Integrating Deep Reasoning and Compiled Reasoning for the Resolution of Interacting Malfunctions in Chemical Process Diagnosis.

James K. McDowell and James F. Davis

Department of Chemical Engineering
and Laboratory for AI Research
The Ohio State University
140 West Nineteenth Avenue
Columbus, Ohio 43210

Accepted for Presentation at QR '91
Fifth International Workshop on Qualitative Reasoning.
Austin, TX. May 19-May 22, 1991

Integrating Deep Reasoning and Compiled Reasoning
for the Resolution of Interacting Malfunctions
in Chemical Process Diagnosis.

James K. McDowell and James F. Davis

Department of Chemical Engineering and Laboratory for AI Research
The Ohio State University

ABSTRACT

Compiled reasoning systems make commitments to specific knowledge representations and inference strategies for efficient and effective problem solving. The diagnostic robustness of such systems has often been thought to depend only on the experience of domain experts, with diagnostic robustness guaranteed only by direct manipulation of deep knowledge. Our evolving view of diagnostic knowledge-based systems holds that robustness and compilation are orthogonal issues. In many circumstances, so called deep knowledge can be effectively organized into a compiled problem solver without the loss of diagnostic robustness.

In exploring the integration of compiled and deep reasoning, questions arise as to what deep process behaviors should be captured in a problem solving architecture and what commitments a problem solving architecture makes in terms of its ability to organize process behaviors. These questions are important to integration for preserving the advantages of each approach.

For diagnosis in the chemical process domain, the forms of compilation that have shown to be the most effective are hierarchical classification (HC) and structured pattern matching (SPM). In HC, a chemical process is decomposed by function-subfunction, producing a hierarchy of malfunction hypotheses. At the tip level, specific process or equipment malfunctions are represented. The remaining nodes in the hierarchy represent malfunction abstractions. Efficient search of this problem space may be achieved by rejecting abstractions high in the hierarchy, thus pruning the nodes below. The compilation procedure in SPM organizes knowledge for the evaluation of these malfunction hypotheses. In SPM, specific symptomatic information is considered in the context of a malfunction hypothesis and mapped into a hypothesis confidence rating.

These frameworks have been useful in the building of knowledge-based systems for the isolation and identification of single malfunctions and independent multiple malfunctions. Cases of interacting malfunctions may require cause-and-effect reasoning about process variables across the process topology. Process systems may interact due to stream integration, control loops and sink-source relations. The resolution of interacting malfunctions is a situation where deep reasoning can be useful and presents an opportunity for integration with compiled reasoning.

In the integration framework, the compiled problem solver, defined in terms of HC and SPM, serves the primary role and the deep reasoner is auxiliary, used only in a case specific manner for the resolution of interacting malfunctions. The compiled problem solver supplies specific diagnostic goals for the deep reasoner as well as a run-time specific assessment of the chemical process. This assessment includes process and equipment malfunctions as well as the identification of properly functioning subsystems. This guidance from the compiled problem solver aids the deep reasoner in what would normally be a computationally expensive procedure of cause-and-effect reasoning.

INTRODUCTION

Task specific architectures play an important role in the construction of diagnostic knowledge-based systems. These task specific architectures express an effective methodology for both robust and efficient run-time problem-solving. Successful systems have been constructed in the chemical process, manufacturing and electronic domains for the diagnosis of single and independent multiple malfunctions using task specific architectures [Shum et al., 1988; Shum et al., 1989; Ramesh et al., 1988; Ramesh et al., 1989; Myers et al. 1989; Myers et al. 1990; McDowell et al. 1990]. However, it is well known that process and equipment malfunctions can interact via stream integration, control loops and sink-source relationships. These causally interacting malfunctions represent a relatively unexplored but very important application area for diagnostic knowledge-based systems. Causally interacting malfunctions include several possible scenarios. One might involve an equipment malfunction that causes an additional malfunction in a far removed part of the system. Another scenario involves a malfunction interfering with the correct operation of a seemingly unrelated part of the process. The resolution of such multiple interacting malfunctions requires the consideration of system structure and system behaviors in the context of the malfunction scenario. This type of knowledge is commonly referred to as deep knowledge, because the representation primitives and inference procedures attempt to model the behavior of a physical system [Bylander, 1990].

This paper presents a framework called Diagnostically Focused Simulation (DFS). The emphasis is on integrating deep knowledge with other forms of problem solving to fully resolve interacting multiple malfunctions. Specifically, the focus is on the integration of deep reasoning into the generic task knowledge-based system framework. The presentation is organized into four sections. The first section closely examines our theory of generic tasks for diagnosis in the process domain. The focus is the process of compilation associated with generic tasks and the role of deep knowledge in the construction of diagnostic compiled problem solvers based on this paradigm.

Given the potential need to directly manipulate deep knowledge in cases of potential interacting malfunctions, the next section reviews and classifies approaches that explicitly use deep knowledge in or as a basis for knowledge-based diagnosis. A common theme is that unconstrained reasoning with deep knowledge can be computationally expensive. This computational problem is partially remedied in these reported approaches through the integration of other kinds of problem solving.

With this foundation in place, the DFS framework for integrating deep reasoning into the generic task theory is described. This conceptual framework is defined in terms of knowledge and the inference that are used in the problem-solving to resolve specific cases of interacting multiple malfunctions.

As noted with other approaches that use deep knowledge, the deep reasoning aspect of the DFS task is potentially computationally explosive. To avoid this problem, the reasoning process in DFS is constrained at several levels:

1) DFS is an auxiliary problem solver and as such is only invoked in specific cases. The primary activity of diagnosis (isolation and identification of malfunctions) is done by the compiled problem solver.

2) The diagnostic results of the compiled problem solver will suggest specific types of interactions between multiple malfunctions. These types of interactions suggest a specific simulation agenda with certain expectations concerning the simulation results.

3) In addition to isolating possible interacting malfunctions, the compiled problem solver provides an assessment of the functional subsystems that are operating correctly. This assessment allows DFS to construct a multi-level view of the process, specific to the diagnostic case and suitable for cause-and-effect reasoning.

DFS attempts to reenact malfunction interactions by simulating a local malfunction, propagating its effect on the system behavior using qualitative simulation and then evaluating any global effects on other malfunction hypotheses. DFS represents a novel approach to integrating deep reasoning and compiled reasoning. The integration is tightly woven and brings together important elements of both types of reasoning. The paper concludes with a summary and some remarks concerning the integration of different types of problem solving.

TASK APPROACH TO DIAGNOSTIC KNOWLEDGE-BASED SYSTEMS

A distributed problem solving framework has been formulated for on-line malfunction diagnosis of complex manufacturing and process operations [Shum et al., 1989]. This approach to knowledge-based systems, known as the generic task approach, was originally proposed by Chandrasekaran [Chandrasekaran, 1983; Chandrasekaran, 1986; Chandrasekaran, 1987]. Each generic task, makes a specific commitment to knowledge structures and inference processes. This forms a basis for capturing domain knowledge and organizing it into an effective problem solving framework. The generic task theory has been successfully applied to building knowledge-based diagnostic systems in several engineering domains. Successful efforts in the chemical process domain include a terephthalic acid process and a fluidized catalytic cracking unit [Shum et al., 1988; Ramesh et al., 1988 and Ramesh et al., 1989]. A fielded system applying this methodology to the automated diagnosis of discrete PLC processes is demonstrated in the area of manufacturing [Myers, et al., 1989 and Myers, et al., 1990]. This theory of knowledge-based systems has also been applied to the diagnosis of electronic devices [McDowell, et al., 1990].

COMPILATION IN TASK SPECIFIC ARCHITECTURES

In these engineering domains, compiled diagnostic problem solvers have been successfully constructed using the paradigms of hierarchical classification (HC) and structured pattern matching (SPM) Compilation in the generic task theory, refers to a commitment to problem solving architectures without any reference to the source of knowledge. The type of problem solver serves as a template for organizing knowledge. The literature has often used the term compiled interchangeably with terms like heuristic and shallow [Venkatasubramanian and Rich, 1988; Fink and Lusth, 1987]. Our view, however, is that the terms heuristic and shallow imply a commitment to a source of knowledge and is very different from compilation. A problem solving framework might be encoded with knowledge from several sources: deep behavioral understanding; expert experience; process history. Given this viewpoint the issues of knowledge source and run-time problem solving seem to be orthogonal [McDowell et al., 1989]. Diagnostic knowledge-based systems can be viewed along two dimensions. The first involves the run-time behavior of the problem solver in terms of its use of simulation or compiled structures. The second dimension involves the source of knowledge used as a basis for these compilation and/or simulation structures. The notions of knowledge-source and run-time behavior enhance the characterization of various knowledge-based diagnostic efforts. In the course of examining task architectures with respect to deep knowledge, the features of these compilation commitments are closely examined.

A key element in the construction of the hierarchical knowledge structure used in HC, involves the identification of functional systems and subsystems in the process or device being diagnosed. This process or device could be a chemical process, a manufacturing process or an electronic device. Figure 1 has been adapted from a process example given by Venkatasubramanian and Rich [Venkatasubramanian and Rich, 1988]. This chemical process is composed of several common process units: pumps, valves, piping, a reactor, heat exchanger, etc. The procedure of identifying functional systems/subsystems decomposes the chemical process into a malfunction hierarchy. The first level of decomposition views the chemical process as being composed of the following major functional systems: Feed System, Reactor System and Product System. The procedure is continued by functionally decomposing each of these functional systems, for example the Reactor System decomposes into: Level Control System, Reactor, Temperature Control System. The ultimate result of this procedure is the functional hierarchy shown in Figure 2. The nodes in this hierarchy represent malfunction hypotheses. The tip-level nodes represent the process units themselves and the intermediate nodes represent malfunction abstractions. As demonstrated, the procedure of constructing the hierarchy is one of trading process connections for connections in a problem solving framework. At the completion of the procedure the process structure is no longer explicit. The purpose of this compilation is to organize knowledge into a useful form for a particular task. The compilation procedure for HC organizes the process behavior in terms of connections in a malfunction hypothesis hierarchy. Diagnostic search operates on this problem solving structure and not on the chemical process connectivities.

As shown in the generation of Figure 2, HC attempts to organize the process malfunctions into a problem solving framework. The nodes of the hierarchy then represent malfunction hypotheses of successive levels of detail. Such a problem solving framework offers an advantage during diagnosis. If a malfunction hypothesis high in the hierarchy can be rejected, then consideration of malfunction hypotheses below this node is not necessary. This inference procedure prunes the search space of malfunctions and only pursues the relevant branches of the hierarchy, until a tip level node or nodes are established. A great deal of focusing can be generated in this way.

Each node in the malfunction hierarchy represents a malfunction hypothesis. The evaluation of such a hypothesis requires different knowledge and inference procedures and represents a different generic task than that of searching the hierarchy itself. The generic task used in evaluating malfunction hypotheses is called structured pattern matching (SPM). In this task, specific symptomatic features are accessed and matched against predefined patterns. Each pattern has associated with it a score or rating called a confidence value. Common in all diagnostic hypotheses are the values "confirmed" and "ruled-out". When symptomatic information matches a pattern of features the appropriate confidence value is assigned to the malfunction hypothesis. An example of the knowledge organization for this task is shown in Figure 3. Each node in the malfunction hierarchy can have associated with it one or several such pattern matchers and the matchers can be organized hierarchically. Compilation of this form eliminates the need for generating symptom patterns at run-time.

Another form of compilation relevant to HC and SPM involves the elimination of unnecessary causal links. If there exists a situation in which a malfunction causes a chain of symptoms to arise for example: A(malfunction)--> B(symptom)--> C(symptom)--> D(symptom). If knowledge ensures that this causal path always occurs then tracing the intermediate causal links is not necessary. The elimination of such causal links is useful in knowledge organization and selection for both HC and SPM.

Our past efforts in building knowledge-based diagnostic systems have demonstrated that for processes or devices where a complete understanding of the diagnostic situation is available, the constructed diagnostic systems are completely robust. By robustness we mean that the diagnostic system was always able to arrive at the correct diagnostic conclusion. These situations involved single and independent multiple malfunctions in processes where the certainty in selecting symptoms for establishing malfunctions were very high. In the cases of multiple interdependent malfunctions, HC is able to robustly identify potential interacting hypotheses, but is not able to bring the situation to complete resolution by identifying which malfunction hypothesis is the genuine root cause. Such situations make the manipulation of deep knowledge a potential part of the problem solving in the overall diagnostic activity.

APPROACHES FOR INTEGRATING DEEP REASONING

There are several approaches to using deep knowledge and reasoning in diagnostic problem solving. These approaches can be categorized as either **augmented** or **transformed** approaches. **Augmented** approaches strongly separate the run-time use of deep knowledge from other types of problem solving knowledge. **Transformed** approaches represent efforts that use deep knowledge and reasoning to generate malfunction states that are then captured into a new problem solving architecture for diagnosis. Each approach will be discussed in terms of the modeling technique for deep knowledge, how the deep knowledge is used and finally the form and content of its compiled component. The compiled component of these approaches should not be confused with task-specific compiled structures discuss previously. The compiled elements of the these integration approaches is compared with the task-specific architectures at the close of this section.

Representative of the augmented approaches is MODEX2 [Venkatasubramanian and Rich, 1988] and Grantham and Ungars' First-Principles-Approach (referred to here as FPA) [Grantham and Ungar, 1989]. MIMIC [Dvorak and Kuipers, 1989] and MIDAS [Oyeleye et al., 1989; Finch and Kramer, 1989] are representative of the transformed approaches to integrating deep knowledge.

AUGMENTED INTEGRATION

Augmented approaches make a commitment to using deep knowledge directly at run-time during diagnosis. This problem solving involves causally searching through the deep model with the objective of generating valid paths from symptoms to root cause malfunctions. To avoid this potentially computationally costly manipulation of deep knowledge, the approaches are augmented with other forms of problem solving that serve to guide or avoid the use of deep knowledge.

In MODEX2, process structure is represented as components (process units) and connections (streams). Deep knowledge about process behavior is represented using qualitative constraints and confluences [deKleer and Brown, 1984]. The compiled knowledge is in the form of associations, mapping symptoms directly to root causes. If the specific symptom to malfunction preenumeration exists in the knowledge-base then the malfunction can be directly isolated without using deep knowledge. Unfortunately, knowledge in the form of direct associations, though efficient, is rarely complete and very difficult to organize. Thus MODEX2 often must resort to the use of deep knowledge which is applied in a strategy separate from the compiled problem solving component.

FPA uses Forbus' process ontology [Forbus, 1984] to model the system structure and behavior. Diagnosis is composed of a generation and a test phase. Generation involves tracing from symptom to cause in the qualitative physics model and proposing changes in the model that would account for the symptoms. The testing phase involves modifying the model and executing a form of inference called qualitative simulation to see if the model modification indeed accounts for the symptoms. Qualitative simulation is a predictive form of reasoning done on deep model representations to produce process behaviors. In FPA, both the generation and the testing phases are very inefficient and produce multiple candidates and multiple simulation results. To control this multiplicity, FPA augments the deep reasoning with a compiled problem solving component composed of strategies for ordering, choosing and testing malfunction candidates.

TRANSFORMED INTEGRATION

This approach to integration uses deep knowledge as the sole source of knowledge for their compiled diagnostic problem solvers. Malfunction scenarios are enacted from deep knowledge using qualitative simulations. The simulation results are organized into a problem solving framework that is used for diagnosis. The transformed approaches avoid directly manipulating the deep knowledge at run-time and instead make use of a more efficient compiled representation for diagnostic problem solving. The compiled component of the transformed approaches is very different from those seen in the augmented or task- specific approaches. The transformed approaches' compiled component is drawn directly from deep knowledge and replaces rather than augments the direct manipulation of deep knowledge. The compiled components in the augmented approaches do not have a formal basis in the deep representation and are derived independent of the deep knowledge.

MIMIC uses Kuipers' constraint ontology [Kuipers, 1986] to model the process. Qualitative simulations are run for each malfunction scenario. The results are then processed using structured induction, an automated technique for feature classification. The product of structured induction is a decision tree, which forms the compiled problem solver in MIMIC. The tip nodes of the decision tree are the malfunctions and the intermediate nodes are single symptoms. Connections in the decision tree represent possible values of the symptoms. Symptoms are requested one at a time and depending on their values the decision tree is traversed in pursuit of the malfunctions.

MIDAS uses Extended Signed Directed Graphs (ESDG) [Kramer and Oyeleye, 1988] to model the behavior of process units. The process model is then transformed into a causal net structure, called an event network, which serves as MIDAS' compiled representation for diagnostic problem solving. This network links malfunctions to a chain of intermediate symptoms and measurable symptoms. These preenumerated causal paths are used for diagnosis, instead of generating them at run-time.

COMMON FEATURES AMONG INTEGRATION APPROACHES

In summary, though all the approaches above use deep knowledge explicitly, their choice of representation in each case is different. First, there is no unique representation for expressing knowledge about process structure and behavior. Secondly, all the approaches implicitly support the notion that unconstrained inference with deep knowledge is computationally explosive. All of these approaches integrate compiled problem solving in some manner (augmented or transformed) with the deep knowledge and reasoning for the specific purpose of controlling computational expense.

DIFFERENCES IN COMPILATION APPROACHES

As previously discussed, the transformed approaches to integration, do not make use of deep knowledge during the process of diagnosis, but use compiled problem solvers that differ from the task architectures. MIMIC's decision tree makes a commitment to feature classification that may be used for diagnostic problem solving. There is however, no explicit system/subsystem decomposition or malfunction hypothesis evaluation. The intermediate nodes in the decision tree relate only to features(symptoms) and are not intermediate malfunction abstractions. Only the tip nodes of the decision tree represent the malfunctioning system units as seen in the functional malfunction hierarchy used in HC. MIDAS's causal net structure does not make explicit any system/subsystem decomposition or malfunction hypothesis evaluation. Nodes in the causal net are either root cause malfunctions or symptoms that arise due to the malfunction. An additional difference in task compilation is that sources other than deep knowledge can be used in the problem solving architecture. From the earlier discussion the transformed approaches make a specific commitment to their deep knowledge as the sole source for the diagnostic problem solvers.

The compiled problem solving component of the augmented approaches do not have the organization of the transformed or task approaches to compilation. This is because they rely on the manipulation of deep knowledge for most of the diagnostic problem solving. The compiled problem solving component serves as a diagnostic shortcut or provides rules of thumb for constraining the deep reasoning. Recall MODEX2 uses compiled knowledge in the form of associations that map directly from symptoms to malfunction. FPA's compiled component is committed to constraining the manipulation of deep knowledge by controlling the generation of diagnostic candidates and providing ordering for candidate testing.

This is not to say any of these problem solving systems would lack in robustness, since robustness depends on the source of knowledge. It does point out that there are different ways to use deep knowledge for the purpose of diagnosis, both in a compiled form and directly at run-time.

TASK SPECIFIC COMPILATION WITH DEEP REASONING

Task specific problem solvers using HC/SPM in the process domain tend to represent systems with a high level of decomposition. The abstract malfunction hypotheses are evaluated in a local manner, drawing on knowledge only related to confirming or ruling-out that specific hypothesis. These compiled systems are effective in identifying single malfunctions and independent multiple malfunctions. This class of malfunctions can be organized in terms of a functional decomposition and local consideration of symptomatic information. Based on our observations to date, single and independent multiple malfunctions cover the majority of scenarios in the chemical process plant domain. The commitments of compilation do not affect the robustness of the diagnostic system for these malfunction scenarios.

With respect to the class of all malfunctions there is the possibility of causally related multiple malfunctions. Interacting situations are possible via stream integration, control loops, and sink-source relations. Thus multiple-interacting malfunctions are a possibility and a concern in the chemical engineering domain.

The resolution of multiple interacting malfunctions requires the consideration of the system topology and the relationships among process variables in the context of malfunction hypotheses. This would suggest that the direct manipulation of deep knowledge would be necessary in such a case. Though all the systems previously discussed use deep knowledge, only MIDAS considers the area of causally interacting malfunctions, and then in the narrow area of induced sensor failure. The remaining systems, MIMIC, MODEX2 and FPA are committed to single malfunctions and independent multiple malfunctions. They may be able to identify cases of interacting multiple malfunctions, however it would be difficult to distinguish from the case of independent multiple malfunctions, unless the interaction was preenumerated.

The focus of this work is to significantly build upon the task specific approach in the area of causally interacting multiple malfunctions. The intent is to leverage information contained in the compiled structures of HC but is not found in the augmented or transformed structures of the other approaches. Namely, we will show that the functional decomposition contains the information to construct the case-specific system structures along which multiple malfunctions may interact. From a generic task view of problem solving, this requires an additional auxiliary problem solver for manipulating deep knowledge. As part of the task architecture, Diagnostically Focused Simulation (DFS) can be viewed as an auxiliary information processing task that involves reasoning about cause-and-effect among process variables across a process topology for the specific purpose of resolving uncertain diagnostic conclusions relating to multiple interacting malfunctions.

THE DETAILS OF THE INFORMATION PROCESSING IN DFS

Because DFS is an auxiliary problem solver, its processing is invoked after the compiled problem solver completes its diagnostic assessment. Figure 4 illustrates the inference procedure which is used by the compiled diagnostic problem solver. The top node in the hierarchy is made active and the problem-solving procedure in Figure 4 begins. Specific symptomatic information is examined in the context of evaluating a selected malfunction hypothesis. If the hypothesis is established (this usually requires a confidence value score of "confirmed" or "very-likely"), then its children in the hierarchy are made available for possible evaluation. When a malfunction is rejected, the portion of the hierarchy below the hypothesis is effectively pruned by the inference procedure. One or several tip level node malfunctions are established in the normal procedure of diagnosis. The results of the initial diagnostic assessment might look like the confidence value hierarchy provided in Figure 5.

OVERVIEW OF KNOWLEDGE AND INFERENCE IN DFS

As part of the task architecture, DFS will make use of specific types of knowledge during problem solving. First, if DFS is going to reason about malfunctions, some form of knowledge concerning fault models must be available. Component fault models provide knowledge about how specific malfunctions affect local process variables. Second, knowledge about process structure is needed. For example, in the chemical process domain, knowledge about process structure is usually found in the process flowsheet or piping and instrumentation diagrams. Knowledge about behavior must also be available at the system, subsystem and component level. Knowledge about structure and behavior will make it possible to reason about the cause-and-effect behavior across the system. Each of these forms of knowledge were used in one or more of the integration approaches discussed previously.

However, DFS requires an additional type of knowledge not found in the other approaches. This additional requirement relates to organizational knowledge links from the compiled problem solving architecture to the system structure. Hypotheses in the compiled problem solver map into relevant corresponding portions of the system. This knowledge serves to focus the causal reasoning on situation specific portions of the process.

A flowchart of the DFS inference procedure that operates on the knowledge representations described above is shown in Figure 6. The diagnostic results are composed of the portion of the malfunction hierarchy explored during diagnosis, each malfunction hypothesis considered is given a confidence value score (recall Figure 5). This map is searched for certain confidence value patterns. These patterns identify specific situations of interaction and define a simulation agenda. Additionally the map of hypotheses is transformed into a Functionally Decomposed Structural Abstraction (FDSA) of the system topology. The FDSA is a representation of the system structure at multiple levels of detail, appropriate for the diagnostic situation and is constructed from the organizational knowledge previously described. The FDSA focuses in detail on the malfunctions and maintains higher level abstractions for other portions of the system. According to the diagnostic goals, the

FDSA is used to establish possible conduits for malfunction interactions. If there are no conduits, then simulation of the malfunction effects is not needed. If a conduit does exist, then candidate malfunctions are considered by propagating their local effects across the FDSA using qualitative simulation. The global effects on other malfunction hypotheses are evaluated by structured pattern matching on the simulation results. In essence the qualitative simulation attempts to reenact the possible malfunction interactions taking place in the chemical process system.

CONFIDENCE VALUE PATTERNS

Possible malfunction interaction situations are identified by analyzing the results of the compiled diagnostic problem solver. As stated earlier the results are in the form of a confidence value hierarchy like that of Figure 5. Specific patterns of confidence values in the confidence value hierarchy suggest possible malfunction interactions. Identifying the type of interaction specifies the diagnostic goals for using qualitative simulation and defines a simulation agenda for DFS. The confidence value hierarchy makes it possible to identify possible malfunction interactions.

The current study has been restricted to two confidence value patterns that have been defined. The first pattern is shown in Figure 7 relates the scenario where involving an equipment malfunction that causes an additional malfunction in a far removed part of the chemical process. The malfunctions represented by hypotheses C, D, E, F and G are tip nodes. These nodes represent specific equipment malfunctions or incorrect process settings. Their respective parents, A and B are not siblings, implying they are in different functional sections of the process. Hypotheses D and F have been confirmed as malfunctions and hypotheses C, E, and G have been ruled-out. The question concerning this pattern is whether malfunctions D and F are causally related. The possible paths of interaction between D and F are not clearly represented by the hierarchy. The possible questions are: Does D cause F? Does F cause D? Are they causally independent? The simulation agenda for DFS in this interaction case would include two simulations: 1) cause malfunction D and determine if it affects F. 2) cause malfunction F and determine if it affects D.

A second confidence value pattern is shown in Figure 8. This scenario involves a malfunction interfering with the correct operation of a seemingly unrelated section of the chemical process. Only a single malfunction has been confirmed (node F). A cluster of tip nodes has been ruled-out, all of which are children of hypothesis A. With their immediate parent being confirmed (A), it is expected that one of the child tip nodes (C, D, or E) would be confirmed and not ruled-out. There are several possible reasons for this pattern. The one of interest is a case of a secondary malfunction. The confirmed malfunction F is causing the parent malfunction abstraction A, to be confirmed even though it is not part of the functional decomposition of A. F is able to influence A across the process topology without being a child of A. A secondary causal relationship may exist between F and A. A strict functional decomposition would not expose this relationship. The goal of the DFS is to identify this secondary causal relationship. The simulation agenda involves a single simulation: cause malfunction F and determine if it affects A.

FUNCTIONALLY DECOMPOSED STRUCTURAL ABSTRACTIONS

Exploring possible interactions among malfunction hypotheses requires the consideration of system structure and behavior like that illustrated in Figure 1. Figure 1 represents the chemical process at the lowest level of abstraction and the greatest level of detail required for the process of interest. However, more knowledge is available at this point in the problem solving concerning the chemical process. A diagnostic assessment in the form of the confidence value hierarchy is available. The compiled problem solver identified what process subsystems were functioning correctly as well as any malfunction(s). The subsystems that are operating correctly need not be analyzed to the detail of their process units. Instead they can be represented as a consolidated whole.

Consider the diagnostic assessment in Figure 5 of the chemical process in Figure 1. Control Valve 2 has been confirmed as a malfunction. Using organizational knowledge links from the compiled problem solver to the process structure, a case-specific view of the process is possible. This case specific view of the chemical process is called the FDSA. Figure 9 provides the FDSA of Figure 1 given the diagnostic assessment in Figure 5. Figure 9 is a view of the process at a level of detail appropriate for the diagnostic situation. When considering the interactions among multiple malfunctions, the FDSA can focus detail at the appropriate level specific to the diagnostic results in the confidence value hierarchy.

Before simulation takes place some additional problem solving can be done. For the malfunctions to interact causally, there must be a conduit of interaction between them. Before simulation proceeds, the existence of such a conduit should be established. The process topology of that in Figure 1 can be used for this task, i.e. considering the process at the greatest level of detail. The FDSA is also appropriate for this task. The candidate hypotheses are represented as well as the structural connections. Since the FDSA contains abstractions of the process it can be expected to have fewer connections and thus be easier to search. If no conduit between the malfunctions exist, then qualitative simulation can be avoided.

COMPONENT FAULT MODELS

To determine if malfunction A causes malfunction B, first it is necessary to have knowledge about the local effects of malfunction A. Local effects refer to how a particular malfunction affects local process variables. For example, a leaking pipe will cause loss of material, a stuck control valve will eliminate control actions, fouling causes a reduction in heat transfer. This knowledge must be enumerated for all the tip level hypotheses in the diagnostic hierarchy. The effect on local process variables represents the initial conditions for qualitative simulation.

QUALITATIVE SIMULATION

Once the FDSA is constructed and the relevant malfunction is identified and its effect on local process variables is enforced, the procedure becomes one of qualitative simulation. Qualitative simulation propagates the effect of the malfunction across the FDSA. For this to occur the FDSA must map into the appropriate qualitative modeling primitives. From the examination of other approaches to using deep knowledge and reasoning there are several ways to qualitatively model processes and process behaviors. The FDSA as a representation, is however, independent of modeling approach. Prior to simulation, the FDSA maps into the appropriate modeling primitives creating a qualitative model specific to the diagnostic case. The important issue is that the modeling approach must be robust enough to represent the engineering systems of interest. It must also be flexible enough to represent processes at several levels of detail.

A major concern and criticism of qualitative simulation is the generation of multiple and incorrect process behaviors [Kramer and Oyeleye, 1988]. The source of this issue is the qualitative nature of the processing. The simulation at some level is necessarily vague and imprecise. Several efforts have been devoted to dealing with this issue [Kuipers, 1986; Kuipers and Berleant, 1988; Kuipers and Chiu, 1987; Faulkenhainer and Forbus, 1988].

DFS addresses this multiplicity situation in two ways. The FDSA is a diagnostically derived process abstraction. Using the FDSA as the basis for the qualitative physics model avoids unnecessary process details that can contribute to the generation of multiple results during qualitative simulation. In this sense it parallels many of the issues described by Faulkenhainer and Forbus [1988], however their work does not address abstraction from a diagnostic foundation. Kramer and Oyeleye [1988] identified the need to locate correctly functioning portions of the process, especially control loops to reduce multiplicity. In the case of DFS, part of the compiled diagnostic problem solver's operation is to identify portions of the plant that are operating correctly as well as identifying malfunctions.

PATTERN MATCHING AND EVALUATION

Once the qualitative simulation is complete the diagnostic goals are evaluated. The evaluation involves the pattern matching of specific hypotheses from the original compiled problem solver against the results of the qualitative simulation. If any multiplicity of final results occurs, then each individual result is evaluated. The evaluation phase might involve using the SPM knowledge of a single tip hypothesis, or the SPM knowledge of the tip hypothesis and all its ancestors. The case will depend on the knowledge engineering of the SPM knowledge and relationship among related hypotheses. The results of the evaluation are dependent on the type of interaction suspected, which is derived from the preliminary diagnostic results of the compiled problem solver.

COMPARISON WITH PREVIOUS APPROACHES

DFS makes a very different commitment to integration than the augmented or transformed approaches. In a sense DFS is a synthesis of the two approaches to integration. Figure 10 demonstrates the differences in integration approaches. DFS classifies deep process knowledge into two types. The "Type 1" knowledge serves as a knowledge source for the compiled diagnostic problem solver as was seen in the transformed approaches. In the discussion of generic tasks it was pointed out that deep knowledge can be a source for HC and SPM. The "Type 2" knowledge is the deep knowledge used at run-time. Though knowledge is separated much like the two tier-approaches, the integration in DFS is tightly woven and not stratified. In DFS, the compiled problem solver has links with the deep knowledge at several levels. The results of the compiled problem solver strongly impact what deep knowledge is used in DFS. Like the transformed approaches, malfunctions are simulated, but in the case of DFS the simulations are used only on an as needed basis to resolve cases of interacting multiple malfunctions. DFS's basis in the task architecture allows this novel and tightly integrated approach to using deep knowledge.

CONCLUSIONS

This paper has described the features of the DFS problem solving task, which is designed to resolve multiple interacting malfunctions using deep knowledge. This task uses qualitative simulation in a diagnostically focused manner for the resolution of interacting multiple malfunctions. The close integration with other problem solvers represents an evolutionary approach to using qualitative simulation in diagnosis. Under ordinary conditions qualitative simulation can be a computationally explosive procedure. Because it is part of a distributed problem solving architecture, DFS uses qualitative simulation in a constrained and diagnostically specific manner. The integration of DFS with the task architecture provides several problem solving advantages:

1) DFS is an auxiliary problem solver and as such is only invoked in specific cases. The primary activity of diagnosis isolation and identification of malfunctions is efficiently and effectively done by the compiled problem solver.

2) The results of the primary compiled problem solver may suggest specific types of interactions between multiple malfunctions. These types of interactions suggest a specific simulation agenda.

3) The compiled problem solver provides an assessment of the functional subsystems that are operating correctly. This assessment allows DFS to construct the FDSA, a multi-level view of the process, specific to the diagnostic case and provides the appropriate level of process detail suitable for cause-and-effect reasoning.

4) Pre-simulation problem solving is done on the FDSA in order to establish a conduit of interaction among malfunctions. If the interaction conduit does not exist, the malfunctions are independent and qualitative simulation can be avoided.

5) Expectations concerning the simulation and potential malfunction interactions, allows DFS to effectively evaluate qualitative simulation results, even in the case of multiple behaviors.

As an integrated approach, DFS brings together multiple sources of knowledge, a situation specific interpretation of diagnostic results and a balance between the use of run-time simulation and compiled problem solving in the activity of diagnosis.

BIBLIOGRAPHY

Bylander, T., "Some Causal Models are Deeper than Others," Artificial Intelligence in Medicine, Vol. 2, No. 3, (1990).

Chandrasekaran, B., "Toward a Taxonomy of Problem Solving Types," AI Magazine, Vol. 4, No. 1, (1983).

Chandrasekaran, B., "Generic Tasks in Knowledge-based Reasoning: High-Level Building Blocks for Expert System Design," IEEE Expert, Vol. 1, No. 3, (1986) 23-30.

Chandrasekaran, B., "Towards a Functional Architecture for Intelligence Based on Generic Information Processing Tasks," Proc. of Tenth Int. J. Conf. on AI, Milan, Italy, August (1987).

deKleer, J. and J.S. Brown, "A Qualitative Physics Based on Confluences.", Artificial Intelligence, Vol. 24, (1984) 7-83.

Dvorak D. and B. Kuipers, "Model-based Monitoring of Dynamic Systems," Proc. of Eleventh Int. Joint. Conf. on AI (IJCAI-89), Detroit, MI (1989).

Grantham S. and L. Ungar, "A Qualitative Physics Approach to Troubleshooting Chemical Plants," presented at the AIChE National Meeting, San Francisco, CA, November (1989).

Faulkenhainer B. and K.D. Forbus, "The Organization of Large Scale Qualitative Models", Proc. of Nat. Conf. on AI (AAAI-88), Morgan Kaufmann, Los Altos, Cal. (1988).

Forbus, K.D., "Qualitative Process Theory.", Artificial Intelligence, Vol. 24, (1984) 85-168.

Finch, F.E. and M.A. Kramer, "The Handling of Dynamics, Multiple Faults, and Out-of-Order Alarms.", presented at the Spring AIChE National Meeting, Houston, TX, April, 1989.

Fink, P.K. and J.C. Lusth, "Expert Systems and Diagnostic Expertise in the Mechanical and Electrical Domains.", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-17, No. 3, (1987) 340-349.

Kramer, M. and O.O. Oyeleye, "Qualitative Simulation in Chemical Engineering: Steady State Analysis.", AIChE J., Vol. 34, No. 9 (1988) 1441-1454.

Kuipers, B., "Qualitative Simulation.", Artificial Intelligence, Vol. 29, (1986) 289-338.

Kuipers, B. and D. Berleant, "Using Incomplete Quantitative Knowledge in Qualitative Reasoning.", Proc. Nat. Conf. on AI (AAAI-88). Morgan Kaufmann, Los Altos, Cal. (1988).

Kuipers, B. and C. Chiu, "Taming Intractable Branching in Qualitative Simulation.", Proc. Tenth Int. Joint Conf. on AI (IJCAI-87). Morgan Kaufmann, Los Altos, Cal. (1987).

McDowell, J.K., D.R. Myers and J.F. Davis, "A Changing Perspective on Model-based versus Compiled Knowledge Systems: An Examination in Terms of a Fielded Expert System," Workshops on AI in Manufacturing, Int. Joint. Conf. on AI (IJCAI-89), Detroit, MI (1989).

McDowell, J.K., J.F. Davis, M.A. Abd-Allah and M. Aref, "A Comparison of Compiled Reasoning Systems and Model-based Systems and Their Applicability to the Diagnosis of Avionics Systems," Proc. of the IEEE 1990 National Aerospace and Electronics Conference NAECON 90, Dayton, Ohio (1990).

Myers, D.R., J.F. Davis, and C.E. Hurley, "An Expert System for Diagnosis of a Sequential, PLC-Controlled Operation," appears in Artificial Intelligence Applications in Process Engineering, Editor: Michael Mavrovouniotis, Academic Press, (1990)

Myers, D.R., J.F. Davis, and C.E. Hurley, "Application of Artificial Intelligence to Malfunction Diagnosis of Sequential Operations Which Involve Programmable Logic Controllers," presented at the Spring AIChE National Meeting, Houston, TX, April (1989).

Oyeleye, O.O., F.E. Finch and M.A. Kramer, "Qualitative Modeling and Fault Diagnosis of Dynamic Processes by MIDAS," presented at the AIChE National Meeting, San Francisco, CA, November (1989).

Ramesh, T.S., S.K. Shum and J.F. Davis, "A Structured Framework for Efficient Problem Solving in Diagnostic Expert Systems." Comput. Chem. Engng. Vol. 12, No. 9/10 (1988) pp. 891-902.

Ramesh, T.S., J.F. Davis and G.M. Schwenzer, "CATCRACKER: An Expert System for Process and Malfunction Diagnosis in Fluid Catalytic Cracking Units," presented at the AIChE National Meeting, San Francisco, CA, November (1989).

Shum S.K., J.F. Davis, W.F. Punch III and B. Chandrasekaran. "An Expert System Approach to Malfunction Diagnosis in Chemical Plants." Comput. Chem. Engng., Vol. 12, No. 1, (1988) pp. 27-36.

Shum S.K., M.S. Gandikota, T.S. Ramesh. J.K. McDowell, D.R. Myers, J.R. Whiteley and J.F. Davis, "A Knowledge-Based System Framework for Diagnosis in Process Plants." in Conference Proceedings of The 7th Power Plant Dynamics, Control and Testing Symposium, (1989).

Venkatasubramanian, V. and S. H. Rich, "An Object-oriented Two-tier Architecture for Integrating Compiled and Deep-level Knowledge for Process Diagnosis." Comput. Chem. Engng., Vol. 12, No. 9/10 (1988) 903-922.

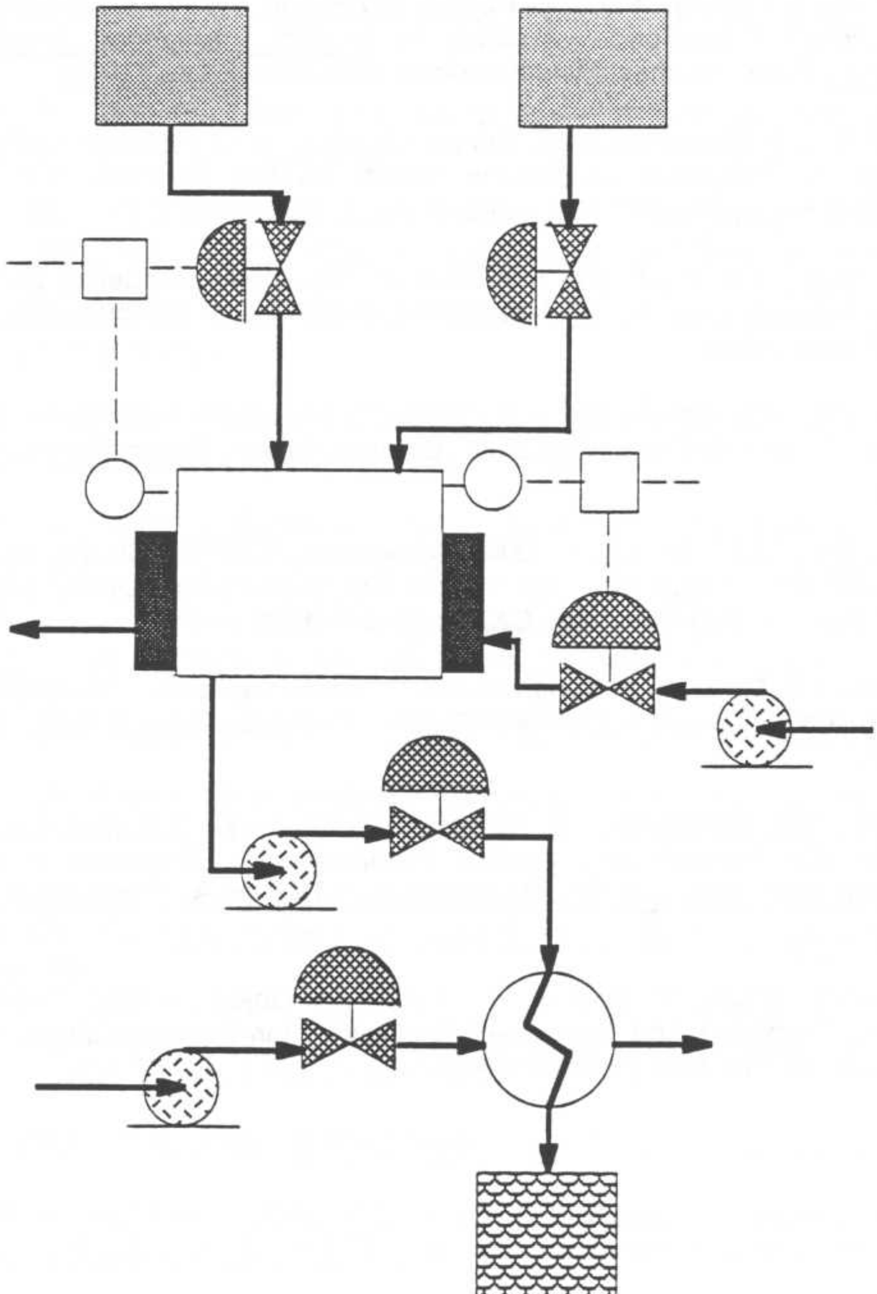

Figure 1. Chemical Process.

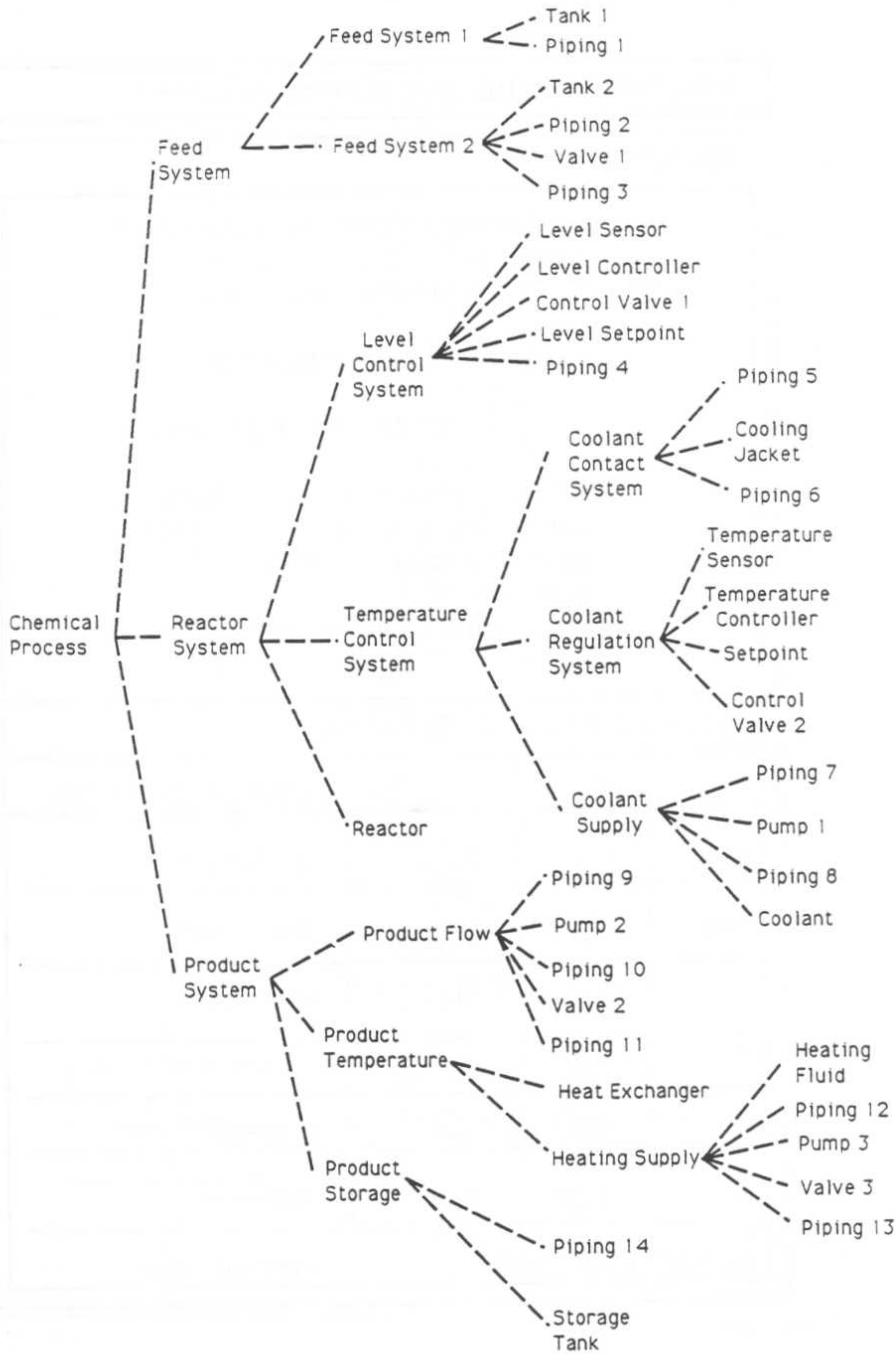

Figure 2. Malfunction Hierarchy for a Chemical Process.

HYPOTHESIS: Malfunction in Pressure Control

FEATURES:

F1: Pressure alarm above condenser activated?
values: (yes, no, uncertain)

F2: Pressure measurement above condenser?
values: (normal, high, low)

F3: How far open is the pressure control valve above the condenser as indicated by the valve positioner?
values:(normal, high, low)

HYPOTHESIS CONFIDENCE RATING:

F1	F2	F3	CONFIDENCE RATING
yes	low	high	confirmed
yes	high	low	confirmed
?	low	high	very-likely
?	high	low	very-likely
?	low	low	rule-out
?	high	high	rule-out
NO MATCH PATTERN			very-unlikely

Figure 3. Pattern Matching Table for Hypothesis Evaluation.

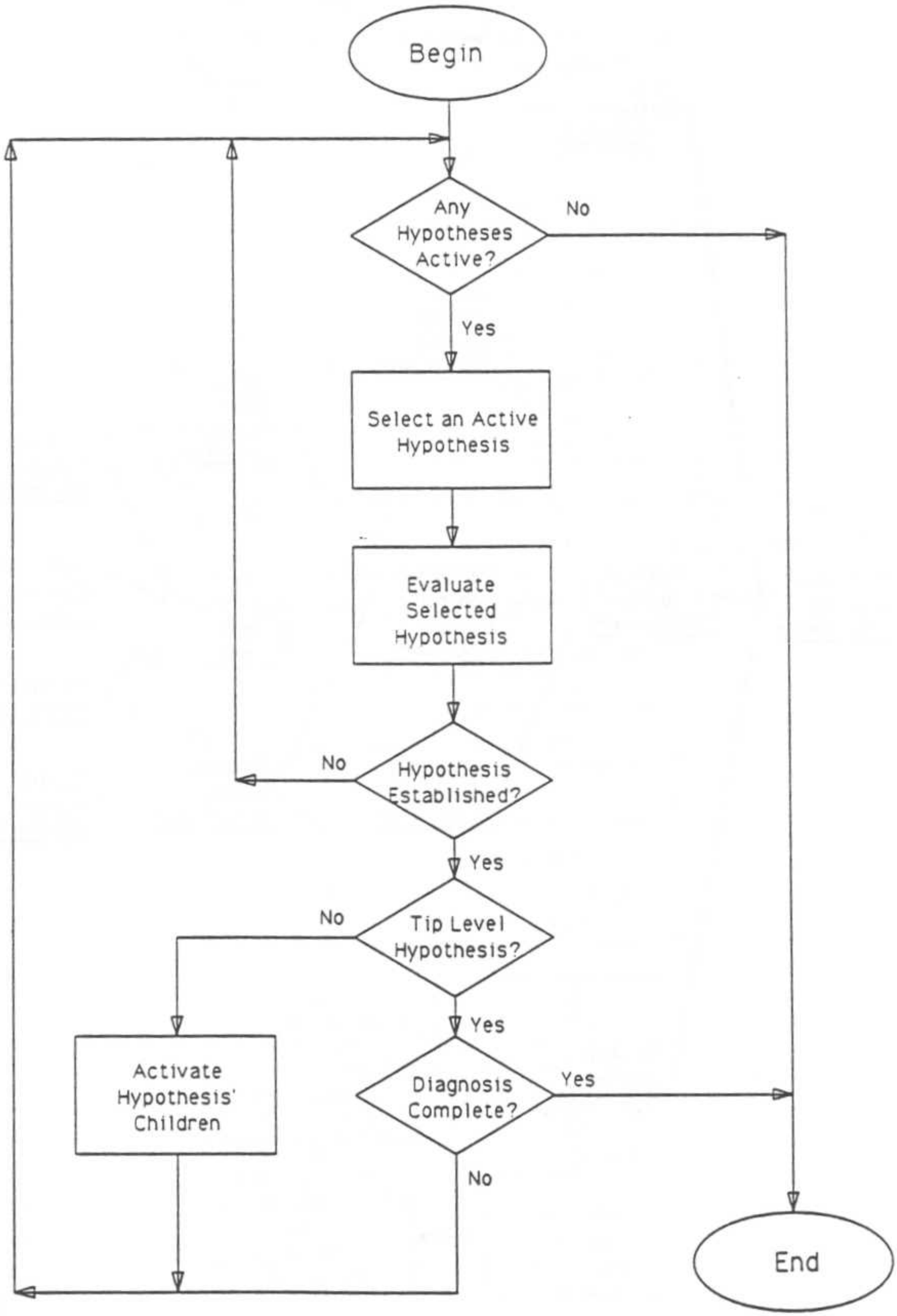

Figure 4. Establish-Refine Inference Procedure.

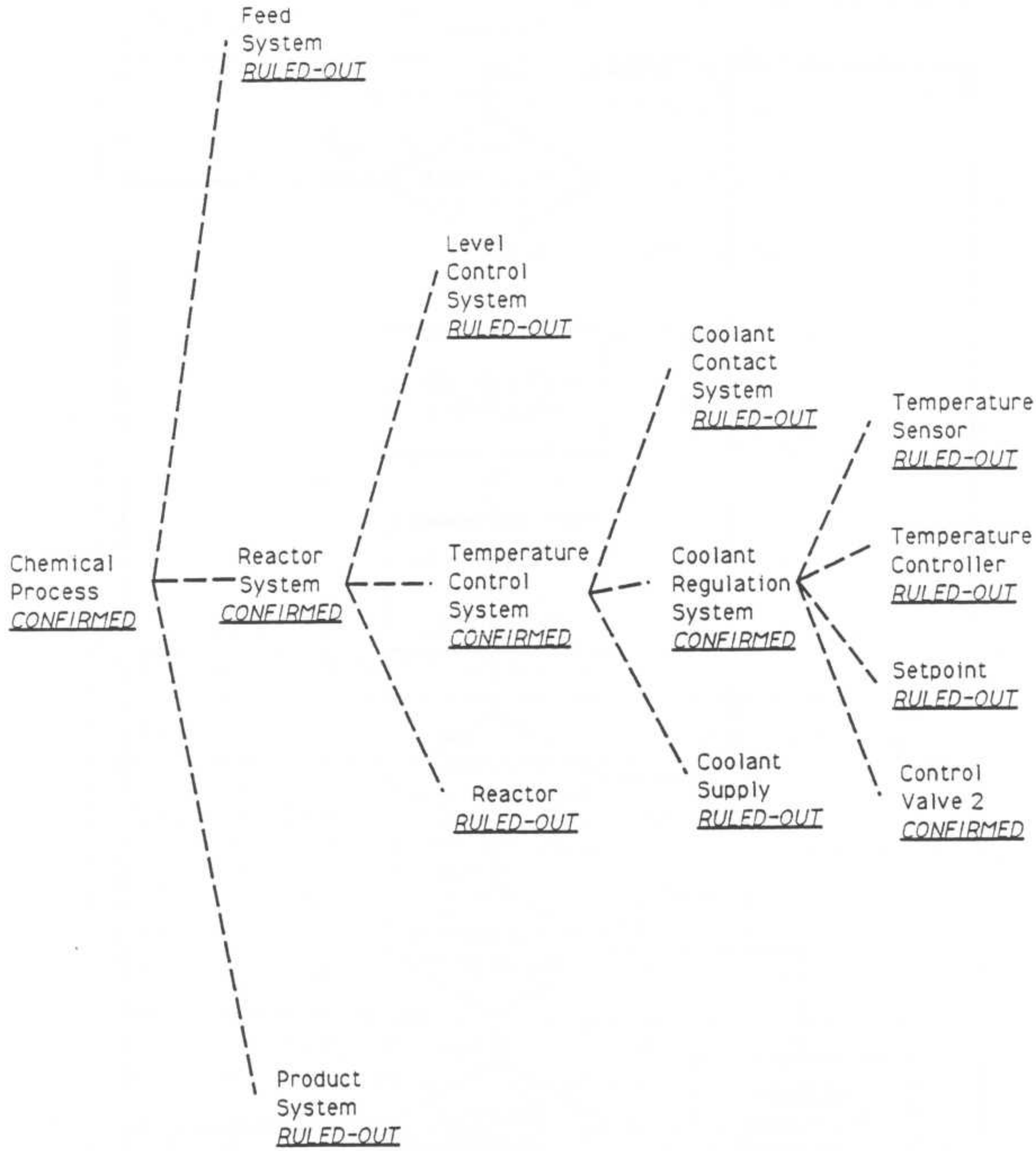

Figure 5. Sample Confidence Value Hierarchy.

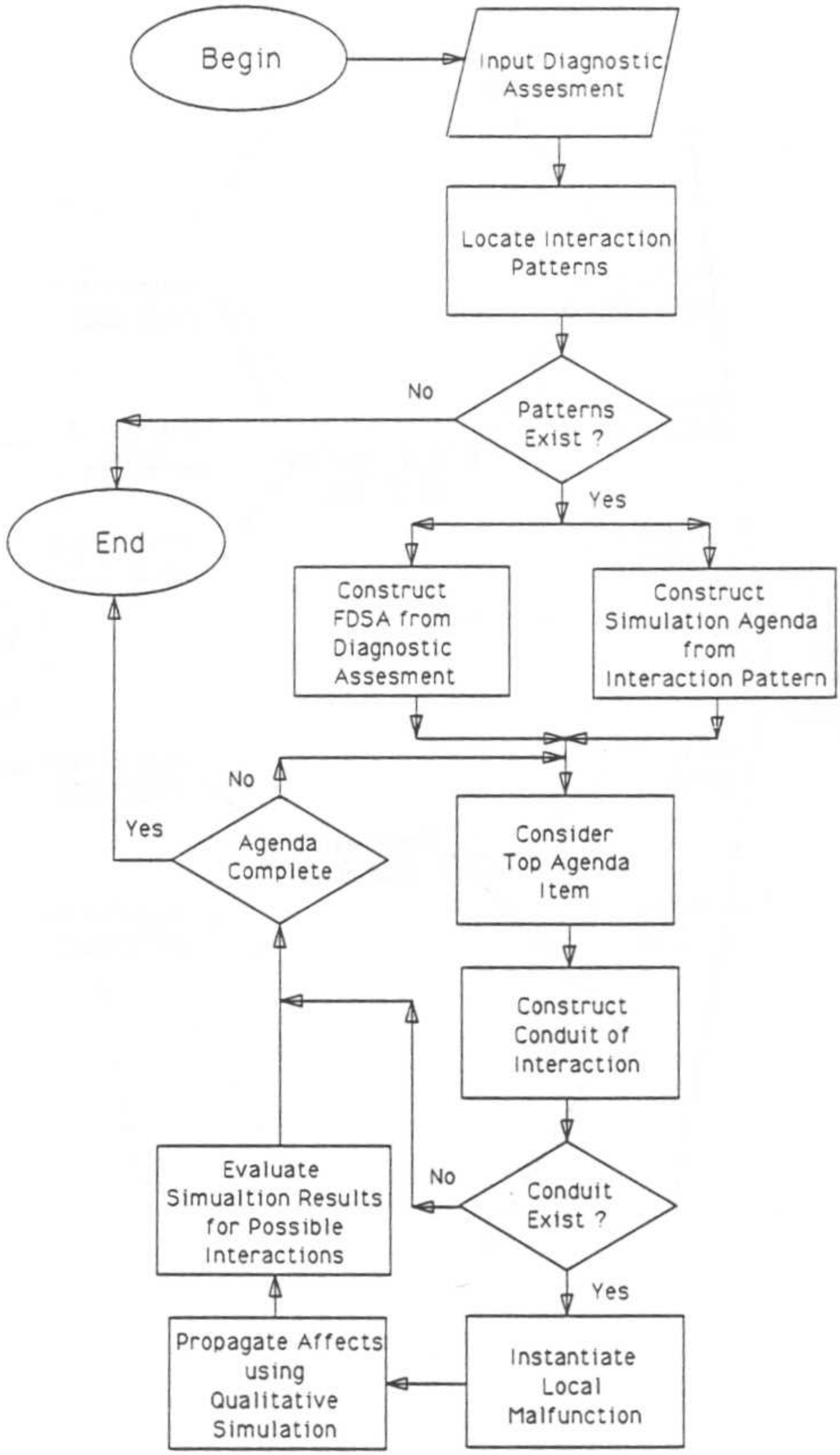

Figure 6. DFS Inference Procedure.

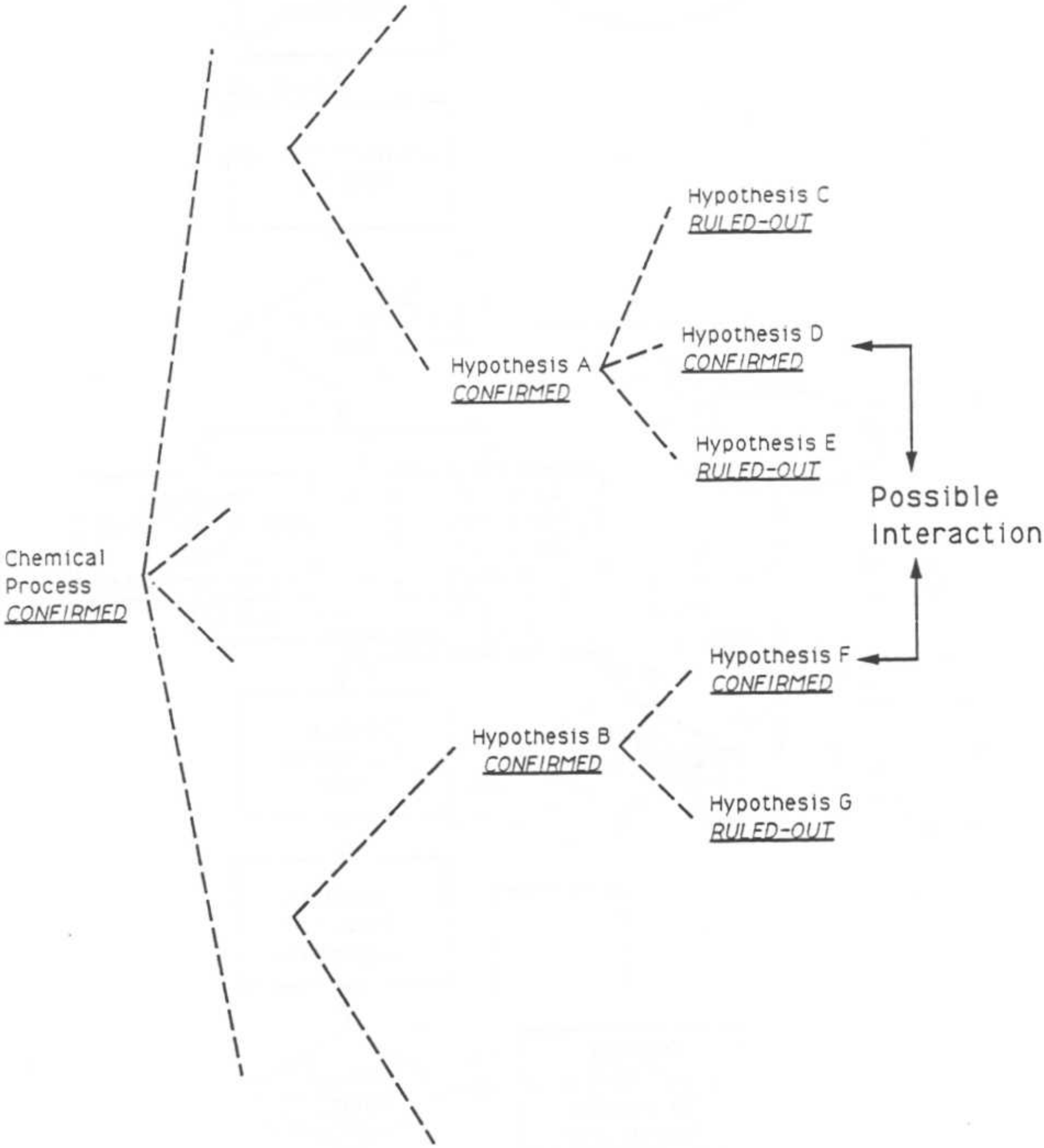

Figure 7. Interaction Pattern One.

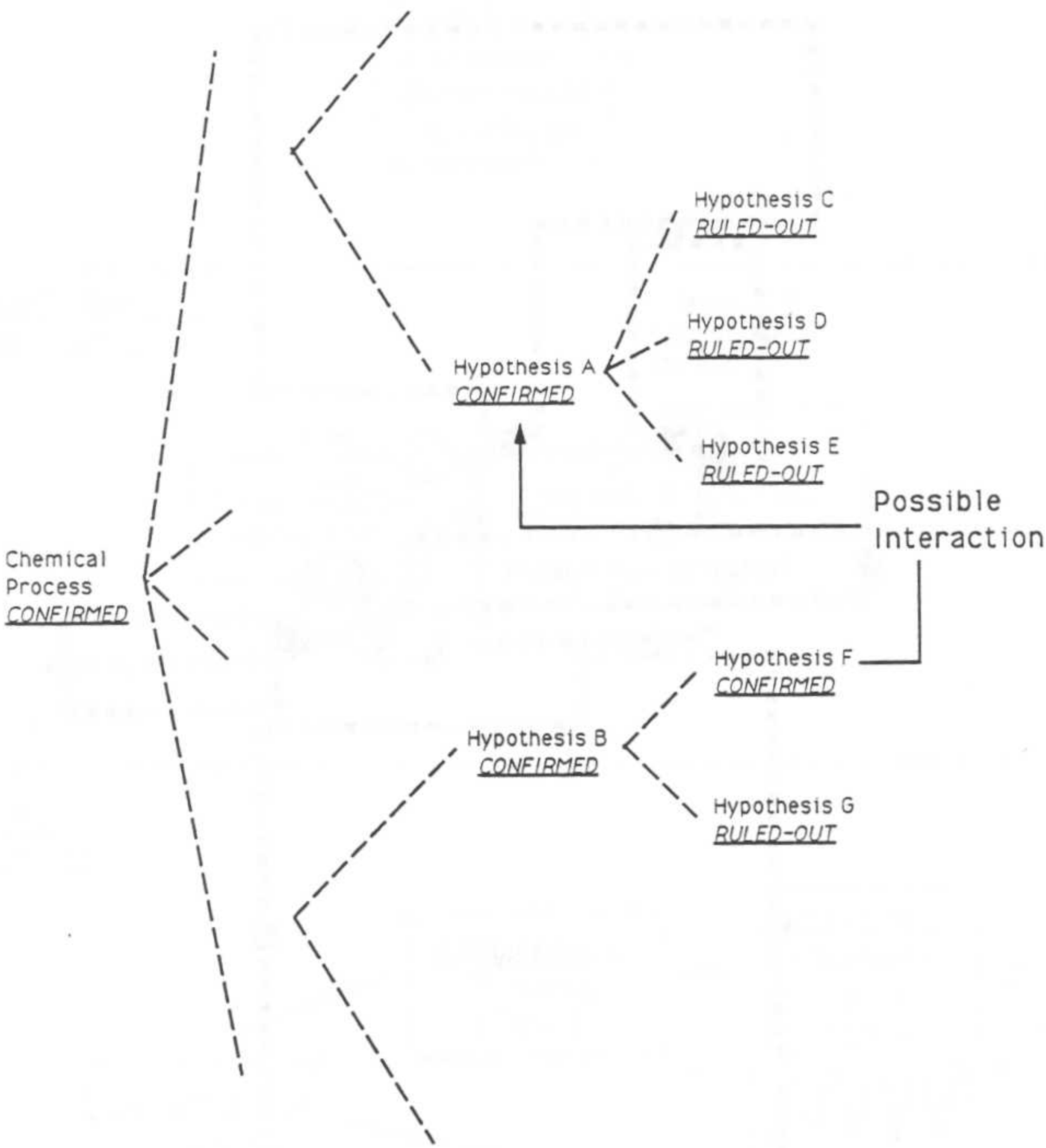

Figure 8. Interaction Pattern Two.

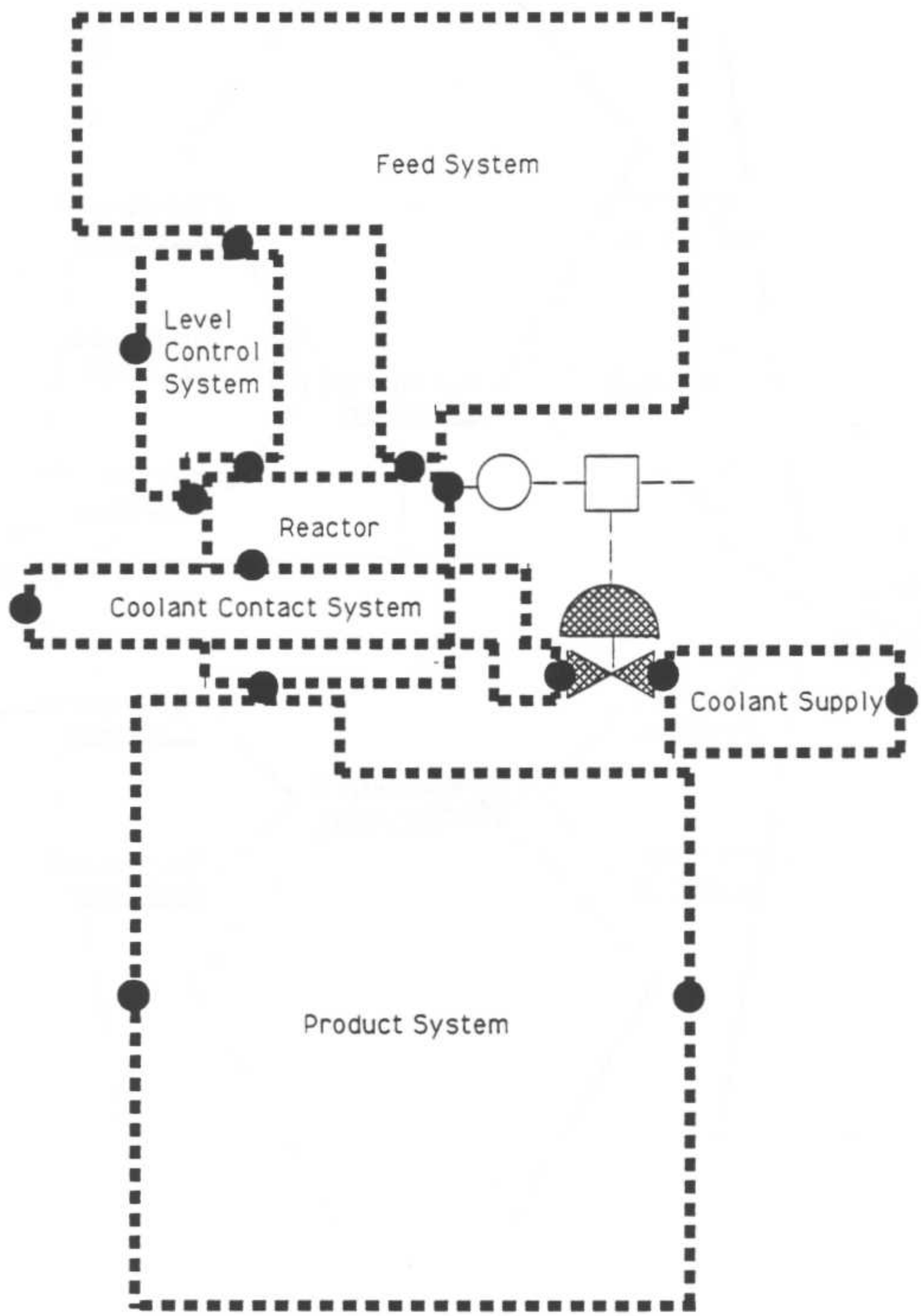

Figure 9. FDSA of the Chemical Process.

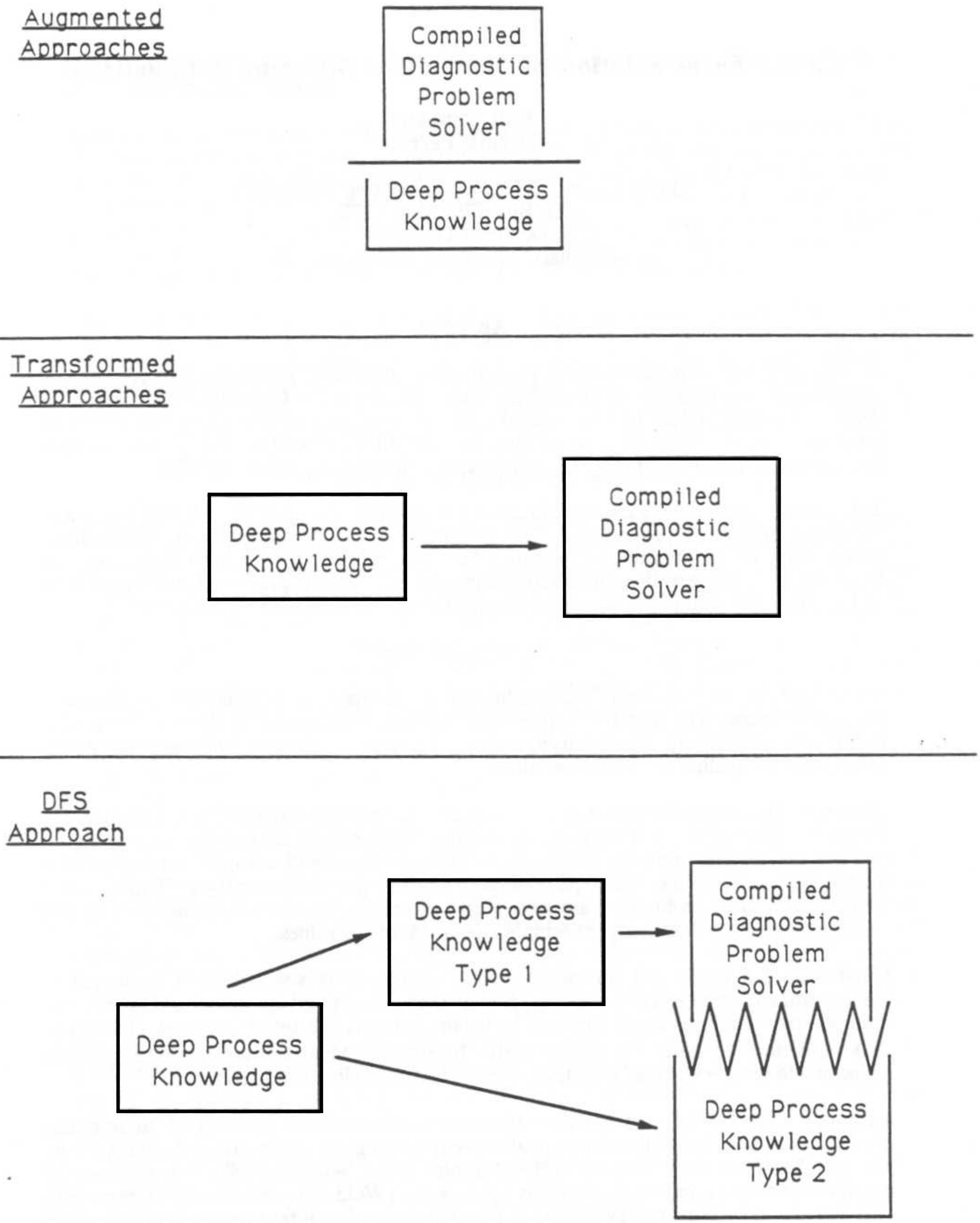

Figure 10. Approaches to Integration.