AI Magazine Volume 11 Number 3 (1990) (© AAAI)

Hoist: A Second-Generation Expert System Based on Qualitative Physics

J. Douglass Whitehead and John W. Roach

This article describes a causal expert system based on hypothetical reasoning and its application to the maintenance of the lower hoist of a Mark 45 turret gun. The system, Hoist, performs fault diagnosis without the use of a repair expert or shallow rules. Its knowledge is coded directly from a structural specification of the Mark 45 lower hoist. The technology reported here for assisting the less experienced diagnostician differs considerably from normal rule-based techniques: It reasons about machine failures from a functional model of the device. In a mechanism like the lower hoist, the functional model must reason about forces, fluid pressures, and mechanical linkages; that is, it must reason about qualitative physics. Hoist technology can be directly applied to any exactly specified device for the modeling and diagnosis of single or multiple faults. Hypothetical reasoning, the process embodied in Hoist, has general utility in qualitative physics and reason maintenance.

Through the technology of expert systems, the expertise of highly skilled personnel can be automated and used to assist lesser skilled personnel in the diagnosis and repair of complex machines. Expert systems that incorporate causal reasoning represent a second-generation approach to the provision of diagnostic assistance. The technology involved performs postdiction by reasoning from first principles.

This article is based on research in qualitative physics and the philosophy of causality. A new implementation vehicle for causal reasoning is described, one that embodies hypothetical or counterfactual reasoning (Roach, Eichelman, and Whitehead 1985) in a language called Wif (What IF). This language was designed specifically for modeling cause-effect relationships. The application considered here applies postdiction analysis to a model of part of a naval weapon. The maintenance aid in use, the Hoist expert system, performs diagnostic assistance for the lower hoist of a Mark 45 naval turret gun. Conventional rule-based expert systems for diagnostic advice heuristically classify the cause of failure from malfunction symptoms alone (Clancey 1985). Unlike Hoist, these systems do not provide a model of the object under analysis. In the Hoist system for hypothetical reasoning, Wif models the functionality of all the components, thereby creating a causal model of the Mark 45 lower hoist. This model not only simulates the correct operation of the lower hoist, it also simulates the boundaries of faulty operation. The lower hoist in its actual form (a machine in the real world) is not computer accessible; a computer cannot connect to the machine, make an alteration, and observe the results. However, Hoist contains a causal model, is computer accessible, and can identify single or multiple components whose failure could explain the faulty hoist behavior.

Conventional rule-based expert systems attempt to capture an expert's opinion. A *causal reasoning expert system* significantly differs from this approach because it does not rely on advice. Instead, the causal expert system, which reasons from first principles, relies on a qualitative model of the physics of the device. This article explains the new approach to the postdiction process in three sections. The first section, Problems of Conventional Expert Systems, investigates several problems of conventional diagnostic expert systems and the role causal reasoning can play in their solution. The second section, Causal Reasoning, probes the nature of causal reasoning and introduces Wif and Hoist. Finally, the third section presents results, comments, and conclusions.

Problems of Conventional Expert Systems

Conventional expert systems, sometimes called *shallow reasoning systems*, have at least three major shortcomings in fault diagnosis. First, shallow reasoning is incapable of handling unanticipated faults. Second, a significant time lag exists between the initial construction of a device and the development of a conventional expert system for maintenance. Third, devices commonly go through a series of design modifications, and these modifications can affect the correctness of a shallow reasoning fault adviser.

Unanticipated Faults

Conventional expert systems cannot handle unanticipated situations. Typically, if a fault has not been anticipated, a shallow reasoning diagnosis either halts with an incorrect answer or supplies little or no information on the suspect components: The performance of a conventional expert system does not gracefully degrade. A causal expert system uses expected versus known machine states to converge on a faulty component and, thus, has the potential to tell the user that a fault exists between two points.

Development without a Repair Expert

Conventional expert systems depend on having an expert diagnostician to emulate. In the field of fault diagnosis, the machine must break several times before an expert diagnostician is developed. To be of use to a knowledge engineer, a diagnostician must adequately understand the device and be able to articulate knowledge of frequently observed and probable faults. When a complex system is involved, many years can pass before the development of an experienced expert who adequately understands observed and probable faults. If it then takes an additional year to produce a system to emulate the expert, there is a substantial lag between the time when the machine is first produced and the time the expert system for maintenance is made available.

An expert system that reasons from first principles (that is, uses causal reasoning) requires a specification of the function of the device components to diagnose faults. A mechanical engineer could obtain this information from device blueprints; thus, a new product could be sold with its repair adviser included in the package.

Maintainability

Alterations are part of the evolution of producing modern machines. Some original designs are modified as components are found to be overstressed in field testing. Also, high sales volume tends to spawn a series of similar products, each with its own unique characteristics. It is unclear how correct a shallow reasoning maintenance adviser would be after such modifications are made. Experience with R1 (Bachant and McDermott 1984) shows that maintenance of the knowledge can become burdensome. Conventional expert systems are produced by observing the expert and imitating his(her) behavior. This behavior is a result of value judgments based on the device as it existed when the expert was interviewed. Additionally, some judgments are made without the knowledge of the expert. If at some later time, the device is slightly modified, how is the knowledge engineer to know how many value judgments are affected? Is the knowledge engineer to reconstruct the entire knowledge base each time an alteration is made? Causal reasoning is based on the structure of the device. If a functional alteration is made, the model must be updated accordingly. That is, in the model, the structure of the existing part must be altered to represent the structure of the new part. Causal reasoning systems for fault diagnosis are easily maintained.

Causal reasoning systems for fault diagnosis are easily maintained.

Causal Reasoning

A repair expert might have some general rules for isolating faults, but s/he does not follow these rules exclusively, as shallow reasoning expert systems would imply. Instead, s/he understands the purpose of the machine and knows its expected behavior. When s/he observes something that is unexpected, s/he initiates a process of deduction to explain the deviation. S/he arrives at one of two conclusions: either the diagnostician’s expectations of the device are found incorrect and in need of updating (as is likely with a new repair person) or postdiction isolates the faulty component. In either case, *causal reasoning*, reasoning based on a causal chain of events, is the tool.

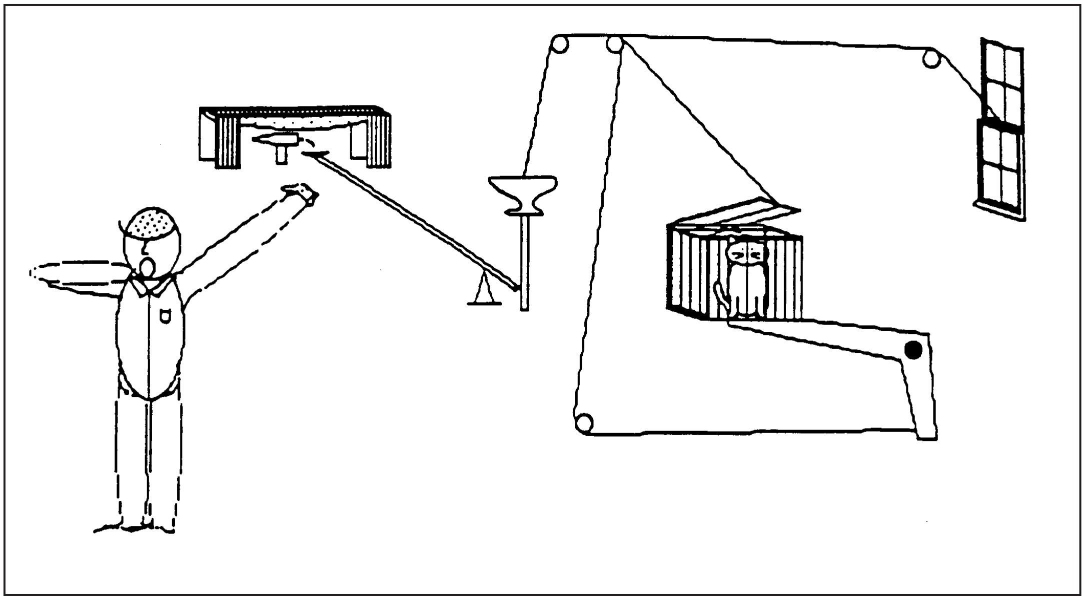


Figure 1. How to Put Your Cat out at Night.

Somehow we can follow that the sleepy fellow lies down on the cot, which causes the cot to sag, which causes the toothpaste to be squashed out of its tube, which causes the extra weight to move the lever, which causes the support of the anvil to be knocked off balance, which causes the string attached to the anvil to be pulled, which causes (1) the window to open; (2) the cage lid to be removed; and (3) the lever attached to the cage to be pulled, which causes kitty to be gingerly "catapulted" out the window for a feline night on the town.

Modeling in Hoist required the implementation of some form of qualitative physics. Qualitative physics and causal reasoning are strongly related, and both have been characterized elsewhere (DeKleer and Brown 1984; Forbus and Brown 1984; Roach, Eichelman, and Whitehead 1985). The next subsection presents an overview of some of these characteristics and discusses their representation in Hoist.

Qualitative Physics

Qualitative physics is the study of the behavior of the world in inexact terms. It has been suggested that humans perceive, understand, and generate expectations about the physics of the world in an imprecise form. *Causality* is the study of how to represent what happens as a result of some action. Wif is a hypothetical reasoning system that can be used to model causality. It is used to emulate the physics in the hydraulic-electronic-mechanical world of the Mark 45 lower hoist. Qualitative reasoning with physics about physical processes occurs so commonly in our understanding of the world that we are rarely aware of it. Consider, for example, the humorous Rube Goldberg device displayed in figure 1. The problem is clearly underconstrained, yet we can easily anticipate the outcome.

Differential equations rarely, if ever, help us calculate the consequence of an action in the world. We can apply fundamental, qualitative knowledge of physics even in highly novel situations. Deducing the consequences of actions on the world has a long tradition in AI and has come to be known as the *frame problem* (McCarthy and Hayes 1969).

Hypothetical reasoning as embodied in Wif is expressive enough to solve the frame problem for Hoist. Given an initial world W_0 and a set of counterfactuals X (that is, X is a set of facts not necessarily consistent with W_0), Wif generates a set of worlds $\{W_1, W_2, . . . , W_n\}$ that represent consistent worlds similar to W_0 but include X. *Causal rules* encode the interconnections between facts within a world. Hence, causal relationships take the form of rules (later called causal equations). In a world with the causal relationships of the Goldberg device of figure 1, a rule such as the following would exist:

*(on *something cot) → (sags cot)* .

Thus, for any rule in the world, if the precondition (the left-hand portion) of the rule is true, then the consequence (the right-hand portion) must also be true. Variables, denoted by an asterisk prefix, can assume any value. Thus, the previous rule states that in any possible world where it is true that something is on the cot, it is also true that the cot sags.

The Level of Abstraction

The atomic elements of the model consist of black boxes. We know that a black box has internal components, but they are not represented in the model because (1) further detail in the model is inappropriate to diagnosis-fixing (the purpose of the model in the first place), (2) we do not have adequate time-memory or other computational resources to use greater detail, or (3) we don't really know what goes on inside the black box. Hence, a model can have uniform granularity, even though black boxes might differ in internal complexity—hypothetical or real.

Before beginning to create rules to model causal relationships, the level of abstraction (granularity, level of detail) of the representation must be carefully chosen. A high level of abstraction is computationally efficient but less expressive in that some important functionality might not be represented. A low level of abstraction (for example, colliding molecules in a hydraulic pipe) allows the modeling of a multitude of phenomena but with a consequent increase in memory requirements and computational complexity. The world should be modeled only with the necessary level of detail for the problem at hand.

Hoist uses Boolean representations to simplify the model whenever possible. Hydraulic pressures and voltages are represented by Boolean variables (that is, high pressure or voltage is represented by on and low pressure and voltage by off). Mechanical linkages are quantized into discrete positions (for example, a given piston might have three positions: on, off, and center). At the level of abstraction chosen, a few components do not translate well into the Hoist model. An orifice, whose purpose is to restrict the rate at which a hydraulic line can change value, is currently modeled no differently than a hydraulic pipe. The high level of abstraction chosen for Hoist is computationally efficient and is sufficient for representing most components of the Mark 45 lower hoist.

Causal Influence Is Local

The principle of *locality* (DeKleer and Brown 1984; Forbus and Brown 1984) states that no single causal relation influences the behavior of a set of causal relations, except through the influence of its neighbors.

In Hoist, the function of each electric, hydraulic, or mechanical part is modeled as a set of causal equations.

In the Rube Goldberg device in figure 1, it appears that the sleepy fellow's act of lying down would cause the cat to be put out. However, this situation should not be modeled as

*(on *something cot) → (is_out cat)* .

Such a rule would violate the principle of locality, in that the action of lying down would dictate the performance of the entire Goldberg device. If such a rule were used, it would be impossible to represent a scenario where the fellow had forgotten to reset the anvil. This objection might sound silly, but it exemplifies the purpose of qualitative physics. That is, qualitative physics is useful because it provides a formalism that can represent in a concise way all the consequences of some action in an arbitrary world.

Causality should be modeled with local influence whenever possible. Granularity should not be increased in modeling; rather, the level of granularity should be consistent. That is, one should not change levels of abstraction while modeling local influence. However, levels of abstraction can be switched to obtain a new perspective on a problem (Davis 1984).

Modeling the Hoist

The lower hoist is part of a naval cannon; it is the transfer mechanism from the ammunition storage room to the ready magazine of the gun. Because of the poor retention rate of skilled repair persons and the complexity of modern weapons, the United States Navy has set computer-assisted maintenance of existing machinery as a priority. The lower hoist is complex: It has approximately 150 components, most of which are pipes. It also contains two solenoids, seven pistons, three latches, four state-detecting switches, a linkage, a chain, a hydraulic rack, and a clutch mechanism.

In Hoist, the function of each electric, hydraulic, or mechanical part is modeled as a set of causal equations. Any component can only have direct influence over neighboring components. No one component can directly

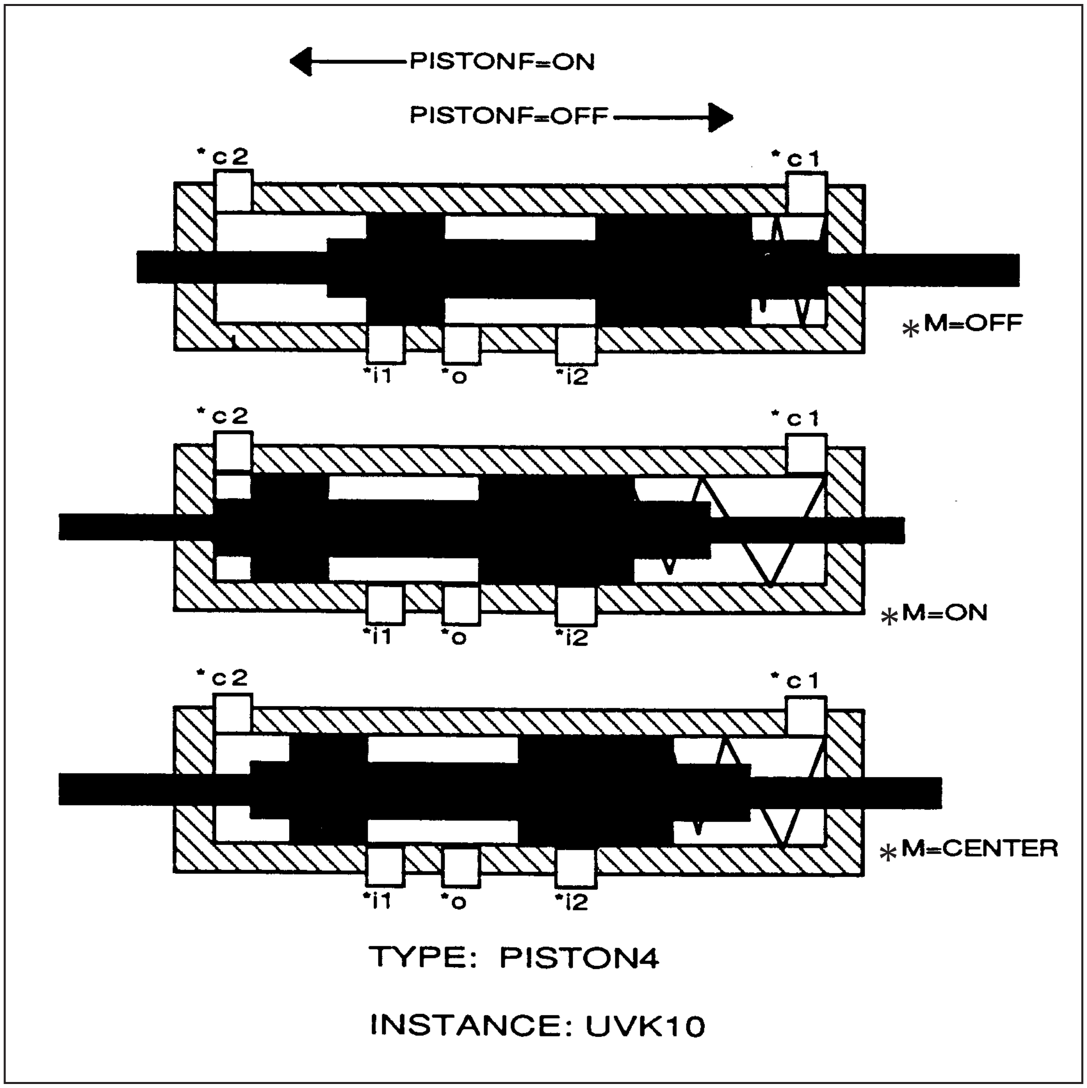


Figure 2. Hydraulic Schematic of Piston UVK10 in Three Positions.

influence the behavior of the entire machine. In the example that follows, two components (a piston and a block) are presented to illustrate the implementation of the principles of causal modeling and locality in Hoist. In figure 2, *c1, *c2, *i1, *i2, and *0 are hydraulic pressure lines. *pistonf is the direction of force exerted by the piston as a result of the relative pressures of *c1, *c2, and a spring. *m is the mechanical position of the piston. Figure 2 shows a picture of piston UVK10 in three different mechanical positions: *m = off, *m = on, and *m = center.

The dark section of each drawing represents the piston itself, and the slashed area represents the piston housing. The piston has a spring on the right-hand side. UVK10 is modeled by two different sets of causal relationships. The first set of causal equations (equation set 1) specifies that *c1, *c2, and the spring determine the direction of force of the piston (that is, the direction the piston would travel if uninhibited):

(*c1 = off) and (*c2 = on) → (*pistonf = off)
{push against spring}
(*c1 = on) → (*pistonf = on)
{reinforce spring default}
(*c2 = off) → (pistonf = on)
{no resistance to spring default}

Equation Set 1. *cl, *c2, and spring Determine the Direction of Force of the Piston.

Equation set 2 dictates that the mechanical position *m of the piston specifies whether *i1 or *i2 is connected to the output *0:

(*F = off) → (*i2 = *0)
{figure 2a, *i2 and *0 are joined}
(*m = on) → (*i1 = *0)
{figure 2b, *i1 and *0 are joined}
(*m = center) → (*i1 = *0)
{figure 2c, *i1 and *0 are joined}

Equation Set 2. The Mechanical Position (*m) Determines Whether *0 Is Connected to *i1 or *i2.

Note that force along the piston (*pistonf) and the mechanical position of the piston (*m) are variables that can assume values and are modeled no differently than hydraulic pressure. *c1 and *c2 of equation set 1 have no direct influence over *m of equation set 2, even though all three variables are part of the same piston. The only influence that *c1 and *c2 can have over *m is that they specify *pistonf, and *pistonf is connected to some neighboring device that is eventually connected back to *m. This sequence of interconnections is referred to as a *logical pipe* and is described later.

The second component to be modeled is called a *block.* This block is a simplification of another piston in the lower hoist. The block can move vertically and can be found in only one of two positions. When the block is in position *pos1*, the slot in the block is aligned with the latch. When the block is in position *pos2*, the slot is not aligned with the latch. *f_latch is the direction of force the latch is exerting. *latch is the mechanical position of the latch.

This model represents part of the behavior of the latch tongue in a simple mechanical latch. The block can move up or down. (Actually, the block is prevented from moving up when *latch = on, but this equation is not shown here.) If the slot of the block is aligned with the latch (that is, *block = pos1), then the tongue can move as force *f_latch dictates. However, if *block is in position *pos2* while *_latch is on, the most *latch can do (equation set 3, figure 3) is rest against the block (that is, *latch = center).

(*f_latch = off) → (*latch = off)
{latch can always travel to position off)
(*f_latch = on) and (*block = pos1) → (*latch = on)
{latch is allowed to travel to position on}
(*f_latch = on) and (*block = pos2) → (*latch = center)
{latch is prevented from traveling to position on}

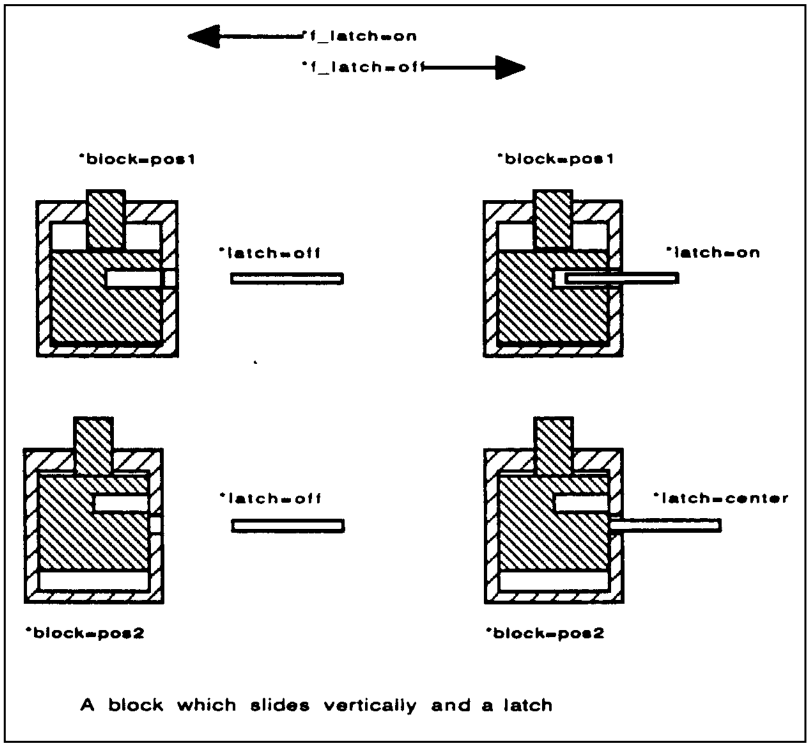


Figure 3. A Block That Slides Vertically and a Latch in Four Different Positions.

Equation Set 3. With Latch Force (*f_latch) in Direction On, Only *block in pos1 Will Allow *latch to Actually Move to Position On.
The pictorial representation of equation set 3 appears in figure 3.

Now connect these three sets of causal equations. Place a connection between *pistonf of equation set 1 and *f_latch of equation set 3. Also, place a connection between *latch of equation set 3 and *m of equation set 2. The combined device created by this union is pictured in figure 4. No pipes are depicted connecting *f and *f_latch or *m and *latch. These variables are not hydraulic pressures; they are forces and mechanical position indicators and are not well captured in a picture. In the case of *f and *f_latch, the direction of force of the piston UVK10, *f, causes a similar direction of force on the latch tongue *f_latch. This causal connection is similar to the properties of hydraulic pipes. The value at one end of the connection (be it hydraulic pressure, direction of force, or mechanical position) must be the same as the value at the other end. This simple form of causal equation is labeled a *pipe.* Hydraulic connections of this form are called *physical pipes.* Force connections and mechanical linkages are

The principle of locality provides a guide for model construction.

called *logical pipes*. The causal equations indicating the connections between variables for the two logical pipes in figure 4 are expressed in equation set 4.

We can see in figure 4 that **c1* and **c2* of equation set 1 effectively drive a mechanical latch and that the state of the latch is indicated by **0*. The dissection of the latch into subcomponents that interrelate only by affecting their neighbors' variables was useful for reducing the complexity of modeling. Further, any design change in UVK10 that has the same variables will not affect the block or any other device in Hoist.

$\rightarrow (*f = *f\_latch)$ {under any condition $*f = *f\_latch$}
$\rightarrow (*m = *latch)$ {under any condition $*m = *latch$}

Equation Set 4. Two Logical Pipes.
Physical and logical pipes are useful constructs for modeling simplification. They connect independently modeled units.

It might seem that the last component in a causal chain, say, component *X*, has an output that can be interpreted as the behavior of the whole device (assuming that the device has only one output). It might be argued that the principle of locality is violated by *X*. This observation is incidental because the device can be augmented by an extra component after *X* at some future date. This extra component can change the output of the device as a whole, but the functionality of component *X* remains the same. Thus, component *X* adheres to the principle of locality, and the fact that its output coincides with the functionality of the entire device is unimportant.

Wif: Hypothetical Reasoning

The principle of locality provides a guide for model construction. We now concentrate on the mechanism that uses the model to perform causal reasoning. The mechanism for causal reasoning in Hoist is hypothetical reasoning, also known as *counterfactual reasoning* (Rescher 1964; Roach, Eichelman, and Whitehead 1985).

Counterfactual reasoning is unlike first-order logic because it must successfully reconcile contradictory statements. A counterfactual clause is presumed to contradict currently believed facts, but the clause is assumed true for the sake of argument. However, a relationship between clauses must be maintained. Rules allow one to represent the laws of the modeled domain. For example, a mobile robot domain might include rules defining equality and inequality, location (because things can only be in one place at one time), and position (if a robot holds something, it must be in the same room). In Rescher's (1964) theory, formulas are assigned an ordering or modal category as a means of indicating relative believability. The modal category indicates what formulas will be cast out first when a counterfactual is introduced.

A rule in the counterfactual logic takes the form of

Rule 1: $P, Q, R \rightarrow S, T$.

This rule has three clauses for preconditions and two clauses for consequences. It is to be interpreted as follows: When *P*, *Q*, and *R* match formulas in the knowledge base (that is, when *P*, *Q*, and *R* are true in a world), *S* and *T* must also match formulas in the knowledge base (*S*, *T* must be true in this world). If either *S* or *T* is not true, then consistency must be restored if possible.

A world is made up of a set of consistent facts and a set of rules defined based on the facts. If a counterfactual is introduced, then the rules themselves suggest possible worlds where the counterfactual would be an element of the set of consistent facts. Counterfactual reasoning uses restoration to generate all possible worlds similar to the initial world, with the addition of counterfactuals. The following is a simplified restoration algorithm:

```
FOR every counterfactual clause C intro-
duced into world W
  begin
  IF ~C is a counterfactual THEN fail
  Remove ~C from W
  For every rule R in W
     IF C is in R's preconditions [left-hand
     side] OR ~C is part of R's consequences,
         THEN begin
         IF R's preconditions are in W AND
         R's consequences are not in W,
         THEN
             invoke RESTORATION with
             copies of W and copies of the
             counterfactuals that are aug-
             mented by the addition of
                 a) The set of R's consequences
                 (which, in effect, implements
                 modus ponens)
                 b) The negation of each
                 member of R's preconditions
                 (which, in effect, implements
                 modus tollens)
                 c) The negation of rule R
                 (which removes the
                 contradicted relationship
                 between facts)
         end
  end
```

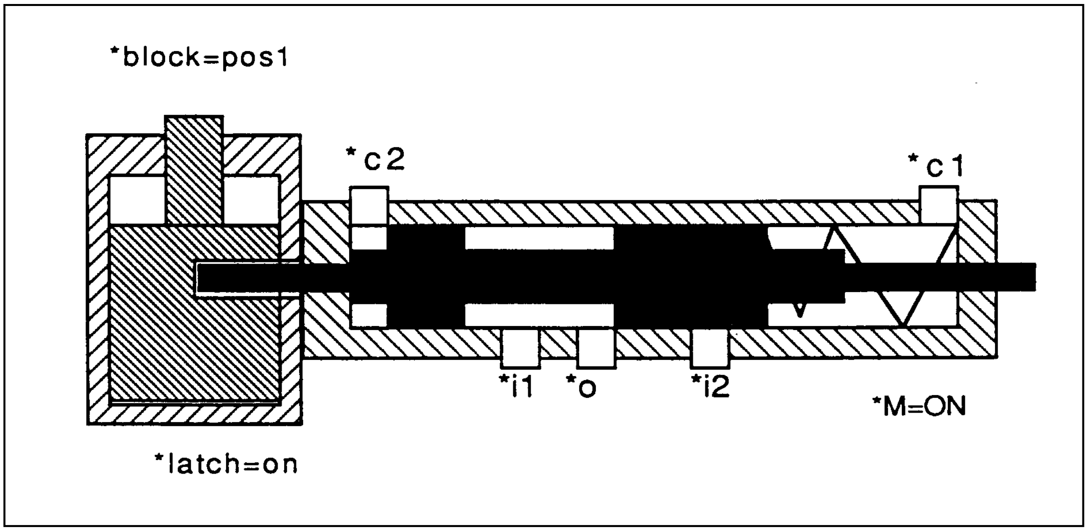


Figure 4. The Entire Hydraulically Powered Mechanical Latch, as Modeled in Equation Sets 1 through 4.

Consider a world W_0 containing only *P*, *R*, all other assumed negated facts, and rule 1 involving *P* through *T*. Now, if the counterfactual *Q* is introduced (*Q* is a counterfactual because ~*Q* is assumed to be true in W_0), rule 1 in W_0 (with counterfactual *Q*) has all its preconditions met in W_0 (that is, *P*, *Q*, and *R* are true). *S* and *T*, however, are not true in W_0 . Rule 1 itself suggests that a possible world that would contain *Q* would be one similar to W_0, except that *S* and *T* must also be true in this world (which corresponds to selection a in the restoration algorithm). Two other possible worlds where *Q* could exist are one with ~P and another with ~R because if the precondition of the rule is not true in a world, then the consequence need not be true in this world (which corresponds to selection b in the restoration algorithm). Further, a world that does not contain rule 1 is also a possible world (which corresponds to selection c in the restoration algorithm). In more complicated examples, these suggested alterations to W_0 must recursively invoke restoration with themselves as counterfactuals because the possible world must be consistent.

In the restoration algorithm, a possible world is found when the algorithm ceases to recur without failure (that is, a recursive invocation successfully terminates). A complete algorithm should return the set of possible worlds where the counterfactuals exist. An implementation that explicitly returns a set of worlds would be unwieldy. In Wif, the tree of differences from the original world is returned. Each invocation of the restoration algorithm generates a node in the tree that consists of the newly introduced counterfactuals. If this invocation causes recursion, then this node has a subtree associated with it. The method of difference trees avoids having to reconstitute the entire knowledge base. Each traversal from branch to root of the tree collects a set of alterations of the original world, which define a new world where the counterfactual is consistent (figure 5). Thus, the set of all branch-to-root traversals defines the set of all possible world solutions where the counterfactual is consistent.

Results, Comments, and Conclusions

In building Hoist, we wanted to compute what single and multiple component failures could explain malfunctions of the lower hoist. After a hoist malfunction, various internal machine states are known because of internal state-detecting switches. Thus, the question is asked, Why are components *X*, *Y*, and *Z* in states *A*, *B*, and *C*? This question is a request for postdiction.

Postdiction

Our theory of diagnosis is simple and intuitive: (1) A properly functioning device has predictable behavior. (2) Malfunction is detected by deviation from this behavior. (3) A set of properly functioning devices placed in combination has predictable behavior. (4) A device that malfunctions is composed of two sets: Set A is a finite set of properly functioning subdevices, and Set B is a finite nonempty set of malfunctioning subdevices. (5) Only elements of set B can explain deviated behavior. (6) It requires less effort to isolate the subdevices in set B if one assumes that *|B| = 1*, which is known as the *single fault assumption.*

By design, Wif generates all possible worlds where a solution exists. In the diagnostic domain, Wif generates all possible worlds where the machine would exhibit the observed behavior. Thus, Wif generates all possible sets of malfunctioning subdevices *B* and the internal states of the machine when subdevices *B* malfunction. Wif automatically diagnoses all possible simultaneous faults in the machine for a set of symptoms.

The number of solutions when allowing multiple faults can typically be large. Take, for example, a widget that during some test phase exhibits 10 output values contrary to the widget's correct performance. The malfunction of a single component might explain all the observed incorrect output values. If we allow multiple simultaneous faults, however, there might be 10 faults, each of which is immediately before one of the observed incorrect output values. The number of combinations of faults for complex machines is tremendous, and this combinatorial explosion is an unavoidable consequence of diagnostic systems allowing multiple faults.

Given the structure of the widget and the observed output, Wif generates all possible sets of faults and, thus, encounters the problem of combinatorial explosion. Combinatorial explosion leads to disastrous run times but is unavoidable if one wishes to isolate multiple faults. In Hoist, an additional constraint was added to curb combinatorial explosion: the single fault constraint. In other words, we request that Wif generate all possible worlds explaining the observed phenomena in which at most one component malfunctions.

The single fault assumption is merely a solution constraint. The assumption of as many as two faults is also a solution constraint. Hoist allows the user to specify any value of *n*, where *n* specifies the maximum number of faults allowable in the solution. By invoking Hoist multiple times, a user or a program can first search for a single fault, then two faults, three faults, and so on, until the fault(s) are isolated. Such a strategy solves the simple problems first and addresses the more general and more difficult problems only if needed. The user systematically relaxes constraints, thereby incurring only as much search (and as much computation) as needed.

Controlling the number of possible faults allows one kind of constraint on the search space. Heuristic search can also be added by the user. A heuristic would allow a search of a subset of the entire search space that still tends to yield solutions. Any heuristic a user might have should be presented with the fault isolation request to Wif. The added constraint of the heuristic will accordingly define a subset of the entire search space. If the user's hypothesis is correct, the solution will still be in the reduced search space, and Wif will find it. Wif itself is not a heuristic for multiple fault isolation; it is a tool for implementing a heuristic for multiple fault isolation.

Wif requires no patch or reconstitution to diagnose multiple faults; indeed, Wif addresses the multiple fault problem by default. In Hoist, we included the single fault assumption to prune the search space. Wif can diagnose multiple faults after the single fault assumption fails, thereby achieving speedy results on simple problems without giving up the capacity to solve the more difficult problems. The capability of isolating single and multiple faults is more than a neat feature of our counterfactual reasoning system: In science, the simplest explanation is usually perceived as the closest to the truth.

Postdiction on the Lower Hoist

The example of the UVK10 forming a latch is used to exemplify Hoist postdiction. For the purpose of this section, assume that the values of **c1*, **c2*, and **0* in figure 4 are directly verifiable by someone who reads a dial on the real machine. Assume further that all other internal states cannot be immediately checked. Unverifiable internal states are the norm in the lower hoist because direct observation of parts usually requires a hydraulic shutdown that destroys the state to be observed.

Assume that both **c1* and **c2* are known to be off, and **0* is known to be on. Figure 4 pictures the latch with input **c1* and **c2* off when the latch is working as designed. The predicted state of **0* is off because **0* is connected to **i1*, and **i1* is off. This predicted state contradicts the known observation that **0* is on. Wif is invoked with a statement that corresponds to the following:

Hypothesize that *(*c1 is off), (*c2 is off), (*0 is on),* and the single fault assumption. What deductions can be made using the structure of the device? A partial trace of the execution of Wif shows how this hypothesis is resolved.

*c1 and *c2 are off as in the model (*c1 and *c2 are not allowed to change value because they are part of the hypothesis). *0 is on, which does not correspond with the model. If *0 is on, then *0 is not off (because of a rule of mutual exclusion). In causal equation set 2, a relationship is asserted between *i1 and *0 when UVK10 is in mechanical position on (*m is on). Thus, if *0 is not off, then one of three possibilities exists:

A: UVK10 is not functioning as designed.
B:*i1 is not off.
C:*m is not in position on.

Because the full trace of these options can be tedious, we concentrate here on the most interesting and instructive alternative, C. If *m is not on, then two possibilities exist (because *m must be one of three values):

C1: *m is center.
C2: *m is off.

Possibility C1: If *m is center, then *latch must be center (by equation set 4). The only way *latch could be center is if *block is in pos2, and *f_latch is on. *f_latch is confirmed on; however, *block was thought to be in pos1; so, remove the (*block is pos1) fact. Back to the matter of *m being center, it follows that *i1 must not be off. Thus, *i1 must be on. The following substitutions to the original world would repair consistency:

{(*0 is on), (*0 is not off), (*m is not on),
(*m is center), (latch is center), (*latch is not on),
(*block is pos2), (*block is not pos1),
(*i1 is not off), (*i1 is on)} .

Possibility C2: If *m is off, then *latch must be off. If *latch is off, then one of the two following possibilities exists:

C2A: BLOCK is not acting as designed.
C2B: *f_latch is off.

Possibility C2A: If the block is not acting as designed, then the performance of the block is not predictable. Thus, *f_latch being on and *latch being off is acceptable. The following alterations would restore consistency in the original world:

{(*0 is on), (*0 is not off), (*m is not on), (*m is off)
(*latch is off), (*latch is not on), (BLOCK malfunction)} .

Possibility C2B: If *f_latch is off, then *f must be off (by equation set 4). *f can be off in one of two possible ways:

C2B1: *c1 is off and *c2 is on.
C2B2: UVK10 is not performing as designed.

Possibility C2B1: *c1 is off is in accord with the hypothesis. However, *c2 is on is inconsistent with the hypothesis that *c2 is off. Possibility C2B1 cannot restore consistency to the original world. Possibility C2B2 is similar to C2A and is not pursued further here.

```
(and  (*0 is on)
      (*0 is not off)
      (or   (UVK10 malfunction)                                   [A]
            (and(*i1 is not off)                                  [B]
                  (*il is on))
            (and(*m is not on)
                  (or  (and   (*m is center)
                              (*latch is center)
                              (*latch is not on)
                              (*block is pos2)
                              (*block is not pos1)
                              (*i1 is not off)
                              (*i1 is on))                        [C1]
                       (and   (*m is off)
                              (*latch is off)
                              (*latch is not on)
                              (or(block malfunction)              [C2A]
                                  (and    (*f_latch is off)
                                          (*f_latch is not on)
                                          (*f is not on)
                                          (*f is off)
                                          (UVK10 malfunction))))   )[C2A2]
```

Figure 5. Full Answer to the Query, What could explain *c1 = off, *c2 = off, and *0 = on, where at most one component might malfunction?

Wif generates solutions in a depth-first fashion and returns the solution as a tree of alterations. Any traversal from root to leaf represents a single set of alterations that restores consistency to the original world. The full answer to the original query is given in figure 5.

Wif as a Tool of General Application

Wif has no control or structure specific to fault diagnosis; Wif is a language for hypothetical reasoning. Hypothetical reasoning is a general inference technique that can be applied to a multitude of problems. Scheckler (1990) used Wif to model the reaction of the heart to the introduction of drugs by asserting causal relationships between tissue

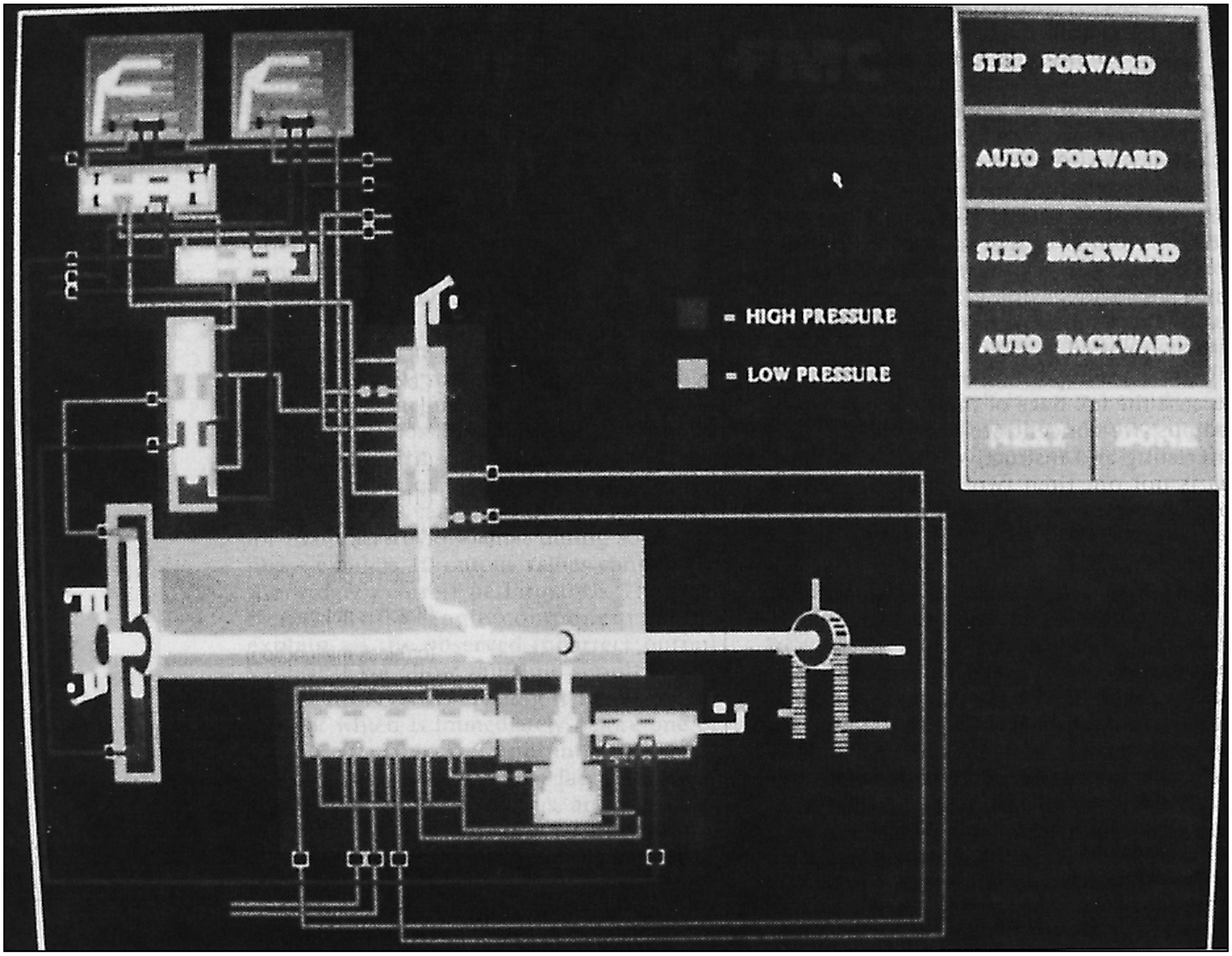


Figure 6. Graphic Display of the Lower Hoist, as Specified by Hoist.
Animated graphics were possible because the system computes the internal states of the lower hoist. Picture courtesy of FMC Corporation.

response, physiology, and clinical effects. Graham's (1986) multiple robot domain used Wif to rectify truth when an individual robot discovers that its knowledge base is incomplete or inaccurate (this phenomenon is sometimes referred to as *belief revision* or *reason maintenance*). Diverse application suggests generality of approach.

Results

A full model of the lower hoist was encoded in coherence rules, and a running simulation was produced and delivered to FMC Corporation in the fall of 1986. The model was built to act as a diagnostic expert system, and the model has been successfully tested in several diagnostic situations.

Shortly after delivery, FMC developed a graphics front end to the Hoist causal expert system (see figure 6). This step was a natural one because Hoist deduces all internal states of the lower hoist for a possible diagnostic solution (that is, all things true in the world) or, for that matter, all internal states as the lower hoist properly functions. A graphics front end simply displays a unique icon for each possible state of each component, preferably one that looks like the component in the state, as in Steamer (Hollan, Hutchins, and Weitzman 1984). The system can be used as a training device and interactive reference material as well as for fault diagnosis. The Hoist causal reasoning expert system has successfully developed from an interesting theory to a useful tool for industry.

Conclusions

First-generation expert systems have been based on compiled knowledge gleaned from human experts. A more fundamental approach would use functional models to automatically generate advice. A methodology is needed to specify the nature of individual components and their interrelationships. Causal reasoning is emerging as a technology for just this purpose: modeling qualitative physics.

The advantages of this approach over conventional expert systems stem from the nature of causal reasoning. The principle of locality ensures locality of modification and the ability to uniformly model many different complex mechanisms by reducing them to interconnecting simple relationships. From the knowledge engineer's perspective, this approach enhances system maintainability and allows development with an expert of device behavior, without the need for an expert of device malfunction usually found in diagnostic expert system projects.

Building a causal model of the lower hoist required us to tackle qualitative physics. Using a hypothetical reasoning language called Wif, we modeled the functionality of the mechanical, hydraulic, and electric systems of the lower hoist of a turret gun. One year after its genesis in the spring of 1986, Wif had spawned work in a number of different domains: causal-based fault diagnosis, robot world belief revision, and qualitative heart simulation. Hypothetical reasoning is a major facet of human intelligence and is not tied to any specific application area.

Acknowledgments

This research was supported in part by grants from the Virginia Center for Innovative Technology, Texas Instruments, and the Naval Systems Division of FMC Corporation.

References

Bachant, J., and McDermott, J. 1984. R1 Revisited: Four Years in the Trenches. *AI Magazine* 5(3): 21–32.

Clancey, W. J. 1985. Heuristic Classification. *Artificial Intelligence* 27:289–350.

Davis, R. 1984. Diagnostic Reasoning Based on Structure and Behavior. *Artificial Intelligence* 24:347–410.

DeKleer, J., and Brown, J. S. 1984. A Qualitative Physics Based on Confluences. *Artificial Intelligence* 24:7–83.

Forbus, K. D., and Brown, J. S. 1984. Qualitative Process Theory. *Artificial Intelligence* 24:85–168.

Graham, R. 1986. A Simulation for Multi-Agent Robot Problem Solving. B.S. honors thesis, Dept. of Computer Science, Virginia Polytechnic Institute and State University.

Hollan, J.; Hutchins, E.; and Weitzman, L. 1984. Steamer: An Interactive Inspectable Simulation-Based Training System. *AI Magazine* 5(2): 15–27.

Lewis, D. 1973. *Counterfactuals.* Cambridge, Mass.: Harvard University Press.

McCarthy, J., and Hayes, P. 1969. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In *Machine Intelligence* 4, eds. B. Meltzer and D. Michie, 463–503. New York: Halsted.

Rescher, N. 1964. *Hypothetical Reasoning.* Amsterdam: North-Holland.

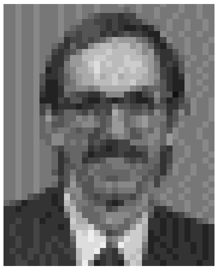
Roach, J. W.; Eichelman, F.; and Whitehead, J. D. 1985. A Coherence Logic for Counterfactual Reasoning, Dept. of Computer Science, Virginia Polytechnic Institute and State University.

Scheckler, R. 1990. M.S. thesis, Dept. of Computer Science, Virginia Polytechnic Institute and State University. Forthcoming.

Simon, H. A. 1953. Causal Ordering and Identifiability. In *Studies in Econometric Method,* W. C. Hood and T. C. Koopmans, 39–74. New Haven, Conn.: Yale University Press.

Simon, H. A., and Rescher, N. 1966. Cause and Counterfactual. *Philosophy of Science* 33:323–340.

J. Douglass H. Whitehead is a member of the technical staff of the Intelligent Systems Laboratory at Contel Technology Center, Chantilly, Virginia. He is pursuing a Ph.D. at Virginia Polytechnic Institute and State University. In 1984 and 1985, he was a knowledge engineer for FMC Corporation, where he acquired the application domain for this article. He is currently seeking to apply and extend the work published here to diagnose failures in telecommunications equipment.



John W. Roach is an associate professor of computer science at Virginia Polytechnic Institute and State University. He received his Ph.D. from the University of Texas at Austin. His research interests include counterfactual reasoning, logic programming, natural language processing, and planning.