Formulating diagnostic problem solving using an action language with narratives and sensing

Chitta Baral
Department of CSE
Arizona State University
Tempe, AZ 85287
chitta@asu.edu

Sheila McIlraith
Knowledge Systems Lab
Stanford University
Stanford, CA 94305
sam@ksl.stanford.edu

Tran Cao Son
CS Department
Univ. of Texas at El Paso
El Paso, Texas 79968
tson@cs.utep.edu

Abstract

Given a system and unexpected observations about the system, a diagnosis is often viewed as a fault assignment to the various components of the system that is consistent with (or that explains) the observations. If the observations occur over time, and if we allow the occurrence of (deliberate) actions and (exogenous) events, then the traditional notion of a candidate diagnosis must be modified to consider the *possible occurrence of actions* and events that could account for the unexpected system behavior.

In the presence of multiple candidate diagnoses, we may need to perform actions and observe their impact on the system, to be able to narrow the list of possible diagnoses, and possibly even initiate some repair. A plan that guarantees such narrowing will be referred to as a *diagnostic plan*, and if this plan also guarantees that at the end of the execution of the plan, the system has no faults then we refer to it as a *repair plan*.

Since actions and narrative play a central role in diagnostic problem solving, we characterize diagnosis, diagnostic planning and repair with respect to the existing action language $\mathcal{L}$, extended to include static constraints, sensing actions, and the notion of observable fluents. This language is used to provide a uniform account of diagnostic problem solving.

1 Introduction

Consider the following narrative involving diagnosis.

> John gets up in the morning. He turns on the switch of his lamp, and reads the morning newspaper. He then turns off the switch and does other things before going to work. After he gets home from work, he enters his room and turns on the switch of his lamp again. This time, *the lamp does not turn on.* John thinks that maybe either the bulb is broken, or the switch of the lamp is broken, or the power cord is broken, or there is no power at the outlet. He does nothing about it and goes to his bathroom and turns on the light switch, observing that even that light does not turn on. He thinks perhaps there is no power at home, but then he notices that his electric clock is working, so he figures that there is power in at least part of his home. Now he is worried and goes to his garage to check his fuse box and finds that one of the fuses is blown. He replaces that fuse and comes back to his room. He turns on his lamp switch and voila it works.

This narrative illustrates the process of diagnostic problem solving. In particular it illustrates that diagnostic problem solving must involve reasoning about the evolution of a dynamical system. Triggered by an observation of system behavior that is inconsistent with expected behavior – in this case, the fact that when John turned on the lamp it did not emit light, diagnostic problem solving involves:

• generating candidate diagnoses based on an incomplete history of events that have occurred and observations that have been made.

• in the event of multiple candidate diagnoses, performing actions to enable observations that will discriminate candidate diagnoses. The selection of a particular actions is often biased towards confirming the

most likely diagnosis, or the one that is easiest to test.
• generating plans (possibly with conditionals and sensing actions) to perform these discriminatory observations.
• updating the space of diagnoses in the face of changes in the state of the world, and in the face of new observations.

The long-term objective of our work is to develop a knowledge representation and reasoning capability that emulates diagnostic problem solving processes such as John's. Following [McI97b], we argue that such a comprehensive account of diagnostic problem solving must involve reasoning about action and change. In this paper we augment and extend the work of [McI97a, McI98, McI97b] in several important ways. The main contributions of this paper are:
• We define diagnosis with respect to a narrative.
• We define the notion of diagnostic and repair planning, within a language that integrates sensing actions and world-altering actions. Thus, we are able to distinguish between changes in the state of the world, and changes in an agent's state of knowledge.
• In support of this endeavor, we extend the action language $\mathcal{L}$ to support static causal laws, sensing actions and the notion of observable fluents. $\mathcal{L}$ was originally developed to support narratives (e.g., [MS94, Pin94]).

None of the above issues have been explored either in the model-based diagnosis literature or in the reasoning about action literature. Also notable is that unlike most other accounts of diagnosis, our account allows nondeterministic effects of actions. Finally, our work is distinguished from most previous work in defining diagnosis in terms of a diagnostic model, rather than in terms of failing components and/or actions sequences.

The rest of the paper is organized as follows. In Section 2 we give an overview of the language $\mathcal{L}$ and how to add static causal laws to it. In Section 3 we use the extended language to define when we may need to do a diagnosis and what a diagnosis is with respect to a narrative. In Section 4 we further extend our action language to allow sensing actions and to accommodate the distinction between an observable fluent and a unobservable fluent. We then use this language to define the notion of a conditional plan, and the related notions of diagnostic and repair planning. Finally, in Section 5 we summarize and discuss related work.

2 Specifying narrative in $\mathcal{L}$

The propositional language $\mathcal{L}$ was developed in [BGP97, BGP98] to specify narratives and to reason with them. In this paper, we will describe the main aspects of the language $\mathcal{L}$ by dividing it into three components: a domain description language $\mathcal{L}_D$, a language to specify observations $\mathcal{L}_O$, and a query language $\mathcal{L}_Q$. In Section 4.1, we extend our language further with sensing actions, and observables.

2.1 $\mathcal{L}_D$: The domain description language

The alphabet of $\mathcal{L}_D$ – a language that closely follows the language $\mathcal{AC}$ from [Tur97] – comprises two nonempty disjoint sets of symbols: the set of fluents $\mathbf{F}$, and the set of actions, $\mathbf{A}$. A *fluent literal* (or *literal*) is a fluent or a fluent preceded by $\neg$. A *fluent formula* is a propositional formula constructed from literals. Propositions in $\mathcal{L}_D$ are of the following forms:

$$a \textbf{ causes } \varphi \textbf{ if } \psi \tag{1}$$

$$\varphi \textbf{ if } \psi \tag{2}$$

$$\textbf{impossible } a \textbf{ if } \psi \tag{3}$$

where a is an action, and φ, and ψ are fluent formulas.

Propositions of the form (1) describe the direct effects of actions on the world and are called *dynamic causal laws*. Propositions of the form (2), called *static causal laws*, describe causal relation between fluents in a world. Propositions of the form (3), called *executability conditions*, state when actions are not executable.

A *domain description* D is a set of propositions in $\mathcal{L}_D$.

The main difference between $\mathcal{L}_D$ and the action description part of $\mathcal{L}$ [BGP97, BGP98] is the presence of static causal laws in $\mathcal{L}_D$, which are critical for representing the behavior of the device being diagnosed.

A domain description given in $\mathcal{L}_D$ defines a transition function from actions and states to a set of states. (Recall, actions may be nondeterministic.) Intuitively, given an action, a and a state, s the transition function $\Phi(a, s)$ defines the set of states that may be reached after executing the action a in state s. If $\Phi(a, s)$ is an empty set it means that a is not executable in s. We now formally define this transition function.

Let D be a domain description in the language of $\mathcal{L}_D$. An *interpretation* I of the fluents in $\mathcal{L}_D$ is a maximal consistent set of fluent literals of $\mathcal{L}_D$. A fluent f is said to be true (resp. false) in I iff $f \in I$ (resp. $\neg f \in I$). The truth value of a fluent formula in I is defined recursively over the propositional connective in the usual way. For example, $f \wedge q$ is true in I iff f is true in I and q is true in I. We say that φ holds in I (or I satisfies φ), denoted by $I \models \varphi$, if φ is true in I.

A set of formulas from $\mathcal{L}_D$ is *logically closed* if it is closed under propositional logic (wrt $\mathcal{L}_D$).

Let V be a set of formulas and K be a set of static causal laws of the form φ **if** ψ. We say that V is closed under K if for every rule φ **if** ψ in K, if ψ belongs to V then so does φ. By $Cn(V \cup K)$ we denote[1] the least logically closed set of formulas from $\mathcal{L}_D$ that contains V and is also closed under K.

A *state* of D is an interpretation that is closed under the set of static causal laws of D.

An action a is *prohibited (not executable)* in a state s if there exists an executability condition of the form

$$\textbf{impossible } a \textbf{ if } \varphi$$

in D such that φ holds in s.

The *effect of an action* a in a state s is the set of formulas $e_a(s) = \{\varphi \mid D$ contains a law a **causes** φ **if** ψ and ψ holds in $s\}$.

Given the domain description D containing a set of static causal laws R, we formally define $\Phi(a, s)$, the set of states that may be reached by executing a in s as follows.

1. If a is not prohibited (i.e., executable) in s, then

   $$\Phi(a, s) = \{s' \mid Cn(s') = Cn((s \cap s') \cup e_a(s) \cup R)\};$$

2. If a is prohibited (i.e., not executable) in s, then $\Phi(a, s)$ is $\emptyset$.

The intuition behind the above formulation is as follows. The direct effects (due to the dynamic causal laws) of an action a in a state s are given by $e_a(s)$, and all formulas in $e_a(s)$ must hold in any resulting state. In addition, the static causal laws (R) dictate additional formulas that must hold in the resulting state. While the resulting state should satisfy these formulas, it must also be otherwise closed to s. These three aspects are captured by the definition above. For additional explanation and motivation behind the above definition please see [Tur97].

2.2 $\mathcal{L}_O$: The observation language

We assume the existence of a set of situation constants $\mathbf{S}$ which contains two special situation constants s_0 and s_c denoting the initial situation and the current situation, respectively. Note that *situations* written as s (possibly with subscripts) are different from *states* which are written as s (possibly with subscripts). As with the situation calculus, the ontology of our language differentiates between a situation, which is a history of the actions from the initial situation, and a state, which is the truth value of fluents at a particular situation.

Observations in $\mathcal{L}_O$ are propositions of the following forms:

$$\varphi \textbf{ at } \mathsf{s} \tag{4}$$

$$\alpha \textbf{ between } \mathsf{s}_1, \mathsf{s}_2 \tag{5}$$

$$\alpha \textbf{ occurs\_at } \mathsf{s} \tag{6}$$

$$\mathsf{s}_1 \textbf{ precedes } \mathsf{s}_2 \tag{7}$$

where φ is a fluent formula, α is a (possibly empty) sequence of actions, and $\mathsf{s}, \mathsf{s}_1, \mathsf{s}_2$ are situation constants which differ from s_c. (Since the world can be changed without the agent's knowledge, we do not allow the agent to have observations about s_c.)

Observations of the forms (4) and (7) are called *fluent facts* and *precedence facts*, respectively. Observations of the forms (5) and (6) are referred to as *occurrence facts*. These two types of observations are different in that (5) states exactly what happened between two situations s_1 and s_2, whereas (6) only says what occurred in the situation s.

2.3 Narratives

A *narrative* is a pair (D, Γ) where D is a domain description and Γ is a set of observations of the form (4)-(7).

Observations are interpreted with respect to a domain description. While a domain description defines a transition function that characterize what states *may* be reached when an action is executed in a state, a narrative consisting of a domain description together with a set of observations defines the possible situation histories of the system. This characterization is achieved by two functions, Σ and Ψ. While Σ maps situation constants to action sequences, Ψ picks one among the various transitions given by $\Phi(a, s)$ and maps action sequences to a unique state with the condition that $\Psi(\alpha \circ a) \in \Phi(a, \Psi(\alpha))$.

More formally, let (D, Γ) be a narrative. A *causal interpretation* of (D, Γ) is a partial function from action sequences to interpretations of $Lang(\mathbf{F})$, whose

[1] Note that a fluent formula φ can be equivalently represented as a static causal law φ **if** *true*.

domain is nonempty and prefix-closed[2]. By $Dom(\Psi)$ we denote the domain of a causal interpretation Ψ. Notice that $[] \in Dom(\Psi)$ for every causal interpretation Ψ, where $[]$ is the empty sequence of actions.

A *causal model* of D is a causal interpretation Ψ such that:

(i) $\Psi([])$ is a state of D; and

(ii) for every $\alpha \circ a \in Dom(\Psi)$, $\Psi(\alpha \circ a) \in \Phi(a, \Psi(\alpha))$.

A *situation assignment* of $\mathbf{S}$ with respect to D is a mapping Σ from $\mathbf{S}$ into the set of action sequences of D that satisfy the following properties:

(i) $\Sigma(\mathsf{s}_0) = []$;

(ii) for every $\mathsf{s} \in \mathbf{S}$, $\Sigma(\mathsf{s})$ is a prefix of $\Sigma(\mathsf{s_c})$.

An *interpretation* M of (D, Γ) is a pair (Ψ, Σ), where Ψ is a causal model of D, Σ is a situation assignment of $\mathbf{S}$, and $\Sigma(\mathsf{s_c})$ belongs to the domain of Ψ. For an interpretation $M = (\Psi, \Sigma)$ of (D, Γ):

(i) α **occurs_at** s is true in M if the sequence $\Sigma(\mathsf{s}) \circ \alpha$ is a prefix of $\Sigma(\mathsf{s_c})$;

(ii) α **between** $\mathsf{s}_1, \mathsf{s}_2$ is true in M if $\Sigma(\mathsf{s}_1) \circ \alpha = \Sigma(\mathsf{s}_2)$;

(iii) φ **at** s is true in M if φ holds in $\Psi(\Sigma(\mathsf{s}))$;

(iv) s_1 **precedes** s_2 is true in M if $\Sigma(\mathsf{s}_1)$ is a prefix of $\Sigma(\mathsf{s}_2)$.

An interpretation $M = (\Psi, \Sigma)$ is a *model* of a narrative (D, Γ) if:

(i) facts in Γ are true in M;

(ii) there is no other interpretation $M' = (\Psi, \Sigma')$ such that M' satisfies condition i) above and $\Sigma'(\mathsf{s_c})$ is a subsequence of $\Sigma(\mathsf{s_c})$.

Observe that these models are minimal in the sense that they exclude extraneous actions.

A narrative is *consistent* if it has a model. Otherwise, it is *inconsistent*.

[2] A set X of action sequences is prefix-closed if for every sequence $\alpha \in X$, every prefix of α is also in X.

2.4 $\mathcal{L}_Q$: The query language

Queries in $\mathcal{L}_Q$ are of the following form:

$$\varphi \textbf{ after } \alpha \textbf{ at } \mathsf{s} \tag{8}$$

When α in (8) is an empty sequence of actions, and s is the current situation $\mathsf{s_c}$, we often use the notation **currently** φ as a simplification of (8).

A query of the form φ **after** α **at** s is true in a model $M = (\Psi, \Sigma)$ if φ is true in $\Psi(\Sigma(\mathsf{s}) \circ \alpha)$.

A query q is entailed by a narrative (D, Γ), denoted by $(D, \Gamma) \models q$, if q is true in every model of (D, Γ).

3 Diagnosis wrt narratives

We are now ready to formulate the notion of diagnosis with respect to a narrative. The representation of the system to be diagnosed comprises static causal laws that describes the behavior of the system itself, as well as the description of the effects of actions on system state, and observations about action occurrences and fluent values over the evolution of the system We follow the diagnosis literature (e.g., [DMR92]) and assume that the system is composed of a distinguished set of components that can malfunction. Associated with each component c, is the distinguished fluent $ab(c)$, denoting that the component c is abnormal or broken. Also associated with each component is the distinguished fluent $break(c)$, a wildcard action which may be used to explain unexpected observations about $ab(c)$. Note that the representation of the system is likely to contain other actions and static causal laws that affect the truth of $ab(c)$. Building on the established diagnosis notation:

Definition 1 (System)
A *system Sys* is a tuple $(SD, COMPS, OBS)$ where

$COMPS = \{c_1, \ldots, c_n\}$ is a finite set of components.

SD is a domain description characterizing the behavior of the system, and augmented with dynamic laws of the form $break(c)$ **causes** $ab(c)$, for each component c in $COMPS$.

Given SD, by SD_{ab}, we denote the subset of SD consisting of static causal laws of the form "ψ **if** φ" and dynamic laws of the form "a **causes** ψ **if** φ", where ψ contains $ab(c)$ for some component c.

OBS is a collection of observations starting from the situation s_1, and the precedence fact

s_0 **precedes** s_1. Specification of fluent facts at s_0 are not included in OBS.

In our formulation of diagnosis, we make the assumption that there is an initial situation in the history where all components are operating normally[3]. This is achieved by adding the set $OK_0 = \{\neg ab(c) \ \textbf{at} \ s_0 \mid c \in COMPS\}$ to our observations.

Example 1 Consider a slight variation of the story in our introduction. Assume that the only breakable component in the domain is the *bulb*. Furthermore, assume that John observed that the light is off *immediately* after he turned on the lamp when coming back from work. The story can then be described by a system description $Sys_0 = (SD_0, \{bulb\}, OBS_0)$, where

SD_0 :

$$
\begin{array}{c}
turn\_on \ \textbf{causes} \ light\_on \ \textbf{if} \ \neg ab(bulb) \\
turn\_off \ \textbf{causes} \ \neg light\_on \\
\neg light\_on \ \textbf{if} \ ab(bulb) \\
break(bulb) \ \textbf{causes} \ ab(bulb) \\
\textbf{impossible} \ break(bulb) \ \textbf{if} \ ab(bulb)
\end{array}
$$

OBS_0 :

$$
\begin{array}{rl}
turn\_on \ \textbf{occurs\_at} \ s_1 & s_0 \ \textbf{precedes} \ s_1 \\
turn\_off \ \textbf{occurs\_at} \ s_2 & s_1 \ \textbf{precedes} \ s_2 \\
turn\_on \ \textbf{between} \ s_3, s_4 & s_2 \ \textbf{precedes} \ s_3 \\
\neg light\_on \ \textbf{at} \ s_1 & s_3 \ \textbf{precedes} \ s_4 \\
light\_on \ \textbf{at} \ s_2 & \\
\neg light\_on \ \textbf{at} \ s_3 & \\
\neg light\_on \ \textbf{at} \ s_4 &
\end{array}
$$

OK_0 : $\neg ab(bulb) \ \textbf{at} \ s_0$. □

Intuitively, we say a system needs a diagnosis, if the following assumptions are inconsistent with the observations (i) all components are initially fine, and (ii) no action that can break a component occurs. To define diagnosis, we assume that all components were initially operating normally, and we try to conjecture minimal action occurrences to account for the observations. Since the semantics of $\mathcal{L}$ minimizes action occurrences, all we need to do is to consider the various models of the narrative and extract our diagnosis from each.

[3] Hence, our formulation of diagnosis can be alternately referred to as 'big-bang diagnosis'. We can slightly modify it to define *incremental diagnosis*, when we already know that a set $X \subseteq COMP$ is abnormal, and we want to figure out if some additional components have malfunctioned by having $OK_0 = \{ab(c) \ \textbf{at} \ s_0 \mid c \in X\} \cup \{\neg ab(c) \ \textbf{at} \ s_0 \mid c \in COMP \setminus X\}$.

Definition 2 (Necessity of Diagnosis) We say a system $Sys = (SD, COMPS, OBS)$ needs a diagnosis if the narrative $(SD \setminus SD_{ab}, OBS \cup OK_0)$ does not have a model.

Note that the notion of a system needing a diagnosis is not meant to capture the notion that there is some fault in the system. It is a much weaker notion. We now establish the notion of a diagnosis in terms of a diagnostic model.

Definition 3 (Diagnostic Model) Let $Sys = (SD, COMPS, OBS)$ be a system that needs a diagnosis. We say M is a diagnostic model of *Sys* if M is a model of the narrative $(SD, OBS \cup OK_0)$.

We can now extract information about any particular situation from the diagnostic model. In particular,

Definition 4 (Diagnosis) A diagnosis with respect to situation s is the set of components $\Delta \in COMPS$ such that there exists a model $M = (\Psi, \Sigma)$ of the narrative $(SD, OBS \cup OK_0)$ and $\Delta = \{c \mid ab(c) \ \textbf{at} \ s \text{ holds in } M\}$. We refer to a diagnosis with respect to s_c as a *current fluent diagnosis*. We say a diagnosis Δ (wrt a situation s) is *minimal* if there exists no diagnosis Δ' (wrt s) such that $\Delta' \subset \Delta$.

Example 2 (Continuation of Example 1) Consider the system $Sys_0 = (SD_0, \{bulb\}, OBS_0)$, from Example 1, with $SD_{ab} = \{break(bulb) \ \textbf{causes} \ ab(bulb)\}$.

Let narrative $N_0' = (SD_0 \backslash SD_{ab}, OBS_0 \cup OK_0)$. Due to the proposition "$\neg light\_on \ \textbf{if} \ ab(bulb)$", $SD_0 \backslash SD_{ab}$ has only three distinct states: $s_0 = \emptyset$, $s_1 = \{light\_on\}$, and $s_2 = \{ab(bulb)\}$.

The transition function of $SD_0 \backslash SD_{ab}$ is given by

$$
\begin{array}{ll}
\Phi(turn\_on, s_0) = \{s_1\} & \Phi(turn\_off, s_0) = \{s_0\} \\
\Phi(turn\_on, s_1) = \{s_1\} & \Phi(turn\_off, s_1) = \{s_0\} \\
\Phi(turn\_on, s_2) = \{s_2\} & \Phi(turn\_off, s_2) = \{s_2\}
\end{array}
$$

We now prove that N_0' is inconsistent. Assume the contrary, N_0' has a model (Σ, Ψ). Because of $OBS_0 \cup OK_0$, we conclude that $\Psi([]) = s_0$. Let $\Sigma(s_3) = \alpha$, where α is an action sequence. By the definition of a model of a narrative, we have that $\Sigma(s_4) = \alpha \circ turn\_on$. As there is no action in $SD_0 \backslash SD_{ab}$ whose effect is $ab(bulb)$, we conclude that $ab(bulb) \notin \Psi(\alpha)$. This implies that $light\_on \in \Psi(\alpha \circ turn\_on)$, i.e., $light\_on$ must hold in s_4. This contradicts the observation "$\neg light\_on \ \textbf{at} \ s_4$", i.e., N_0' is inconsistent.

Narrative N_0' is inconsistent, and hence, Sys_0 needs a diagnosis. We compute the diagnosis as follows.

The narrative $N_0 = (SD_0, OBS_0 \cup OK_0)$, has one model $M = (\Psi, \Sigma)$ where $\Psi([]) = s_0$ and

$\Sigma(\mathsf{s}_0) = \Sigma_1(\mathsf{s}_1) = []$, $\Sigma(\mathsf{s}_2) = turn\_on$,
$\Sigma(\mathsf{s}_3) = turn\_on \circ turn\_off \circ break(bulb)$,
$\Sigma(\mathsf{s}_4) = \Sigma(\mathsf{s}_c) = turn\_on \circ turn\_off \circ break(bulb) \circ turn\_on$.

We can easily verify that $ab(bulb)$ **at** s_c is true in M. Hence a current diagnosis for Sys_0 is $\Delta = \{bulb\}$. Moreover, it is easy to check that Δ is also a minimal current diagnosis for Sys_0. □

3.1 Explanation vs diagnosis

Often the observations in a narrative can be *explained* by the sequence of actions (possibly exogenous) that have occurred. Unfortunately, this is not true in all cases because incomplete knowledge of the initial situation, and/or non-deterministic actions can lead to uncertainty in the outcome of a sequence of actions.

The definition of a diagnostic model in the previous section uses a *consistency* criterion to account for the observations. That is, the narrative (SD, Γ), where Γ comprises the sequence of action occurrences and initial situation (including OK_0) dictated by the diagnostic model, do not necessarily *entail* OBS. They are merely consistent with OBS. Here we define the notion of an explanatory diagnostic model, which has the stronger criterion that (SD, Γ) must entail OBS.

Definition 5 (Explanatory Diagnostic Model) Suppose $M = (\Psi, \Sigma)$ is a diagnostic model of (SD, $COMPS$, OBS), where

- $actions(M)$ is the set of occurrence facts and precedence facts of the forms (5), (6), and (7), (i.e., facts of the forms α **between** $\mathsf{s}_1, \mathsf{s}_2$, α **occurs_at** s_1, s_1 **precedes** s_2) that are true in M; and

- initial(M) is the set of fluent facts of the form f **at** s_0 that are true in M, including OK_0.

Then M is an *explanatory diagnostic model* iff

$$(SD,\ actions(M) \cup initial(M)) \models OBS$$

Following in this spirit, it is straightforward to define the notion of an explanatory diagnosis, a set of action occurrences that entails the observations.

4 Diagnostic and repair planning

The diagnostic process discussed in the previous section will generate a set of candidate diagnoses, however diagnosis is only the first step in dealing with an errant system. In most cases we will attempt to discriminate these diagnoses with the objective of identifying a unique diagnosis and/or reducing our space of candidate diagnoses to a point where a repair plan can be conceived. We are operating under the assumption that we cannot directly observe the state of abnormality of the various components of the system. Nevertheless, we can make other observations about the system, add them to OBS, and then refine our diagnoses.

In general, the fluents in the system are of two kinds: *observable* and *unobservable*. A simple *generic observation*[4] leads the agent to know the value of the observable fluents. By knowing the relationship between the observable and unobservable fluents, and the values of the observable fluents, we can sometimes deduce the values of unobservable fluents. We can also use direct sensing actions to sometimes determine their value. Given a set of candidate diagnoses, we can execute a plan – perhaps including some sensing actions and conditional branches – and make the generic observations to obtain additional information that will help reduce the space of possible diagnoses. Such plans are distinguished in that they can have knowledge goals in addition to goals relating to the state of the world. We refer to plans that attempt to reduce our space of diagnoses as *diagnostic plans*. A diagnostic plan that includes some repair is called a *repair plan*.

4.1 Adding sensing and observables to $\mathcal{L}$

In order to define the notion of a diagnostic plan, we must first augment $\mathcal{L}_D$ and $\mathcal{L}_Q$ to incorporate sensing actions and observable and unobservable fluents. In this section, we briefly describe these augmentations. The resulting theories are called $\mathcal{L}_{DS}$ and $\mathcal{L}_{QS}$, respectively. ($\mathcal{L}_O = \mathcal{L}_{OS}$.)

• We allow *knowledge producing laws* of the following form in $\mathcal{L}_{DS}$:

$$\mathsf{a}\ \textbf{determines}\ f \tag{9}$$

where a is an action and f is a fluent. A law of this form tells us that after a is executed, the value of the

[4] Here we distinguish between generic observations and sensing actions. We assume that the agent is constantly performing 'generic observations' and thus knows the truth value of the observable fluents at all times. In contrast, sensing actions require the agent's effort.

fluent f will be known. An action occurring in a knowledge producing law is called a *sensing action.*

• With the addition of sensing actions, we need to distinguish between a state of the world and the state of the agent's knowledge about the world. The later will be referred to as a *combined state* (or *c-state*) and will be represented by a pair of the form $\langle s, \mathcal{S}\rangle$, where s is a state (representing the real state of the world) and $\mathcal{S}$ is a set of states (representing the set of states an agent thinks it may be in).

• We extend the transition function Φ to also map pairs of actions and c-states into sets of c-states.

1. for any c-state $\langle s, \mathcal{S}\rangle$ and non-sensing action a,

   $\Phi(a, \langle s, \mathcal{S}\rangle) = \{\langle s', \mathcal{S}'\rangle \mid s' \in \Phi(a, s)$, and $\mathcal{S}'$ is the set of states in $\Phi(a, \mathcal{S})$ that agree with s' on $\mathbf{F_O}$, the **observable** literals $\}$.

   (Note that if a is not executable in $\langle s, \mathcal{S}\rangle$ then $\Phi(a, \langle s, \mathcal{S}\rangle) = \emptyset$.)

2. for any c-state $\langle s, \mathcal{S}\rangle$ and sensing action a whose knowledge producing laws are
   a **determines** f_1 ... a **determines** f_m
   (a) if a is executable in $\langle s, \mathcal{S}\rangle$, $\Phi(a, \langle s, \mathcal{S}\rangle) = \{\langle s, \{s' \mid s' \in \mathcal{S}$ such that s and s' agree on the literals from $\mathbf{F_O} \cup \{f_1, \dots, f_m\}\}\rangle\}$;
   (b) otherwise, $\Phi(a, \langle s, \mathcal{S}\rangle) = \emptyset$.

• In the presence of incomplete information and knowledge producing actions, there may not exist simple plans consisting of sequence of actions and we may need to extend the notion of a plan to allow conditional statements. We refer to such plans as conditional plans (e.g., [Lev96, BS97, BS98]), described below.

• In order to query the system, we specify a query language $\mathcal{L}_{QS}$. A query in $\mathcal{L}_{QS}$ has the form

$$\varphi \textbf{ after } P \textbf{ at } \mathsf{s} \tag{10}$$

where φ is a fluent formula and P is a conditional plan as formally defined below.

Definition 6 (Conditional Plan)

1. An empty sequence of action, denoted by [ ], is a conditional plan.

2. If a is an action then a is a conditional plan.

3. If $P_1, \dots, P_n$ are conditional plans and φ_j's are conjunction of fluent literals (which are mutually exclusive but not necessarily exhaustive) then the following is a conditional plan. (We refer to such a plan to as a *case plan*).

   Case
   $\varphi_1 \to P_1$
   ...
   $\varphi_n \to P_n$
   Endcase

4. If P_1 and P_2 are conditional plans then $P_1; P_2$ is a conditional plan.

5. Nothing else is a conditional plan.

In order to define when a narrative entails a query that includes a conditional plan, we need to define an *extended transition function* $\hat{\Phi}$, that maps a pair of a conditional plan and a c-state, into a set of c-states. Intuitively, if $\sigma' \in \hat{\Phi}(P, \sigma)$ then the execution of the plan in the c-state σ may take us to the c-state σ'. Before defining $\hat{\Phi}$, we first define the possible trajectories when P is executed in σ.

Definition 7 Let P be a conditional plan and σ be a c-state. We say a sequence of c-states $\sigma_1, \dots, \sigma_n$ is a trajectory of P wrt σ if:

1. $P = []$, and $n = 1$, and $\sigma_1 = \sigma$.

2. $P = [a]$, and $n = 2$, and $\sigma_1 = \sigma$ and $\sigma_2 \in \Phi(a, \sigma)$.

3. $P =$ Case
   $\varphi_1 \to P_1$
   ...
   $\varphi_n \to P_n$
   Endcase,

   and there exists an i such that φ_i is known to be true in σ and $\sigma_1, \dots, \sigma_n$ is a trajectory of P_i wrt σ.

4. $P = P_1; P_2$, and $\sigma_1 = \sigma$, and $\sigma_1, \dots, \sigma_k$ is a trajectory of P_1 wrt σ, and $\sigma_{k+1}, \dots, \sigma_n$ is a trajectory of P_2 wrt σ_{k+1}.

σ_n is referred to as the resulting c-state of P wrt σ.

Definition 8 Let P be a conditional plan and $\sigma = \langle s, \mathcal{S}\rangle$ be a c-state, $\hat{\Phi}(P, \sigma)$ is now defined as follows:

1. $\hat{\Phi}([], \sigma) = \{\sigma\}$;

2. For an action a, $\hat{\Phi}(a, \sigma) = \Phi(a, \sigma)$;

3. For $P =$ Case
$$\varphi_1 \to P_1$$
$$\ldots$$
$$\varphi_n \to P_n$$
Endcase,
$$\hat{\Phi}(P,\sigma) = \begin{cases} \hat{\Phi}(P_i,\sigma) & \text{if } \varphi_i \text{ is known to be true in } \sigma \\ \emptyset & \text{if there exists no } i \text{ s.t. } \varphi_i \text{ is known to be true in } \sigma \end{cases}$$

4. For $P = P_1; P_2$, where P_1 is a conditional plan and P_2 is a conditional plan,
   - if $\hat{\Phi}(P_1,\sigma) \neq \emptyset$, and for every $\sigma' \in \hat{\Phi}(P_1,\sigma)$, $\hat{\Phi}(P_2,\sigma') \neq \emptyset$, then $\hat{\Phi}(P,\sigma) = \bigcup_{\sigma' \in \hat{\Phi}(P_1,\sigma)} \hat{\Phi}(P_2,\sigma')$; and
   - $\hat{\Phi}(P,\sigma) = \emptyset$ otherwise.

It should be noted that $\hat{\Phi}(P,\sigma)$ is not equal to the set of resulting c-states of P wrt σ. This is because some branches of P may lead to unexecutable actions and hence $\hat{\Phi}(P,\sigma)$ will be empty while there may be several trajectories corresponding to other branches.

Our next goal is to define entailment of queries wrt narratives. Intuitively, since the narrative may not be complete, or have sufficient observations to arrive at a unique model, multiple models may tell us that a situation s may correspond to many different states, only one of which corresponds to s in reality. Thus we have a set of c-states from which we need to verify the correctness of a conditional plan with respect to a goal. More formally,

Definition 9 (Possible State wrt a Situation) Let $N = (D,\Gamma)$ be a narrative. We say s is a possible state corresponding to situation s, if there exists a model (Ψ,Σ) of N such that $\Psi(\Sigma(\mathsf{s})) = s$. We say $\sigma = \langle s,\mathcal{S}\rangle$ is a c-state corresponding to situation s, if s is a possible state corresponding to situation s and $\mathcal{S}$ is the set of all possible states corresponding to s.

A query $q = \varphi$ **after** P **at** s of $\mathcal{L}_{QS}$ is said to be entailed by narrative (D,Γ), i.e. $(D,\Gamma) \models q$, if for every c-state $\langle s,\mathcal{S}\rangle$ corresponding to s, $\hat{\Phi}(P,\langle s,\mathcal{S}\rangle) \neq \emptyset$ and φ is known to be true in every c-state belonging to $\hat{\Phi}(P,\langle s,\mathcal{S}\rangle)$.

4.2 Diagnostic and repair plans

We are now ready to define what a diagnostic plan is. Intuitively, it is a conditional plan, possibly with sensing actions which when executed in the current situation gives sufficient information to reach a unique diagnosis. In addition, we may have certain restrictions, such as that:

- Certain literals are not allowed to change during the execution of the plan. We refer to such literals as *protected literals.* (E.g., to stabilize the leaning tower of Pisa, we may not tear down and rebuilt it.)

- Certain literals are allowed to change during the execution of the plan, but we require that at the end of the execution of the plan, their value be the same as it was before the plan was executed. We refer to such literals as *restored literals.* (E.g., disassembling an engine or a flashlight to diagnose it, but putting it back together afterwards.)

- Certain literals are allowed to change during the execution of the plan, but we require that at the end of the plan, their value be either the same as it was before the plan was executed or be $false$. Such literals will be referred to as *fixable literals.* (This accommodates repair, where ab fluents can be made $\neg ab$.)

Definition 10 (Diagnostic Plan) Given $Sys = (SD, COMPS, OBS)$ with a set of protected literals L_P, a set of restored literals L_R, and a set of fixable literals L_F. Let $C \subseteq COMPS$. A conditional plan P is called a *diagnostic plan* for Sys wrt (C, L_P, L_R, L_F), if for every c-state $\sigma = \langle s,\mathcal{S}\rangle$ corresponding to the current situation of Sys, $\hat{\Phi}(P,\sigma) \neq \emptyset$ and for all trajectories of the form $\sigma_1,\ldots,\sigma_n$ of P wrt σ

(i) for every c-state $\langle s',\mathcal{S}'\rangle$ in $\hat{\Phi}(P,\langle s,\mathcal{S}\rangle)$ and $s'' \in \mathcal{S}'$, $s \sim_{AB(C)} s''$ where $AB(C) = \{ab(c) \mid c \in C\}$;

(ii) value of all literals in L_P remain unchanged (wrt the real states) in the trajectory;

(iii) for all literals l in L_R value of l (wrt the real states) in σ_1 and σ_n are the same; and

(iv) for all atoms f in L_F, if f is false in σ_1 then it is false in σ_n. (wrt the real state).

If $C = COMPS$, and the ab-literals are part of L_P, we say that P is a *purely diagnostic plan* for Sys.

If C is a singleton, i.e., $C = \{c\}$ for some $c \in COMPS$, and $ab(c)$ and $\neg ab(c)$ are in L_P, we say that P is a *discriminating diagnostic plan* for c.

Definition 11 (Repair Plan) A diagnostic plan P for Sys wrt (C, L_P, L_R, L_F) is said to be a *repair plan* wrt (C', L_P, L_R, L_F) if (i) $C' \subseteq C \subseteq COMPS$, (ii) ab literals about C' are not in L_P and L_R, and (iii) for every c-state $\sigma = \langle s,\mathcal{S}\rangle$ corresponding to the current situation of Sys, $\hat{\Phi}(P,\sigma) \neq \emptyset$ and for all $c \in C'$, $\neg ab(c)$ is known to be true in all c-states in $\hat{\Phi}(P,\langle s,\mathcal{S}\rangle)$.

Example 3 (Electro-magnetic Door)
Consider an electro-magnetic control door. The door is connected to a RED LED and a YELLOW LED. To enter, an agent needs to put its electro-magnetic card, containing its id-number and password, into the slot connected to the door's controller. The door will open only if the card is valid, the id-number and the password are not corrupted, and the door is not malfunctioning. While the card is in the slot, if it is invalid, the RED LED will be on; and if the id-number or the password is corrupted or the door is defective, the YELLOW LED will be on. The YELLOW LED is on only if the RED LED is not. In this case, pushing the button "message" will print out a message. Reading it, the agent will know whether the door is defective or its card is unreadable.

Our agent, Jack comes to work, and as usual, puts his card into the slot. The door does not open. What is wrong ? The story can be represented by the system $Sys_1 = (SD_1, \{card, door, id\_pwd\}, OBS_1)$ as follows.

The actions of the domain description SD_1 are: *insert_card*, *push_button*, *take_out_card*, look (look at the LEDs), or $\mathsf{read\_msg}$ (read the message).

The fluents of SD_1 are: *ab(card)*, *ab(door)*, *ab(id_pwd)*, *has_card*, *card_in_slot*, *door_open*, *has_msg*, *red*, and *yellow*, where *red* or *yellow* indicate that the RED/YELLOW LED is on, respectively.

SD_1 comprises the following laws:

• dynamic causal laws: describing the effects of the actions *insert_card*, *push_button*, and *take_out_card*. Inserting the card causes the door to open if the card, the door and the card information are all normal. Further, inserting the card causes the card to be in the slot and not in the possession of the agent. I.e.,

$$insert\_card \ \textbf{causes} \ \ door\_open$$
$$\textbf{if} \ \ \neg ab(card), \neg ab(door), \neg ab(id\_pwd)$$

$$insert\_card \ \textbf{causes} \ \ \neg has\_card \wedge card\_in\_slot$$

Pushing the button results in a message. I.e.,

$$push\_button \ \textbf{causes} \ \ has\_msg.$$

If the card is in the slot and the agent takes it out, then the agent has possession of the card and the card is not in the slot. I.e.,

$$take\_out\_card \ \textbf{causes} \ \ has\_card \wedge \neg card\_in\_slot$$
$$\textbf{if} \ \ card\_in\_slot$$

• static causal laws: expressing the relationship between the status (on/off) of the LEDs. I.e.,

$$red \ \textbf{if} \ \ ab(card) \wedge card\_in\_slot$$
$$\neg red \ \textbf{if} \ \ yellow \wedge card\_in\_slot$$
$$yellow \ \textbf{if} \ \ (ab(id\_pwd) \vee ab(door)) \wedge \neg red \wedge card\_in\_slot$$

• sensing actions: characterizing the knowledge effects of sensing actions. For example, performing the look action causes the agent to know whether the RED and YELLOW LEDs are on or off. They are captured by the following k-propositions:

$$\mathsf{look} \ \textbf{determines} \ red$$
$$\mathsf{look} \ \textbf{determines} \ yellow$$
$$\mathsf{read\_msg} \ \textbf{determines} \ ab(id\_pwd)$$
$$\mathsf{read\_msg} \ \textbf{determines} \ ab(door)$$

• executability conditions: characterizing when an action is precluded. I.e.,

$$\textbf{impossible} \ \ insert\_card \ \textbf{if} \ \ \neg has\_card$$
$$\textbf{impossible} \ \ push\_button \ \textbf{if} \ \ \neg yellow$$
$$\textbf{impossible} \ \ \mathsf{read\_msg} \ \textbf{if} \ \ \neg has\_msg$$

• wildcard actions:

$$break(card) \ \textbf{causes} \ \ ab(card)$$
$$break(door) \ \textbf{causes} \ \ ab(door)$$
$$break(id\_pwd) \ \textbf{causes} \ \ ab(id\_pwd)$$

and the set of observations, OBS_1:

$$\neg red \wedge \neg yellow \wedge \neg door\_open \ \ \textbf{at} \ \ \mathsf{s}_1$$
$$has\_card \wedge \neg has\_msg \ \ \textbf{at} \ \ \mathsf{s}_1$$
$$\neg card\_in\_slot \ \ \textbf{at} \ \ \mathsf{s}_1$$
$$insert\_card \ \textbf{between} \ \mathsf{s}_1, \mathsf{s}_2$$
$$\neg door\_open \ \ \textbf{at} \ \ \mathsf{s}_2$$
$$\mathsf{s}_0 \ \textbf{precedes} \ \ \mathsf{s}_1$$
$$\mathsf{s}_1 \ \textbf{precedes} \ \ \mathsf{s}_2$$

The first three observations describe the first observable situation, s_1. The fourth observation states that Jack puts his card into the slot, while the fifth states that the door is not open after Jack puts his card into the slot.

Intuitively, when Jack observes that the door does not open as the result of putting his card into the slot, he should realize that at least one of the three components: the card, the door, or the information on the card is no longer valid. Our diagnostic reasoning systems does likewise. Indeed, the narrative $N_1' = (SD_1 \setminus SD_{ab}, OBS_1 \cup OK_0)$ does not have a model and there are three diagnoses for Sys_1: $\Delta_1 = \{ab(id\_pwd)\}$, $\Delta_2 = \{ab(door)\}$, and $\Delta_3 = \{ab(card)\}$ which correspond to the models M_1, M_2, and M_3 of $N_1 = (SD_1, OBS_1 \cup OK_0)$ defined as follows. $M_1 = (\Psi_1, \Sigma_1)$, $M_2 = (\Psi_2, \Sigma_2)$, and $M_3 = (\Psi_3, \Sigma_3)$, where $\Psi_1([]) = \Psi_2([]) = \Psi_3([]) = s_0$, and

$$\begin{aligned}
&\Sigma_1(\mathsf{s}_0) = [],\\
&\Sigma_1(\mathsf{s}_1) = break(id_pwd),\\
&\Sigma_1(\mathsf{s}_2) = \Sigma_1(\mathsf{s}_c) = break(id_pwd) \circ insert_card,
\end{aligned}$$

$$\begin{aligned}
&\Sigma_2(\mathsf{s}_0) = [],\\
&\Sigma_2(\mathsf{s}_1) = break(door),\\
&\Sigma_2(\mathsf{s}_2) = \Sigma_2(\mathsf{s}_c) = break(door) \circ insert_card,
\end{aligned}$$

$$\begin{aligned}
&\Sigma_3(\mathsf{s}_0) = [],\\
&\Sigma_3(\mathsf{s}_1) = break(card),\\
&\Sigma_3(\mathsf{s}_2) = \Sigma_3(\mathsf{s}_c) = break(card) \circ insert_card.
\end{aligned}$$

where $s_0 = \{has\_card\}$,

$$\begin{aligned}
&\Psi_1(break(id_pwd)) = \{has_card, ab(id_pwd)\},\\
&\Psi_1(break(id_pwd) \circ insert_card)\\
&\qquad = \{card_in_slot, ab(id_pwd), yellow\} = s_1,
\end{aligned}$$

$$\begin{aligned}
&\Psi_2(break(door)) = \{has_card, ab(door)\},\\
&\Psi_2(break(door) \circ insert_card)\\
&\qquad = \{card_in_slot, ab(door), yellow\} = s_2,
\end{aligned}$$

$$\begin{aligned}
&\Psi_3(break(card)) = \{has_card, ab(card)\},\\
&\Psi_3(break(card) \circ insert_card)\\
&\qquad = \{card_in_slot, ab(card), red\} = s_3.
\end{aligned}$$

To narrow the list of the possible diagnoses of the system, Jack can find out the status of the LEDs. If the RED LED is on, he knows for sure that the card is no longer valid. Otherwise, the YELLOW LED must be on. In that case, he can get the message and read it to know if the door is broken or the information on the card is corrupted. This process is captured by the following plan.

$P = look \circ$
  case
    $red \to []$
    $\neg red \to$
      case
        $yellow \to push\_button \circ read\_msg$
      endcase
  endcase

We will now show that P is a diagnostic plan for Sys_1 wrt $(C, L_P, \emptyset, \emptyset)$ where $C = \{id\_pwd, dood, card\}$ and $L_P = \{ab(id\_pwd), ab(dood), ab(card)\}$.

Let $\mathcal{S} = \{s_1, s_2, s_3\}$. There are three possible current situations of Sys_1: $\sigma_1 = \langle s_1, \mathcal{S}\rangle$, $\sigma_2 = \langle s_2, \mathcal{S}\rangle$, and $\sigma_3 = \langle s_3, \mathcal{S}\rangle$. Let $s_i' = s_i \cup \{has\_msg\}$, $i = 1, 2$, then

$$\begin{aligned}
\hat{\Phi}(P, \sigma_1) &= \hat{\Phi}(push_button \circ read_msg, \langle s_1, \{s_1, s_2\}\rangle)\\
&= \hat{\Phi}(read_msg, \langle s_1', \{s_1', s_2'\}\rangle) = \{\langle s_1', \{s_1'\}\rangle\};
\end{aligned}$$

$$\begin{aligned}
\hat{\Phi}(P, \sigma_2) &= \hat{\Phi}(push_button \circ read_msg, \langle s_2, \{s_1, s_2\}\rangle)\\
&= \hat{\Phi}(read_msg, \langle s_2', \{s_1', s_2'\}\rangle) = \{\langle s_2', \{s_2'\}\rangle\};
\end{aligned}$$

$$\hat{\Phi}(P, \sigma_3) = \{\langle s_3, \{s_3\}\rangle\}.$$

The above computations also represent all trajectories of P wrt σ_1, σ_2, and σ_3. Obviously, $\hat{\Phi}(P, \sigma_i) \neq \emptyset$ for $i = 1, 2, 3$. Furthermore, it is easy to check that the values of literals in L_P remain unchanged in all trajectories and in each c-state $\langle s', \mathcal{S}'\rangle$ belonging to $\hat{\Phi}(P, \sigma_i)$ and $s'' \in \mathcal{S}'$, $s' \sim_{AB(C)} s''$. For example, $\langle s_1', \{s_1'\}\rangle$ is the only c-state in $\hat{\Phi}(P, \sigma_1)$, and trivially, $s_1' \sim_{AB(C)} s_1'$. Thus, P is a diagnostic plan for Sys_1 wrt $(C, L_P, \emptyset, \emptyset)$. □

5 Summary and Related Work

In this paper we provided an account of diagnostic problem solving in terms of the action language, $\mathcal{L}$. A prime objective of this work was to characterize diagnostic problem solving with narrative and sensing. $\mathcal{L}$ proved ideal for this task because it already had most of the necessary expressive power. In particular, $\mathcal{L}$ includes narrative, sensing actions, and additionally nondeterministic actions, which are common in diagnostic domains. In this paper, we extended $\mathcal{L}$ by adding static causal laws that are necessary for describing the behavior of the systems we diagnose. We also distinguished notions of observable fluents, and protected, restored and fixable fluents.

The main contributions of this paper, in addition to the supporting language extensions, are the characterization of the diagnosis task as a narrative understanding task, and the definition of diagnosis in terms of a diagnostic model – a particular model of the narrative. We further distinguish between a diagnostic model and the stricter notion of an explanatory diagnostic model. As discussed throughout the paper, diagnostic problem solving is more than just determining a set of candidate diagnoses. In the second half of the paper, we define the notion of a diagnostic plan, and a repair plan – conditional plans that exploit both world-altering actions and sensing actions with the goal of achieving some diagnostic knowledge or repair objective. These present new contributions to the research on model-based diagnosis and reasoning about action.

We contrast our contributions to related work. In the area of diagnosis of dynamical systems, there has been research both within the control theory community (e.g., [SSLST96]) on the diagnosis of discrete event systems using finite state automata, and within the AI community. Most of this work is fairly recent, and can be differentiated with respect to the expressive power of the language used to model the domain (e.g., propositional/first order, ramifications, nondeterministic actions, concurrent actions, narrative, sensing, probabilities); how the notion of diagnosis is defined (e.g., models, sequences of actions, sets of abnormal components, probabilistic criteria); how observations are expressed; whether diagnosis is active or offline; and what aspects of diagnostic problem solving, beyond diagnosis, are addressed (e.g., diagnostic planning, repair).

Our work was influenced by previous work of McIlraith (e.g., [McI97a, McI98, McI97b]), but extends and builds on aspects of that work in several important ways. [McI97b] argued that a comprehensive account of diagnostic problem solving must involve reasoning about action and change, and provided such an account in a dialect of the situation calculus that included causal ramification constraints, but did not include nondeterministic actions, sensing actions or narrative. Aberrant behavior was assumed to be caused by unobserved exogenous actions. Multiple definitions of diagnosis were provided both in terms of sequences of actions that explained the observations, and designations of normal and abnormal components with respect to a situation. The notion of a diagnostic model was not employed. Most importantly, this account did not exploit narrative for expressing and accounting for observations, consequently the assertion of observations and exogenous actions was much less elegant. [McI97b] also introduced the notion of testing to discriminate hypotheses, and analogues to the ideas of diagnostic and repair planning; however since the dialect of the situation calculus she employed did not include knowledge-producing actions, the important integration of sensing and world-altering actions that was done in this paper, was argued for but was left to future work.

Also of note is the work of Thielscher on a theory of dynamic diagnosis in the fluent calculus [Thi97]. Thielscher characterizes diagnoses in terms of minimally failing components, where his minimization preference criterion is with respect to the abnormalities in the initial state, but can additionally exploit some a prior likelihood. Thielscher does not exploit exogenous actions to account for abnormalities as we do, and does not allow for the occurrences of actions beyond what are observed. Thielscher does not take his work beyond a characterization of diagnosis.

A third important piece of work from the AI community is the work of Cordier, Thiébaux and their coauthors, (e.g., [TCJK96, CT94]). Their work is similar in spirit to ours, viewing the diagnosis task as the determination of an event-history of a system between successive observations. While this work is related, the representation of the domain uses state transition diagrams and is much less expressive and elaboration tolerant than ours. That said, their representation system is sufficiently expressive for the power distribution domain they have been examining, and more recently, their work has focused on the necessary tradeoffs required to address hard computational issues associated with their domain. Cordier and Thiébaux also discuss the notion of repair planning, but without distinguishing between sensing and world-altering actions.

Other notable work on the diagnosis of dynamical systems includes the work of Nayak and Williams on online mode identification for the NASA remote agent system (e.g., [WN96]), the work of Baroni et al. on the diagnosis of large active concurrent systems [BLPZ99], and work on temporal aspects of diagnosis by Brusoni et al. (e.g., [BCTD98]).

In the area of diagnostic and repair planning, Sun and Weld [SW93] proposed a decision-theoretic planner which was invoked by a diagnostic reasoner to plan repair actions. The associated planning language distinguished between information-gathering and state-altering actions, but did not provide for the specification of knowledge or diagnostic goals. Similarly Heckerman et al. [HBR94] have examined the problem of interactively generating repair plans under uncertainty using Bayes nets, a single fault assumption and a myopic lookahead heuristic. Actions are limited to simple observations and component replacement. In contrast Friedrich et al. (e.g., [FN92]) developed a set of greedy algorithms to choose between performing simple observations and repair actions, assuming a most likely diagnosis. They do not limit their system to repair alone but rather generalize their goal to some notion of purpose; purpose does not include specification of diagnostic goals. Finally, and perhaps most notably, Rymon [Rym93] developed a goal-directed diagnostic reasoner and companion planner, called TraumAID 2.0. The primary task of the diagnostic reasoner was to generate goals for the planner and to reason about whether those goals were satisfied.

While the above work has some of the same goals as our work, none has an explicit notion of knowledge, and hence the integration of sensing and world-altering actions is engineered, rather than treated formally within a logic. In the spirit of such integration, we point to the work of [SL93] and more recently [Lev96] and [DL99] as examples of actions theories with knowledge and sensing; however, to the best of our knowledge ours is the first action theory that allows both sensing and narrative.

References

[BLPZ99] P. Baroni, G. Lamperti, P. Pogliano and M. Zanella Diagnosis of large active systems *Artificial Intelligence*, 110(1):135–183, 1999.

[BGP97] C. Baral, M. Gelfond, and A. Provetti. Representing Actions: Laws, Observations and Hypothesis. *Journal of Logic Programming*, 31(1-3):201–243, 1997.

[BGP98] C. Baral, A. Gabaldon, and A. Provetti. Formalizing Narratives using nested circumscription. *Artificial Intelligence*, 104(1-2):107–164, 1998.

[BS97] C. Baral and T. Son. Regular and special sensing in robot control - relation with action theories. In *Proc. of AAAI 97 Workshop on Robots, Softbots, and Immobots - Theories of Action, Planning and Control*, 1997.

[BS98] C. Baral and T. Son. Relating theories of actions and reactive control. *Electronic Transactions on Artificial Intelligence*, 2(3-4), 1998.

[BCTD98] V. Brusoni, L. Console, P. Terenziani, and D. Theseider Dupré. A spectrum of definitions for temporal model based diagnosis. *Artificial Intelligence*, 102:39–79, 1998.

[CT94] M. Cordier and S. Thiébaux. Event-based diagnosis for evolutive systems. Technical Report 819, IRISA, Cedex, France, 1994.

[DL99] G. DeGiacomo and H. Levesque. Projection using regression and sensors. In *IJCAI 99*, pages 160–165, 1999.

[DMR92] J. de Kleer, A.K. Mackworth and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56(2–3):197–222, 1992.

[FN92] G. Friedrich and W. Nejdl. Choosing observations and actions in model-based diagnosis/repair systems. In *Proc. of KR 92*, pages 489–498, 1992.

[HBR94] D. Heckerman and J. Breese and K. Rommelse. Troubleshooting under uncertainty. Microsoft Research, Technical Report MSR-TR-94-07, 1994.

[Lev96] H. Levesque. What is planning in the presence of sensing? In *Proc. of AAAI 96*, pages 1139–1146, 1996.

[McI94] S. McIlraith. Generating tests using abduction. In *Proc. of KR 94*, pages 449–460, 1994.

[McI97a] S. McIlraith. Representing actions and state constraints in model-based diagnosis. In *Proc. of AAAI 97*, pages 43–49, 1997.

[McI97b] S. McIlraith. *Towards a formal account of diagnostic problem solving*. PhD thesis, University of Toronto, 1997.

[McI98] S. McIlraith. Explanatory diagnosis: Conjecturing actions to explain observations. In *Proc. of KR 98*, pages 167–177, 1998.

[MS94] R. Miller and M. Shanahan. Narratives in the situation calculus. *Journal of Logic and Computation*, 4(5):513–530, 1994.

[Pin94] J. Pinto. *Temporal Reasoning in the Situation Calculus*. PhD thesis, University of Toronto, Department of Computer Science, 1994.

[Rei87] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

[Rym93] R. Rymon. Diagnostic reasoning and planning in exploratory-corrective domains. PhD thesis, University of Pennsylvania, 1993.

[SSLST96] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen and D. Teneketzis. Failure diagnosis using discrete-event models. *IEEE Trans. on Control Systems Technology,* vol. 4, no. 2, pp. 105-124, 1996.

[SL93] R. Scherl and H. Levesque. The frame problem and knowledge producing actions. In *Proc. of AAAI 93*, pages 689–695, 1993.

[SW93] Y. Sun and D. Weld. A framework for model-based repair. In *Proc. of AAAI 93*, pages 182–187, 1993.

[TCJK96] S. Thiébaux, M.-O. Cordier, O. Jehl and J.-P. Krivine. Supply Restoration in Power Distribution Systems – A Case Study in Integrating Model-Based Diagnosis and Repair Planning. In *Proc. of UAI 96*, pages 525–532, 1996.

[Thi97] M. Thielscher. A theory of dynamic diagnosis. *ETAI*, 2(11), 1997. Available electronically at http://www.ep.liu.se/ea/cis/1997/011.

[Tur97] H. Turner. Representing actions in logic programs and default theories. *Journal of Logic Programming*, 31(1-3):245–298, 1997.

[WN96] B. Williams and P. Nayak. A model-based approach to reactive self-configuring systems. In *AAAI 96*, pages 971–978, 1996.