognize when certain goals are being achieved by the physician and not continue (in further sessions regarding the same patient) to suggest measures to achieve these goals. Similarly, the

automated planner should recognize cases in which overall policies are still (possibly indirectly) being adhered to by the physician's plan, and thus should adjust accordingly both its critique of the physician's plan and its own plans.

3. The Need for Plan Recognition in Support of Guideline-Based Care

Clinical guidelines and protocols involve an explicit or an implicit set of *policies*, or *intentions*, of the designers of the guideline. For instance, a general policy might be intended, to the effect that hemoglobin (Hb) values should be, in a certain context (e.g., an AIDS-treatment protocol), kept above 7 gm/dl, and that an episode of severe anemia should not last for more than 2 weeks. Representing such intentions explicitly requires *annotation* of the plans and actions prescribed by the protocol. In Section 5, we present one annotation formalism for representing intentions over time.

Even if intentions are represented by proper annotations to the protocol, they do not suffice for recognition of the physician's plan by the automated planner. The assistant planner also needs a library of *actions*, or, in general, of *plans* corresponding to particular goals, subgoals and expected problems that the planner might encounter while executing a plan (the expected problems are called **plan bugs** in the **case-based planning** literature [Hammond, 1989]). Combining the intentions with the plan library enables the automated planner to realize that the physician is following the same higher-level policy that it attempts to follow, even when the physician has apparently overridden the system's recommendation by executing a different set of actions. Thus, the ability to represent goals and higher-level policies, combined with knowledge about the semantics of therapy-planning actions, would support a limited form of **plan recognition**. The program would be reasoning about the mental states of the physician defining the clinical guideline, of the physician executing that guideline, and, in a sense, of the automated planner itself.

For example, assume that a severe anemia state (that is, very low values of the Hb parameter) is detected for the second consecutive week by the automated planner, and that the physician records in the patient's electronic record that she intends to give the patient a transfusion of blood. This action seems to contradict the automated planner's suggestion (following the protocol) of attenuating the dose of a drug (that the patient is being given) that is toxic to the bone marrow (which generates blood elements, including red blood corpuscles that contain Hb). However, given sufficient knowledge regarding effects of planning operators such as drug administration, the automated planner can note that the transfusion increases the value of Hb through an external event, while the planner's recommendation increases the value of the same parameter (Hb) by reducing the magnitude of an event (the dose of the toxic drug) that affects the value of Hb negatively. The physician's plan, although different from the program's initial recommendation, might still be following a higher directive of the guideline in question, namely, avoiding a severe anemia period of more than 2 weeks duration. By recognizing this high-level goal, common to both the automated planner and the physician, and its achievement by a different strategy, the automated planner would accept the physician's suggestion; would realize that she is still following the protocol's policy (i.e., that the patient should still be considered as being treated by the same protocol); would record an appropriate justification for the physician's new plan (i.e., the intention, goal, or policy); and would abort, for that therapy session (and possibly for future ones of the same patient), that part of the automated planner's recommended plan that attempts to achieve the particular recognized goal in question (i.e., avoid a low level of Hb), unless the physician's plan fails

Another example necessitating recognition of the physician's plans is a common situation occurring in management of insulin-dependent diabetes patients. In the context of a finding of high blood-glucose values during bedtimes, the physician might prescribe a decrease in the food intake of a diabetic patient during supper time (thus indirectly decreasing the after-dinner blood glucose value). Such a plan might contradict the automated planner's suggestion of increasing the dose of the patient's presupper insulin (which has the direct effect of lowering the patient's blood glucose, which is the usual effect of insulin). In both cases, however, the higher-level goal of keeping the value of the blood glucose within certain ranges is followed, since both actions lead to reduction of the blood-glucose value.

Thus, the automated planner needs to recognize in which parts of the skeletal plan the physician is in fact following higher-level goals in an acceptable way and in which parts she is indeed deviating significantly from the prescribed guideline, and what are the *justifications* for these deviations. Given sufficient knowledge (i.e., the guideline's intentions and the effects of various actions and plans), the automated planner will still be able to intelligently monitor and support other, independent therapy-planning decisions of the physician. Such an ability would increase the usefulness and acceptance of guideline-based decision-support systems to clinical practitioners, who tend to adhere to the protocol's intentions rather than to particular prescribed actions.

4. The Need for Reasoning About Plan Revision in Guideline-Based Care

Library plans, such as guidelines and protocols are composed of, are often only defaults or initial values, which need to modified when expected or unexpected complications occur. For instance, as mentioned in Section 1, the ESPR method includes plan-revision as one of three major subtasks into which it decomposes the task of protocol-based care.

Three major situations necessitate explicit reasoning, by the automated planner, about plan revision: The physician might override the automated planner's suggestion; the automated planner might detect an (expected) complication

of its major plan; or there might be an inherent ambiguity as to how to further refine the guideline or protocol.

4.1 Overriding of the Automated Planner's Plan by the Physician

The physician may override the program's suggestion, thereby modifying the standard recommended plan. The program now needs to recognize that there was indeed a modification of its recommended plan and to understand precisely what the modification was, what the justification for that modification was, and, in particular, what the patient's situation is now, with respect to the protocol, so that decision support can continue in a correct context (e.g., is advice regarding the dose of radiotherapy still relevant?) In doing so, the program will avoid making erroneous interpretations and recommendations in the patient's next visit (i.e., in the next episode of skeletal planning for the overall plan). The program will know that the context has changed, that some of its recommended actions were aborted, and that some of the guideline's goals are being achieved (perhaps using other methods) by the human planner.

4.2 Detection of an Expected Complication

The standard-protocol execution may encounter potential (expected) difficulties (detected by a problem identification method) manifested as a deviation from some predefined intentions.

For example, a complication such as "a toxic episode due to AZT" is identified. One or more plan revisions can be suggested by the program, such as attenuation of the dose of the offending drug or its substitution by another drug that achieves the same common goal and adheres to the same intentions in the guideline. Often, several revision strategies might be applicable to the same situation. If a set of structured revision options is offered by the assistant planner to the physician in a disciplined fashion, the best specific revision strategy can be selected by the physician, and its prescribed actions can be subsequently further refined and monitored more intelligently by the automated planner.

4.3 Recognition of an Impasse

The protocol may reach an **impasse**—a situation where either no action is clearly specified in the protocol, or where more than one action is possible based on different knowledge sources. For instance, there might be no strategy for a bleeding event, or no clear guideline for a conjunction of several toxicities; the different single-toxicity strategies might even be contradictory.

In all three plan-revision cases, the current plan is or has to be modified using domain-independent as well as domain- and task-specific revision strategies. In Section 6, we discuss several general plan-revision strategies (and the lower-level mechanisms that these strategies use) that can be used during execution time. Knowledge of such strategies not only enhances the performance of the automated planner, but also enables better recognition of the physician's plans, many of which are, in fact, modifications of the overall skeletal plan (i.e., the guideline).

5. Temporal Abstraction as a Tool for Plan Representation and Revision

It is impossible to discuss intelligently an episodic, time-oriented method for refinement of a skeletal treatment plan without a proper language for representing time-oriented concepts. Such a temporally oriented language is also necessary for annotation of a clinical guideline's intentions, since the very nature of clinical guidelines and protocols is to recommend actions and follow their results over time. Finally, reasoning about possible revision strategies during execution of a plan over any significant time period requires abstraction of time-stamped data into more manageable, higher-level concepts. Thus, before discussing the representation of clinical-guideline plans, intentions, and potential revisions, we need to examine briefly what the requirements for such a time-oriented language are, and what tools we have for performing the task implied by such representations, that is, the abstraction of data over time. This brief presentation will facilitate the rest of the discussion.

A major subtask of the guideline-based–therapy task, appearing explicitly in its decomposition by the ESPR method and, explicitly or implicitly, in any other approach for solving that task, is identification of the patient's state during execution of the guideline or protocol. This interpretation-type task is a general one, appearing in other contexts as well. In previous work, we have defined a formal framework for representing and using knowledge about abstraction of higher-level concepts from input, time-stamped data: the *knowledge-based temporal-abstraction theory* [Shahar, 1994]. We defined an *ontology* for the temporal-abstraction *task* and for the knowledge-based temporal-abstraction *method* that solves that task. That is, we defined a domain-independent, but task-specific theory of the domain-specific entities, properties, and relations relevant for the temporal-abstraction task in each domain. This ontology is used by the knowledge-based temporal-abstraction method and by the several temporal-abstraction *mechanisms* solving the subtasks posed by that method [Shahar and Musen, 1993; Shahar et al., 1992].

In the temporal-abstraction ontology, input entities include external *event propositions* (e.g., administration of medications) interpreted over intervals (i.e., *event intervals*), and measured *parameter* values (e.g., Hb values), likewise interpreted over time intervals (possibly zero-length intervals, in the case of time points). Events can be of several *types*; each type can have a list of *arguments* that can be instantiated with values (e.g., dose). Event arguments allow assignment of *magnitude* to an event proposition. Output (and sometimes, input) entities include also *abstractions* of parameters, which may be of type *state*, *gradient*, *rate* or the more general *pattern*. An abstraction of a parameter is a parameter (e.g., the state of Hb). Abstractions must be defined within an *interpretation*

context. Parameter propositions include a parameter (e.g., Hb), an abstraction type (e.g., `state`) a value (e.g., `grade_II_toxicity`) and an interpretation context (e.g., being treated by the CCTG-522 protocol). Parameter propositions are interpreted over some time interval (an ordered pair of time stamps, *start* and *end*; e.g., the interval [1/5/1991:8:00am, 2/3/1991:9:00am]), thus forming a *parameter interval.* Interpretation contexts are *induced* by external events, by certain parameter propositions, by the abstraction process's goals and intentions, and by certain combinations of these entities, but are not necessarily contemporaneous to the inducing entities [Shahar and Musen, 1993; Shahar, 1994]. Note that parameters are not under the automated planner's control directly, and are only modifiable through external events that have an affect on them.

We can now look into some of the implications, relevant to our discussion, of using the knowledge-based temporal-abstraction framework.

One of the benefits suggested by the knowledge-based temporal-abstraction framework is the ability to represent *goals* and *policies* of a time-oriented guideline plan as *temporal-abstraction patterns* that annotate nodes in a tree of goals and subgoals that represents a skeletal plan (the guideline). Annotations can include (1) temporal patterns that should be *maintained*, (2) temporal patterns that should be *achieved*, and (3) temporal patterns that should be *avoided* (either when detected or when generated hypothetically by a planner considering therapy options). Most patterns would include *parameter intervals*, but some might incorporate *event intervals.*

In addition, the ability to recognize temporal patterns of *events* can assist an automated planner in realizing that the physician is employing a particular plan. For example, in the context of treating patients who have insulin-dependent diabetes, it is important to recognize a common therapy regimen comprising one injection of a long-acting insulin in the morning and three injections of a short-acting insulin during the day. This realization would induce an appropriate interpretation context for blood-glucose values and for potential plan revisions.

Thus, a plan involves a set of therapy events to be administered over time, a set of clinical goals (e.g., certain patient states) to be achieved, and a set of policies inherent to the clinical guideline. The goals and policies can be represented as temporal patterns to be maintained, achieved, or avoided. Temporal patterns to be maintained or achieved include "keep the diabetes patient's morning blood-glucose values in the range 70-110 gm/dl." Temporal patterns to be avoided appear in assertions of the type "do not allow more than 2 weeks of grade II Hb toxicity in the context of the CCTG-522 AIDS-treatment protocol" or "avoid contraindicated drug combinations that are administered contemporaneously."

A typical case of representing a policy as an *avoidance* of a temporal pattern is an *event restriction* pattern: For instance, "Do not administer the drug Phenobarbital for the first 3 months following administration of the drug Phenytoin." Event restrictions are temporal-abstraction patterns indexed by the *event type* in question (e.g. off an event of type `Phenobarbital`) or by a more specific *event proposition* (with certain argument values) and can be represented as a *temporal-abstraction–pattern query* (possibly including both parameters and events) that needs to be checked by the automated planner whenever such an event is being instantiated or is proposed to be instantiated by the automated planner or by the physician.

An additional use of the temporal-abstraction language is for representing plan-revision mechanisms and the strategies that employ them.

6. Plan-Revision Mechanisms and Generic Plan-Revision Strategies

Skeletal plans usually need to be modified during execution. Thus, it is useful to examine what types of plan-revision strategies might exist: The automated guideline-based–care assistant needs to consider them whether reasoning about a modification to an executed plan or attempting to recognize the physician's plan. We also need to examine what domain-independent and domain-specific plan-revision knowledge is used by the actual mechanisms employed by the various revision strategies.

Execution-time revision mechanisms and strategies can be represented at four levels:

- *domain-specific* (oncology)
- *task-specific* (protocol-based care)
- *plan-specific* (protocol CCTG-522)
- *action-specific* (using the drug AZT).

The four levels are not a strict hierarchy and can also be described as four orthogonal axes. The same task can occur, for instance, in several domains, while several tasks may be instantiated in the same domain. Both intentions and revision strategies can be associated with any of the four axes.

For example, the event restrictions mentioned in Section 5 can be represented as annotations to a specific protocol, or to class of protocols, or to a whole clinical area (e.g., treatment of hypertension). Avoidance of contraindicated combinations of specific drugs when treating hypertension is a particular case of an event restriction at a domain and action level. Event restrictions are of the form

<event expression> RESTRICTED BY *<negative-patterns list>*

where the event expression is an event type or a fully instantiated event proposition, and the negative patterns include events and parameters.

Handling temporal patterns of *parameters* that are to be avoided can be exemplified by a plan-specific revision strategy such as "in protocol CCTG-522, if a Hb-state abstraction of grade II toxicity exists for 2 weeks, then suspend therapy with the toxic drug until the Hb state can be abstracted as grade I toxicity or less, in which case therapy can be continued."

Such revision strategies are indexed by the *temporal abstractions* created from the time-stamped data in the patient's electronic medical record. The revision strategies are activated when the pattern is recognized, and instantiate a plan, that is, one or more actions. The resulting new plan is carried out concurrently with the rest of the patient's management and is justified (in the logical sense) by the pattern evoking it.

6.1 Revision Strategies and Mechanisms

Revision strategies are driven by the temporal abstractions of the data; the temporal-abstractions language (augmented by additional special attributes, such as patient utilities) can be used to represent various activation conditions. A revision strategy is thus typically of the form

IF <*abstraction pattern*> THEN <*revision plan*>.

Revision strategies employ several basic **revision mechanisms**. The *revision plan* utilizes several specific revision mechanisms, such as replacing one event by another.

A common revision strategy uses the following revision mechanism in response to a conflict involving an event interval:

SUSPEND <*event proposition*> UNTIL <*condition*>.

This mechanism denotes, in fact, a series of lower-level revision mechanisms and actions:

CLIP(*event proposition*) (i.e., set the *end* time stamp of the interval over which *event proposition* is interpreted to the current time stamp);

Put on the planner's task agenda a revision mechanism of the type

IF <*condition*> THEN **START**(*event proposition*) (START initializes an event);

Update the actions' and plans' logical justifications.

Generic revision strategies are domain-independent strategies for revising plans. Although general in nature, they can be associated with specific events (a particular course of therapy), event types (a class of protocols), specific parameter propositions (the moderate-anemia value of the state abstraction of the Hb parameter), or domains (treatment of insulin-dependent diabetes). Thus, they are often indexed by *interpretation contexts*. Similar to case-based planning [Hammond, 1989], possible "bugs" (complications) can be indexed by the skeletal plan, with one or more suggested revision strategies.

The execution-time revision strategies are inspired by Simmons's **general debugging strategies** [Simmons, 1988] that were applied at *planning* time. A modification of these strategies can be used at *execution* time.

Modifying Simmons's debugging categories, we note several generic revision strategies (i.e., a set of conditions and one or more generic revision mechanisms or actions), depending on what seems to be the abstracted problem. Here are some examples:

I) Problem: **EXISTS**(*event-type*, <*arg*, *value*>*, *interval*) an event proposition of type *event-type* is true during *interval* as part of some pattern to be avoided, or as an identified cause for a modifiable parameter value.

The different possible revision strategies employ different revision mechanisms:

a) **CLIP**(*event-type*, *interval*).

For instance, if there is a toxic effect of chemotherapy drug, stop the current chemotherapy. This strategy might also achieve certain other desirable effects indexed off the clipping action for that event type.

b) **REPLACE**(*event-type*, *interval*, *new-interval*).

Replace the event, in a sequential manner, by another event of same type without the same restrictions or undesirable side effects (e.g., replace by another chemotherapy). This strategy typically includes a plan of the type <Clip(*event*, *interval*); Start(*new-event-type* <*arg*, *value*>*, *new-interval*)>.

c) **SUSPEND** <*event* > **UNTIL** <*condition*>

The *condition* depends on the original offending pattern. Several categories of conditions can exist. For instance, the chances of a negative effect of the event are reduced (e.g., the toxicity level due to the drug has decreased), or the chances of a desirable effect of the event are high enough (e.g., enough time has passed since another drug, contraindicated to the suspended one, had been administered)

d) **SHIFT-TIME**(*event-type*, <*arg*, *value*>*, *new-interval*>).

This strategy is relevant when reasoning about hypothetical plans (which have not been executed yet) or about *cyclic plans* that are repeated each time cycle (e.g., administration of the same dose of regular insulin each morning). The (repeating) time stamps of the event interval are changed, but not the event arguments (e.g., the same type of insulin will be administered in the same amount, but in the evenings instead of in the mornings).

e) **MODIFY-ARGUMENT**(*event, argument, value, new-value, mode*).

This strategy is relevant when the current event argument list has a certain value which might be creating the negative pattern, and which might be changed (e.g., by attenuating an overdose). In general, the *mode* argument can indicate whether the modification is permanent or transient and whether other conditions are to be checked. Modes might be quite different for various domains and contexts.

f) **ADD**(*new-event*, <*arg*, *value*>*, *new-interval*>).

Add a new event (using the START mechanism), whose effect counteracts an undesirable effect of the offending event, without clipping the offending event.

II) Problem: **ATTRIBUTE**(*parameter, abstraction-type, value, interpretation-context*). A certain abstraction of a particular parameter has a certain value at a certain interpretation context.

The revision strategies in this case rely on the knowledge that the parameter's value can be changed by increasing or decreasing the magnitude of some event that affects the parameter's values in the desired direction. Thus, if the Hb level is too low, we can transfuse blood, or we can reduce some argument of an event that is known to affect the Hb level negatively (e.g., by decreasing the dose of a drug that is toxic to the bone marrow). Such knowledge can be indexed by the *change* in the parameter. Thus, the revision strategy involves a revision mechanism (e.g., MODIFY-ARGUMENT) that can be indexed off any desired change in the relevant parameters (e.g., INCREASING, DECREASING).

Such indexing can be quite task-independent, being part of the general domain knowledge. The knowledge can be represented as a part of the general annotations of a task class in the domain, since it relies on a model of the domain: how certain events (actions or plans, i.e., one or more event propositions) effect the values of domain parameters. Such knowledge implies a library of actions and plans that lists their effects on various parameters. Thus, we need to include in the planner's knowledge base a set of **event-effect tuples** of the type <*event, argument, relation, parameter*> where *relation* can be `positively proportional`, `negatively proportional`, or, in general, a function type. Thus, the *argument* of *event* has the qualitative relationship *relation* to the value of *parameter*. (An *interpretation-context* argument can be added in certain cases.) Given such a library, most of the revision mechanisms can be indexed automatically from the appropriate parameters and changes.

7. Sources of Disagreement Between Human and Automated Planners

When the human and the automated planners disagree on a particular therapy plan, and the techniques of abstraction and plan recognition fail to explain the differences, there are several possible explanations that can assist an automated planner in reasoning about a discrepancy or that can justify it in a disciplined manner.

One class of explanations can be applied to disagreements that stem from various deficiencies in the database or in the knowledge base of one of the planners (most likely, the automated one, given its narrow channel of input data). Knowledge base deficiencies mainly include a missing event, a missing parameter, or a missing causal link between known events and parameters:

- a missing event-effect tuple that represents an unknown effect of a known event on a known parameter (e.g., an unknown relation between certain drugs used for reducing blood pressure, and heart rate)

- a missing event-effect tuple that represents an effect of a known event on an unknown parameter (e.g., a known antihypertensive drug also has an effect on the patient's unknown mental state)

- a missing event-effect tuple that represents an effect of a unknown event on a known parameter (e.g., blood pressure might be increased by smoking, an event unknown to the automated planner).

In addition, the knowledge base of *temporal abstractions* used for the recognition of temporal patterns of events and parameters might be deficient; apart from obvious omissions of events, of interpretation contexts, or of parameters and their temporal-abstraction properties [Shahar et al., 1992], the knowledge base might be missing a more specific interpretation context that should specialize the interpretation of a known parameter or of a class of parameters in the domain's parameter ontology [Shahar and Musen, 1993] (e.g., blood glucose is expected to have higher values in the context of a postprandial period, such as after lunch).

Missing data commonly include missing event intervals or parameter intervals in the input from the electronic database (e.g., unknown to the automated planner, the patient had a nervous breakdown).

Since generic and specific revision strategies are also part of the automated planner's knowledge base, lack of knowledge of such a strategy might also lead to an apparent discrepancy.

Finally, the lack of an explicit goal or policy in the annotated guideline would be a common reason for a seemingly inadequate recommendation.

There are also other, special, sources of disagreement, that require different types of explanations. The choice, by the physician, of an insulin regimen different from the one recommended by the automated planner might be justified by the physician as being due to a different *patient-oriented goal*. Thus, a particular patient might prefer a regimen involving fewer shots per day of a long-acting insulin, at the expense of less optimal control of blood glucose. Note that such an explanation requires some representation of a **patient model**—at least, a qualitative model of utilities of therapy options and their respective outcomes. Alternatively, the physician might justify her choice in the same case by mentioning that she is more comfortable using a short-acting insulin. Understanding and representing that explanation requires some form of a **physician model**. Thus, both patient and physician (user) preferences might affect the physician's decision when instantiating the skeletal plan represented by the clinical guideline.

In each of the cases described, the physician could indicate in a structured way what, in her opinion, is the source of the disagreement, thus facilitating the maintenance of the knowledge base and providing an automatic justification for her action, such as overriding a recommendation. Given that justification, the automated planner might avoid, for instance, an inappropriate recommendation during the patient's next visit.

8. Summary and Discussion

Knowledge-based decision support systems for human planners, such as physicians planning therapy, require considerable sophistication for even minimal acceptance. An automated (assistant) planner needs to adjust gracefully to the overriding of its suggestions by a physician who might have access to additional data or domain knowledge. However, the automated planner also needs to recognize the cases that fall within its knowledge and where its help might be significant, such as when a complex protocol has to be followed, and there is no obvious reason for changing the planner's recommendations. Thus, successful collaboration between human and automated planners requires some degree of reasoning about the mental states of the other planner. In this paper we focused on the automated planner's view.

Representing a complex clinical guideline or a protocol amounts to representing a highly reusable skeletal reactive plan that needs to be refined over time; goals and policies in that plan can be represented as temporal-abstraction patterns of events and parameters to be maintained, achieved, or avoided.

A plan-recognition ability is crucial for an automated planner that needs to support the physician's plans in the face of complex, ambiguous protocols and multiple methods for achieving similar goals. The plan-recognition ability requires knowledge about a set of domain-independent and domain-specific revision strategies and mechanisms. Reasoning about revision strategies, however, requires knowledge of qualitative relationships among event arguments and parameter values in different contexts. Note also that support of the application of guidelines also requires some opportunistic planning when a higher-level goal has been achieved by means other than those prescribed in the guideline, or an action has already been performed by the human planner.

Explicit representation of the intentions underlying clinical guidelines, of a domain-independent knowledge base of revision mechanisms, and of event-effect tuples would contribute in at least two major ways: (1) enhanced functionality and flexibility of automated planners might increase the acceptance of their (provably useful) assistance in the increasingly common task of application of clinical guidelines, and (2) annotations to the guideline, and clear semantics for the domain knowledge base, should facilitate considerably the maintenance and reusability of the task-specific knowledge involved in the application of such guidelines.

Acknowledgments

This work has been supported in part by grant HS06330 from the AHCPR, by grants LM05157 and LM05305 from the NLM, and by gifts from Digital Equipment Corporation. Dr. Musen is a recipient of NSF Young Investigator Award IRI-9257578.

References

Hammond, K.J. (1989). *Case-Based Planning: Viewing Planning as a Memory Task.* San Diego, CA: Academic Press.

Hickam, D.H., et al. (1985). A study of the treatment advice of a computer-based cancer chemotherapy protocol advisor. *Annals of Internal Medicine* **103**(6), 928–936.

Miller, P.L. (1986). *Expert Critiquing Systems: Practice-Based Medical Consultation by Computer.* New York, NY: Springer-Verlag.

Musen, M.A., et al. (1987). Knowledge engineering for a clinical trial advice system: Uncovering errors in protocol specification. *Bulletin du Cancer (Paris)* **74**, 291–296. Originally in *Proceedings of the AAAMSI Congress 86*, American Association for Medical Systems and Informatics, pp. 24–27, Anaheim, CA, 1986.

Musen, M.A., et al. (1992). T-HELPER: Automated support for community-based clinical research. *Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care* (M. E. Frisse, Ed.) (pp. 719–723), New York, NY: McGraw Hill.

Shahar, Y., Tu, S.W., and Musen, M.A. (1992). Knowledge acquisition for temporal-abstraction mechanisms. *Knowledge Acquisition* **4**(2), 217–236.

Shahar, Y., and Musen, M.A. (1993). RÉSUMÉ: A temporal-abstraction system for patient monitoring. *Computers and Biomedical Research* **26**(3), 255–273. Reprinted in van Bemmel, J.H., and McRay, T. (eds) (1994), *Yearbook of Medical Informatics 1994*, pp. 443–461, Stuttgart: F.K. Schattauer and The International Medical Informatics Association.

Shahar, Y. (1994). *A Knowledge-Based Method for Temporal Abstraction of Clinical Data.* (Knowledge Systems Laboratory Report No. KSL-94-64, Department of Computer Science report No. STAN-CS-TR-94-1529). Doctoral dissertation. Stanford, California: Stanford University, Program in Medical Information Sciences.

Simmons, R.G. (1988). A theory of debugging plans and interpretations. *Proceedings of the Seventh National Conference on Artificial Intelligence* (pp. 94–99), St. Paul, Minnesota.

Tu, S.W., et al. (1989). Episodic skeletal-plan refinement based on temporal data. *Communications of the ACM* **32**(12), 1439–1455.

Tu, S.W., et al. (1992). A Problem-Solving Model for Episodic Skeletal-Plan Refinement. *Knowledge Acquisition* **4**(2), 197–216.

Van der Lei, J., and Musen, M.A. (1991). A model for critiquing based on automated medical records. *Computers and Biomedical Research* **24**(4), 344–378.