Strong Method Problem Solving.

- Introduction
- Overview of Expert System Technology
- Rule-Based Expert Systems
- Model-Based, Case-Based and Hybrid Systems

Terms

- Knowledge -- the information a computer program must have to behave intelligently.
- Domain knowledge -- knowledge about the problem domain; e.g., knowledge about geology in an expert system for finding mineral deposits.

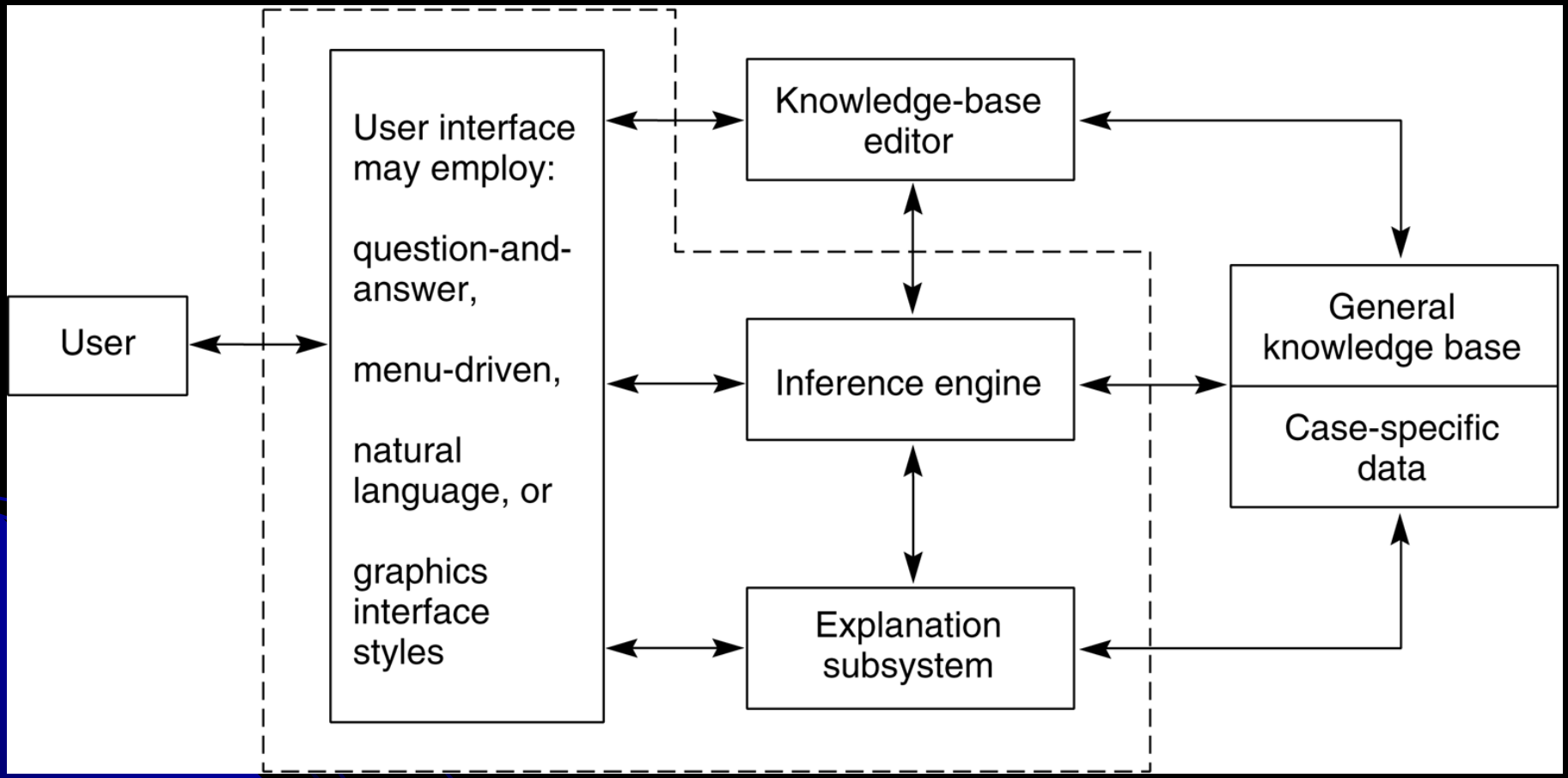
Terms

- Knowledge-based system -- a computer program in which the domain knowledge is explicit and separate from the rest of the system.
- Expert system -- a knowledge-based program that provides "expert quality" solutions to problems in a specific domain.
- Knowledge base -- the portion of a knowledge-based system or expert system which contains the domain knowledge.
- Inference engine -- the part of a knowledge-based system or expert system that contains the general problem-solving knowledge.

Production System vs. Expert Systems

- The production system was the intellectual precursor of modern expert system architecture.
- If we regard the simple expert system architecture shown as a production system, the knowledge base is the set of production rules.
- In a rule-based expert system, these condition-action pairs are represented by if-then rules.
- The inference engine is the recognize-act cycle of the production system.

Fig 8.1 architecture of a typical expert system for a particular problem domain.



The explanation subsystem

- In an expert system, sub goals are often solved by asking the user for information.
- The explanation subsystem keeps track of the search and uses a trace of the search to answer user questions.
- In general, there are two kinds of questions supported by rule-based expert systems, i.e., why? how?

The explanation subsystem

- The explanation is often a restatement of the current rule.
- If each rule corresponds to a single logical step in the problem-solving process, the explanation will behave logically.

Knowledge Base

- the knowledge base is highly modular and modifiable
- knowledge is represented in a concise and intuitively appealing form
- the correctness of the knowledge is not automatically guaranteed.

KB systems are generally:

- open to inspection
- easily modified -- both in adding and deleting the knowledge
- heuristic -- using (often imperfect) knowledge to obtain solutions.

Guidelines to determine whether a problem is appropriate for expert system solution:

1. The need for the solution justifies the cost and effort of building an expert system.

2. Human expertise is not available in all situations where it is needed.

3. The problem may be solved using symbolic reasoning.

4. The problem domain is well structured and does not require commonsense reasoning.

5. The problem may not be solved using traditional computing methods.

6. Cooperative and articulate experts exist.

7. The problem is of proper size and scope.

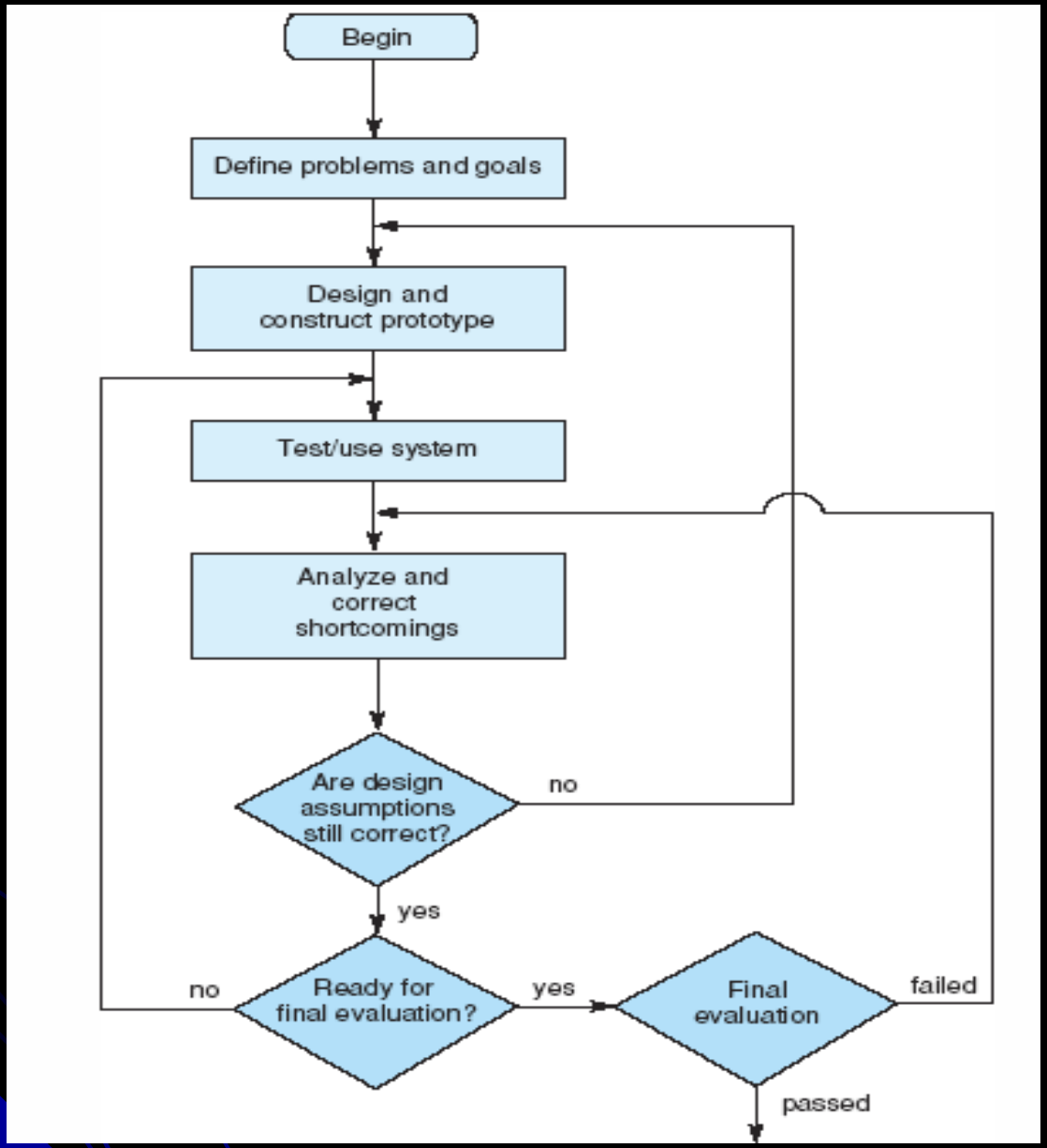
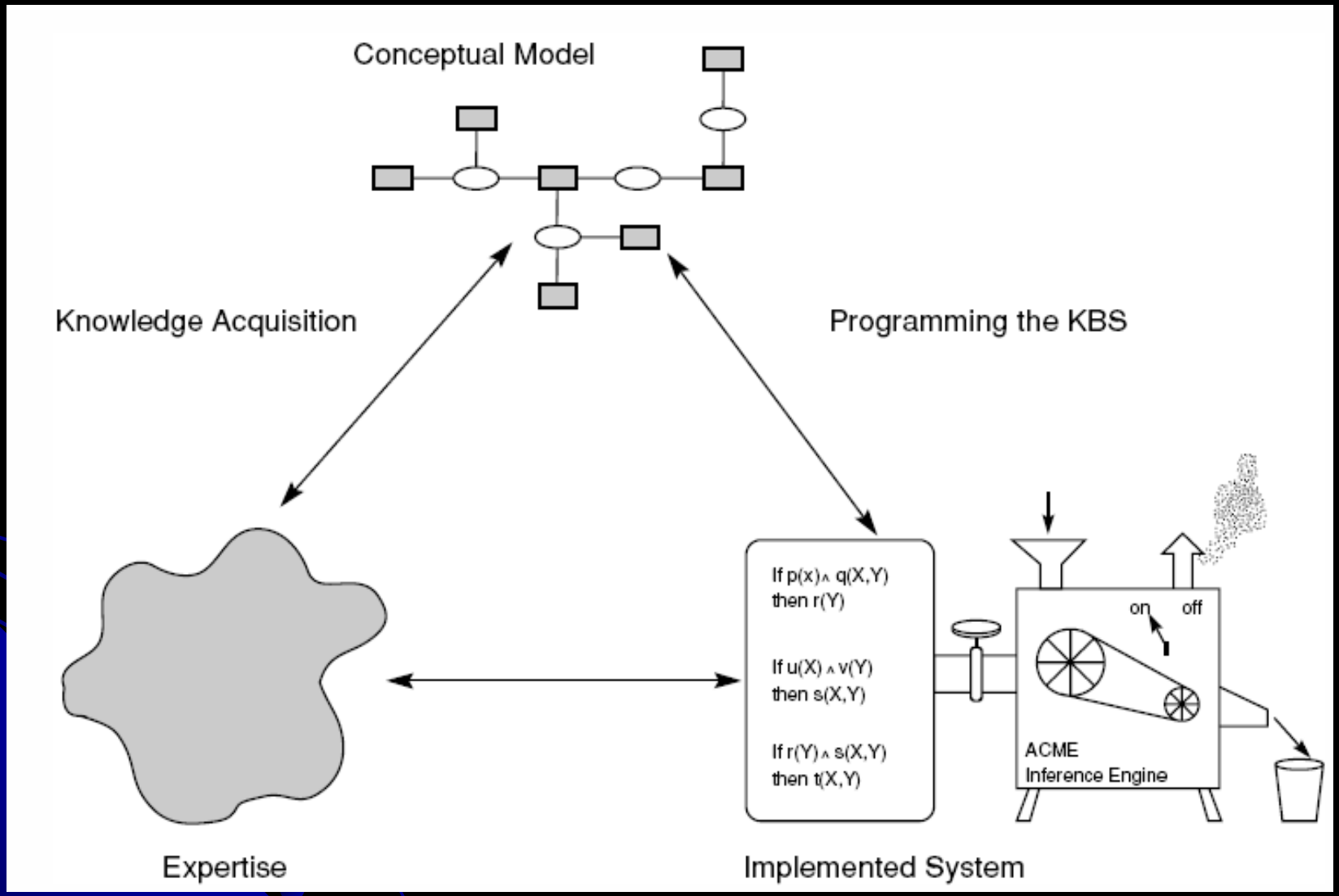


Fig 8.2 Exploratory development cycle.

Expert System Shell

- The expert system shell has all the components of the above expert system except the knowledge base contains no information.
- Since different problems often require:
  - different knowledge representations, and
  - different reasoning processes which require different control strategies,
  - it is important to select the proper expert system shell.

Fig 8.4 The role of mental or conceptual models in problem solving.



A small expert system for analysis of automotive problems.

Rule 1: if
the engine is getting gas, and
the engine will turn over,
then
the problem is spark plugs.

Rule 2: if
the engine does not turn over, and
the lights do not come on
then
the problem is battery or cables.

Rule 3: if
the engine does not turn over, and
the lights do come on
then
the problem is the starter motor.

Rule 4: if
there is gas in the fuel tank, and
there is gas in the carburetor
then
the engine is getting gas.

Fig 8.5 The production system at the start of a consultation in the car diagnostic example.

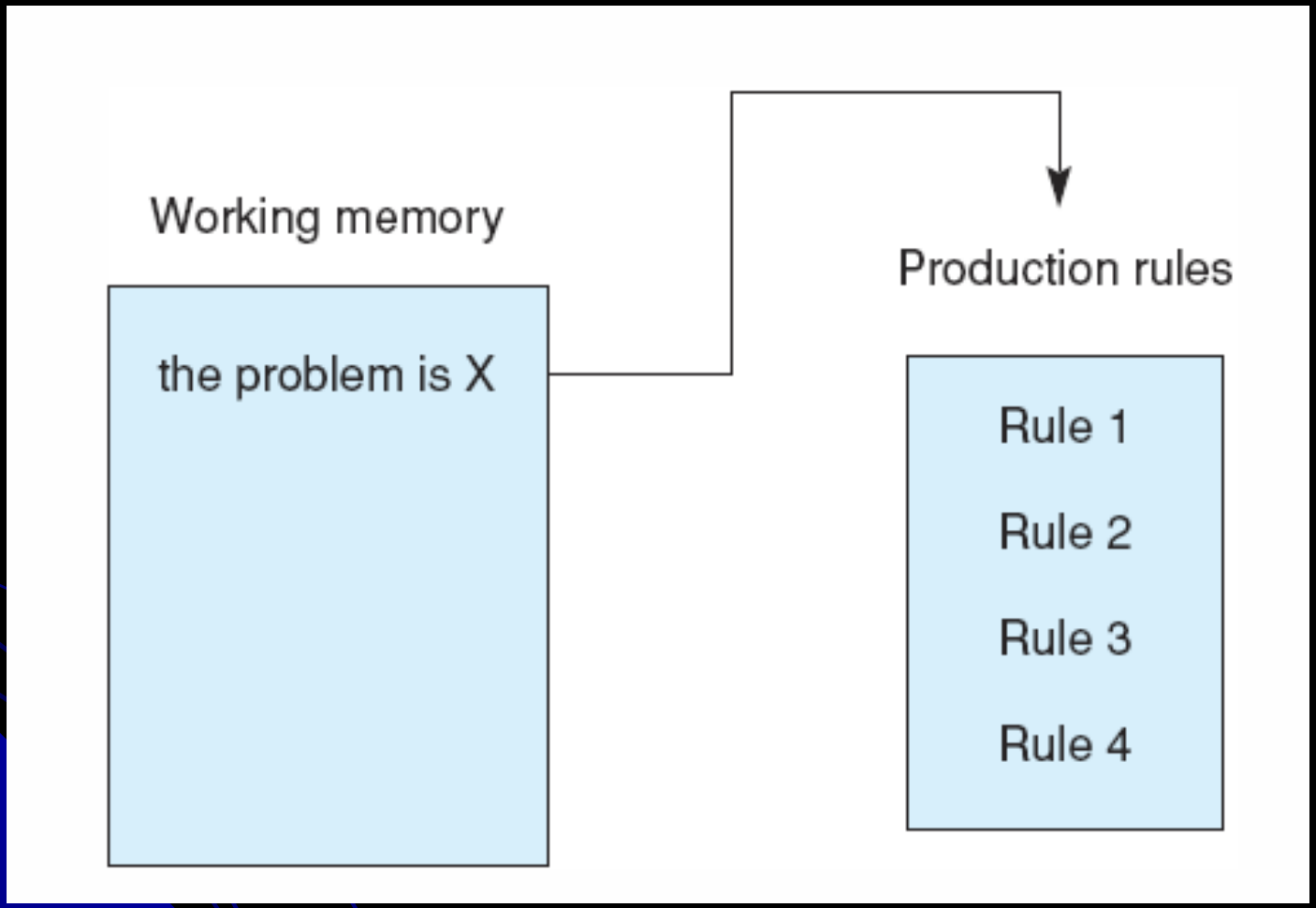


Fig 8.6 The production system after Rule 1 has fired.

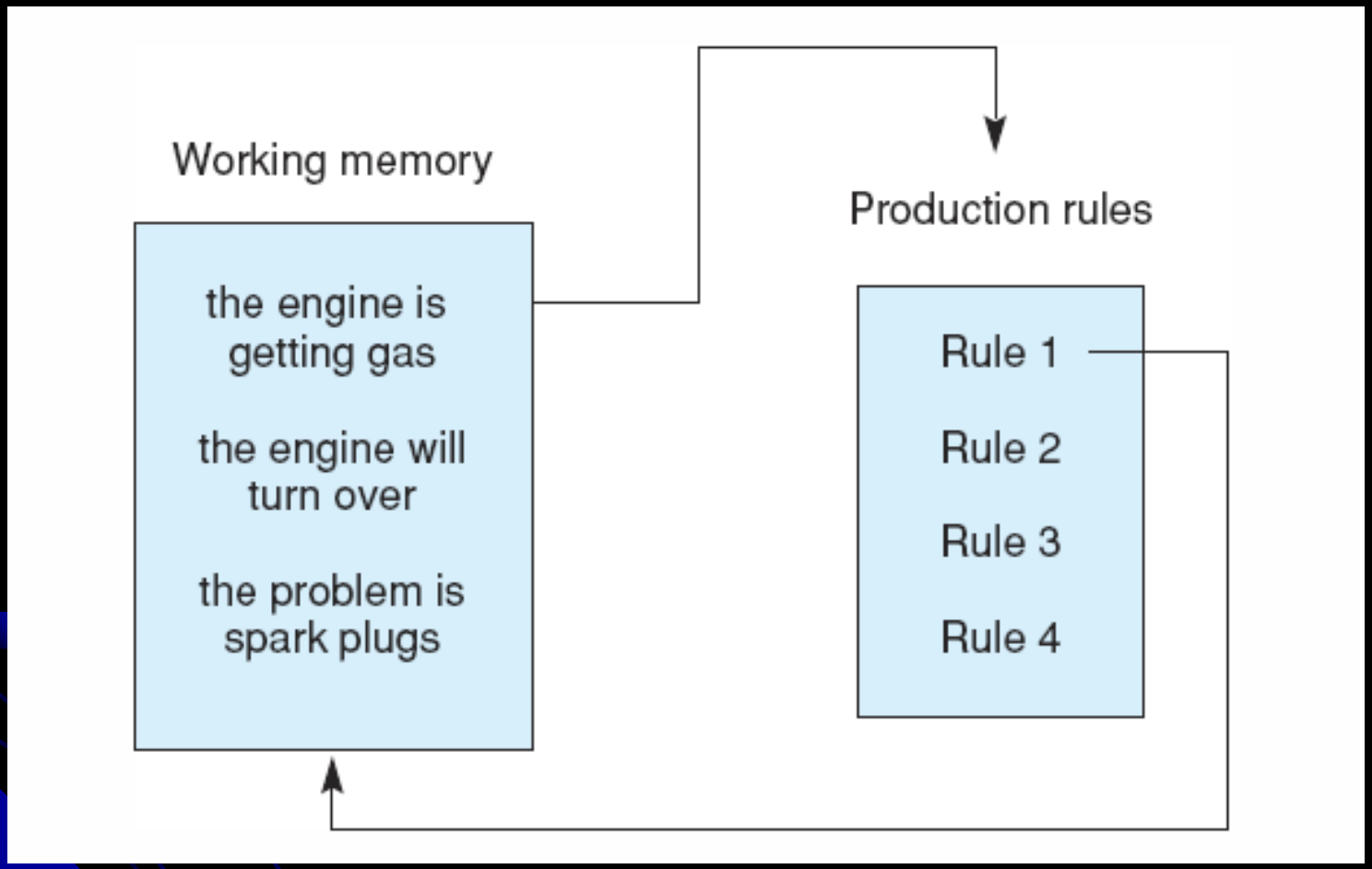


Fig 8.7 The system after Rule 4 has fired. Note the stack-based approach to goal reduction.

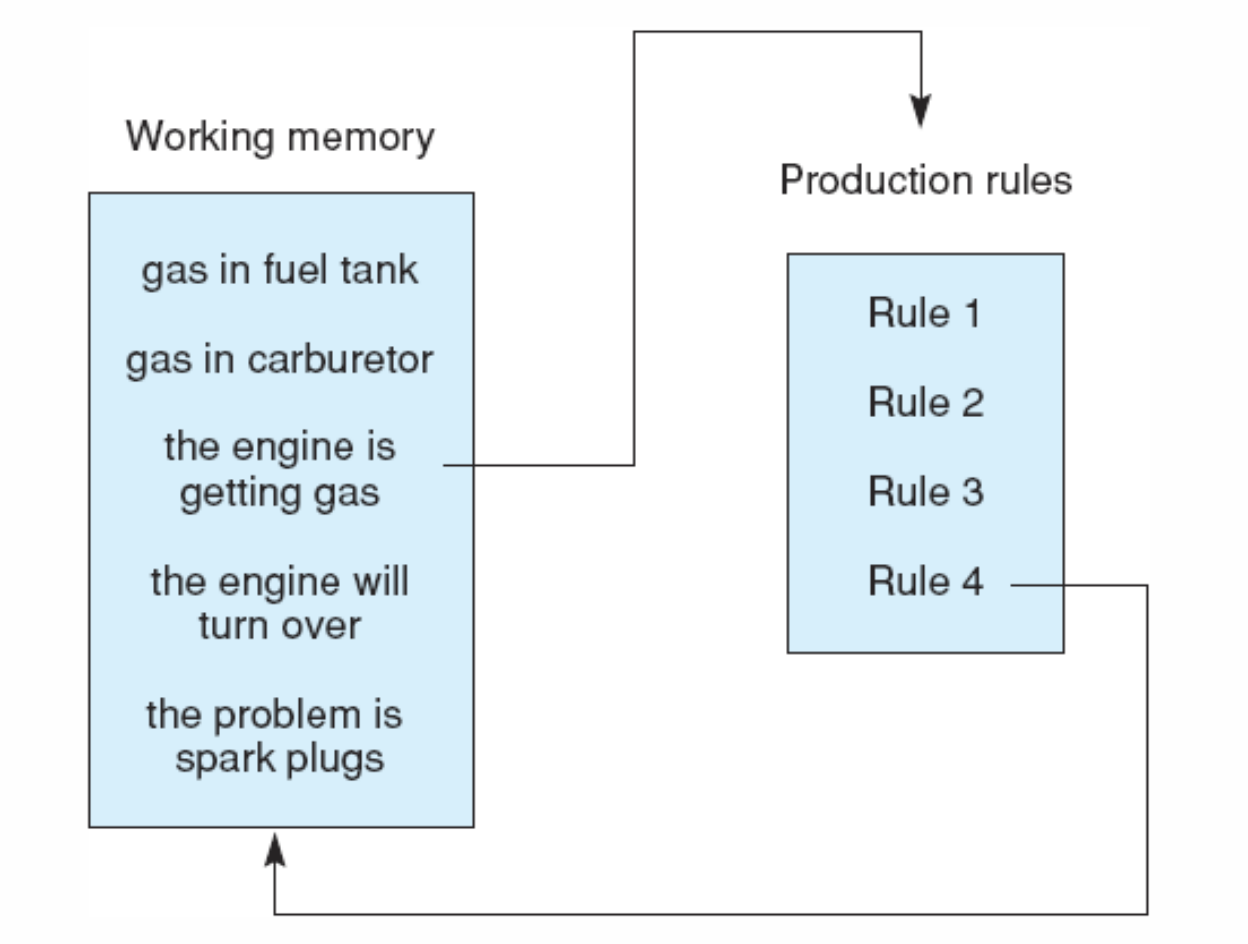
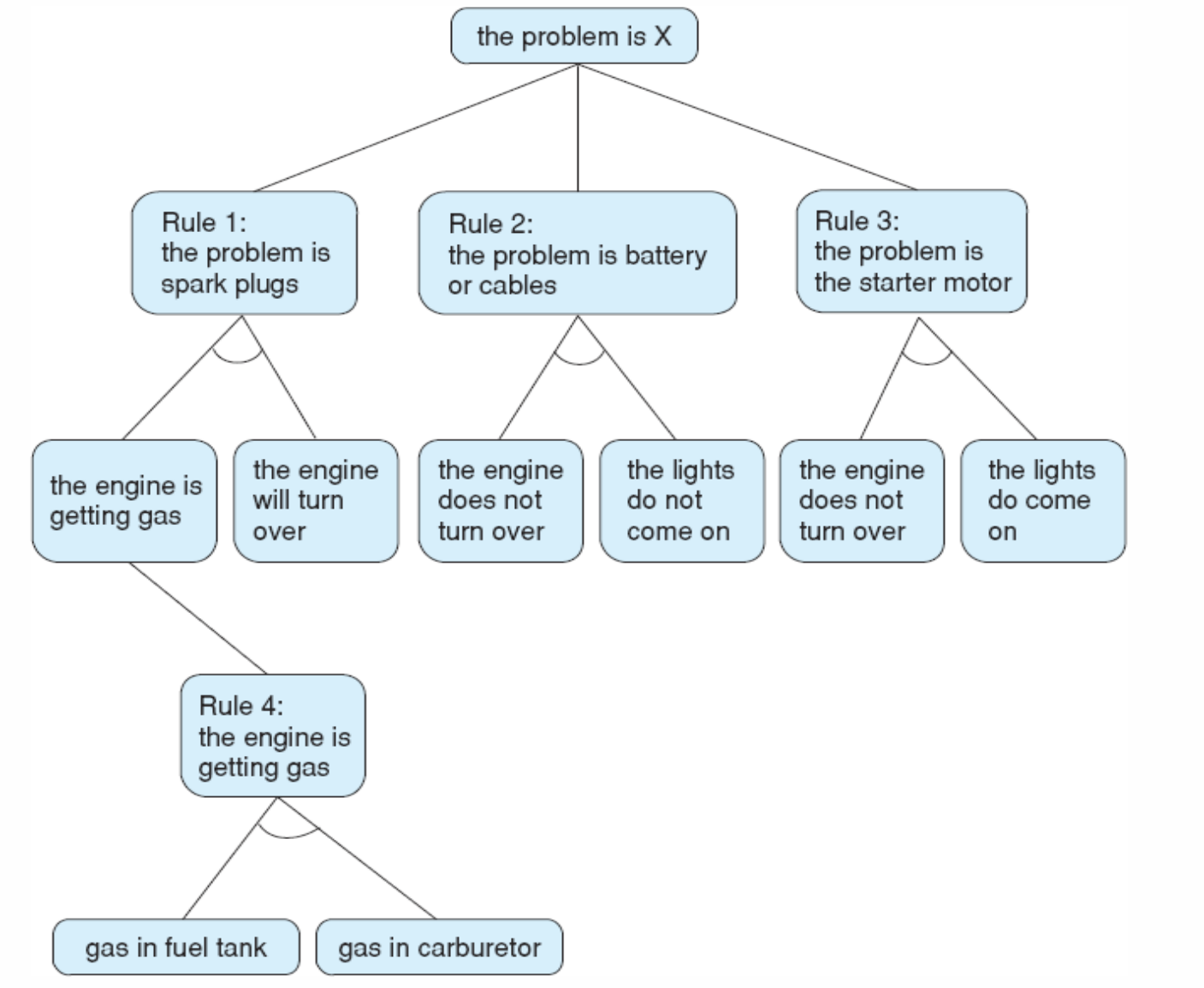


Fig 8.8 The and/or graph searched in the car diagnosis example, with the conclusion of Rule 4 matching the first premise of Rule 1.



The following dialogue begins with the computer asking the user about the goals present in working memory.

Gas in fuel tank?
Yes
Gas in carburetor?
Yes
Engine will turn over?
Why

It has been established that:
1. The engine is getting gas,
2. The engine will turn over,
Then the problem is the spark plugs.

How the engine is getting gas

This follows from rule 4:
if
gas in fuel tank, and
gas in carburetor
then
engine is getting gas.
gas in fuel tank was given by the user
gas in carburetor was given by the user

Fig 8.9 The production system at the start of a consultation for data-driven reasoning.

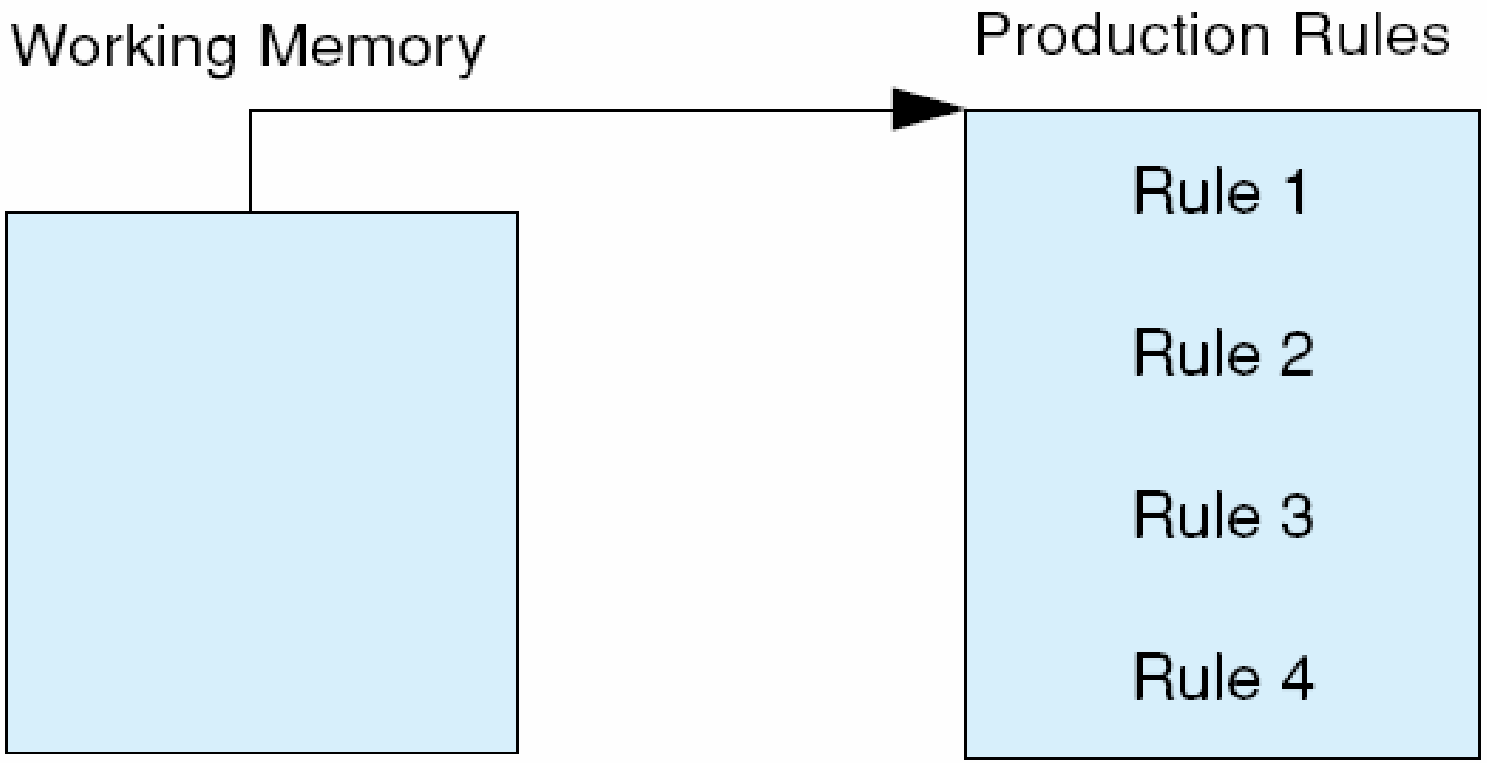


Fig 8.10 The production system after evaluating the first premise of Rule 2, which then fails.

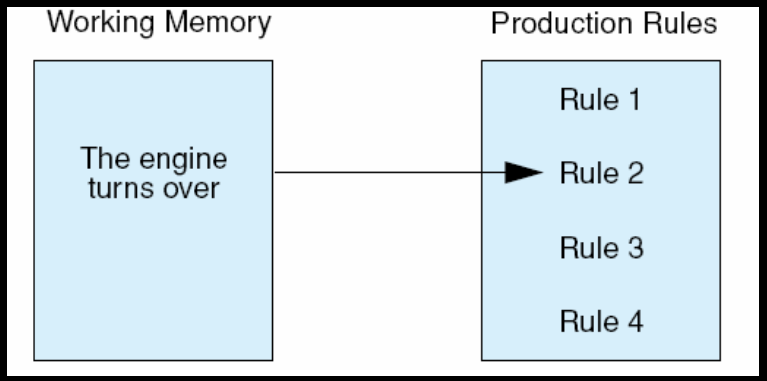


Fig 8.11 The data-driven production system after considering Rule 4, beginning its second pass through the rules.

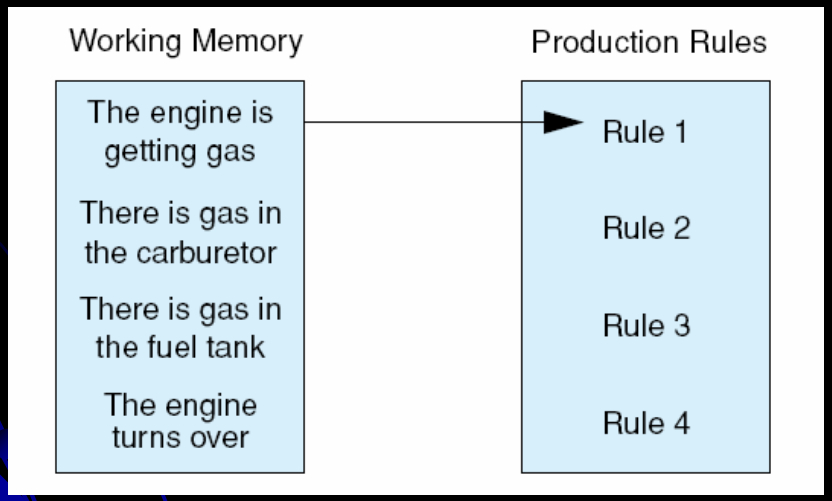


Fig 8.12 The search graph as described by the contents of working memory (WM) for the data-driven breadth-first search of the rule set of Section 8.2.1

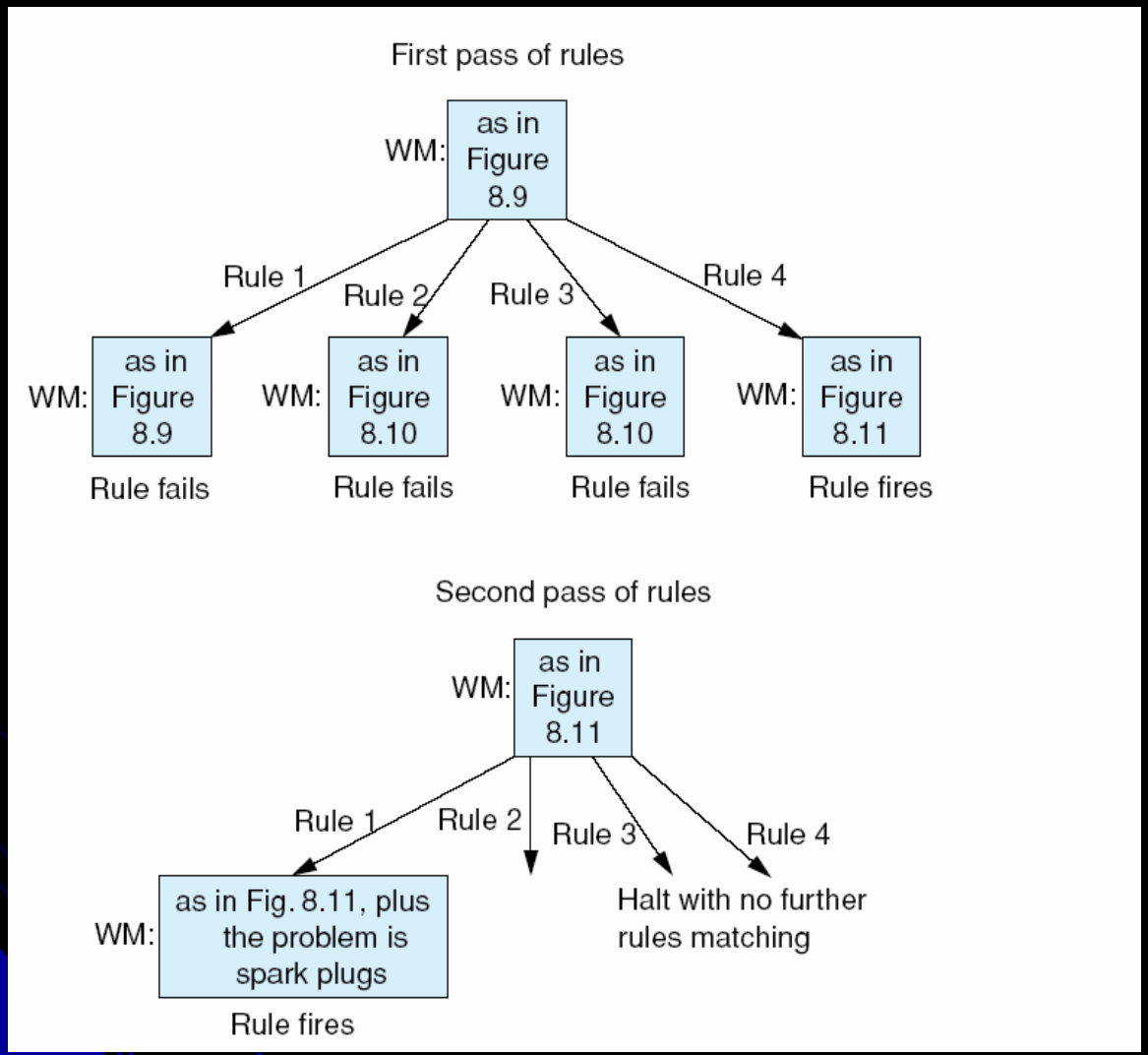


Fig 8.13 The behavior description of an adder after Davis and Hamscher (1992)

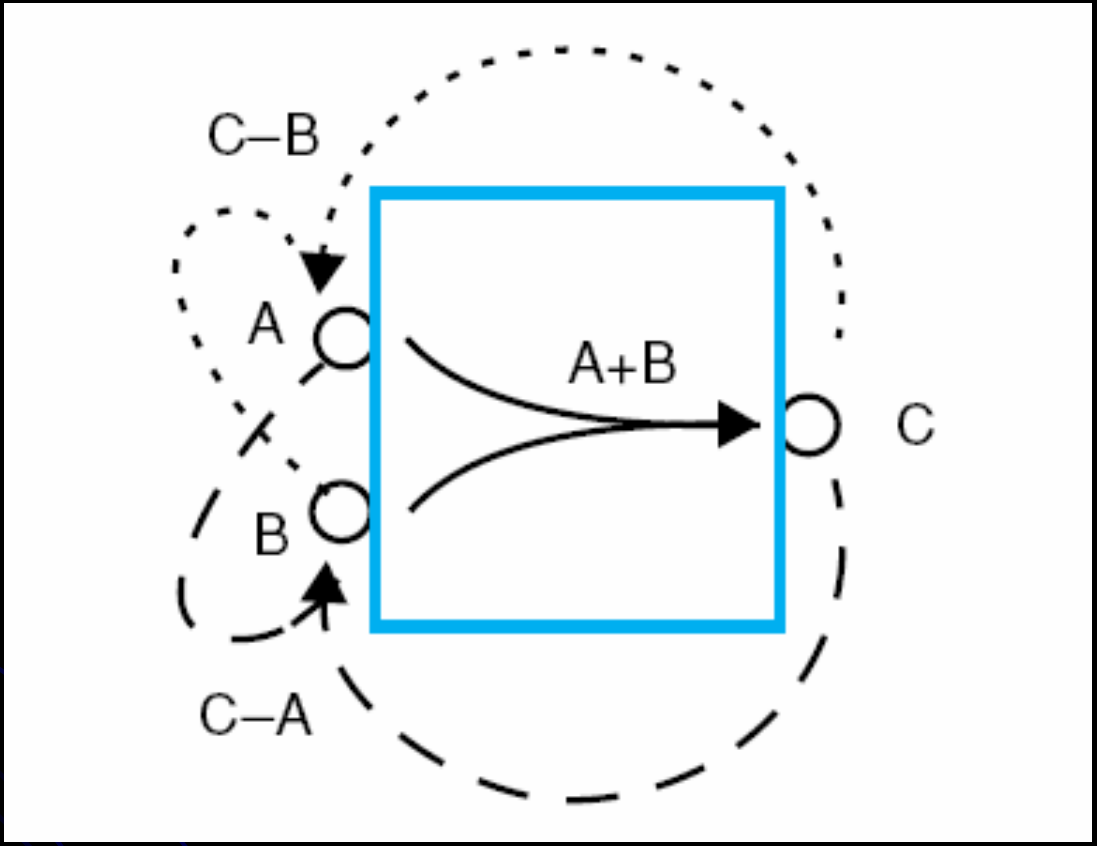


Fig 8.14 Taking advantage of direction of information flow, after Davis and Hamscher (1992).

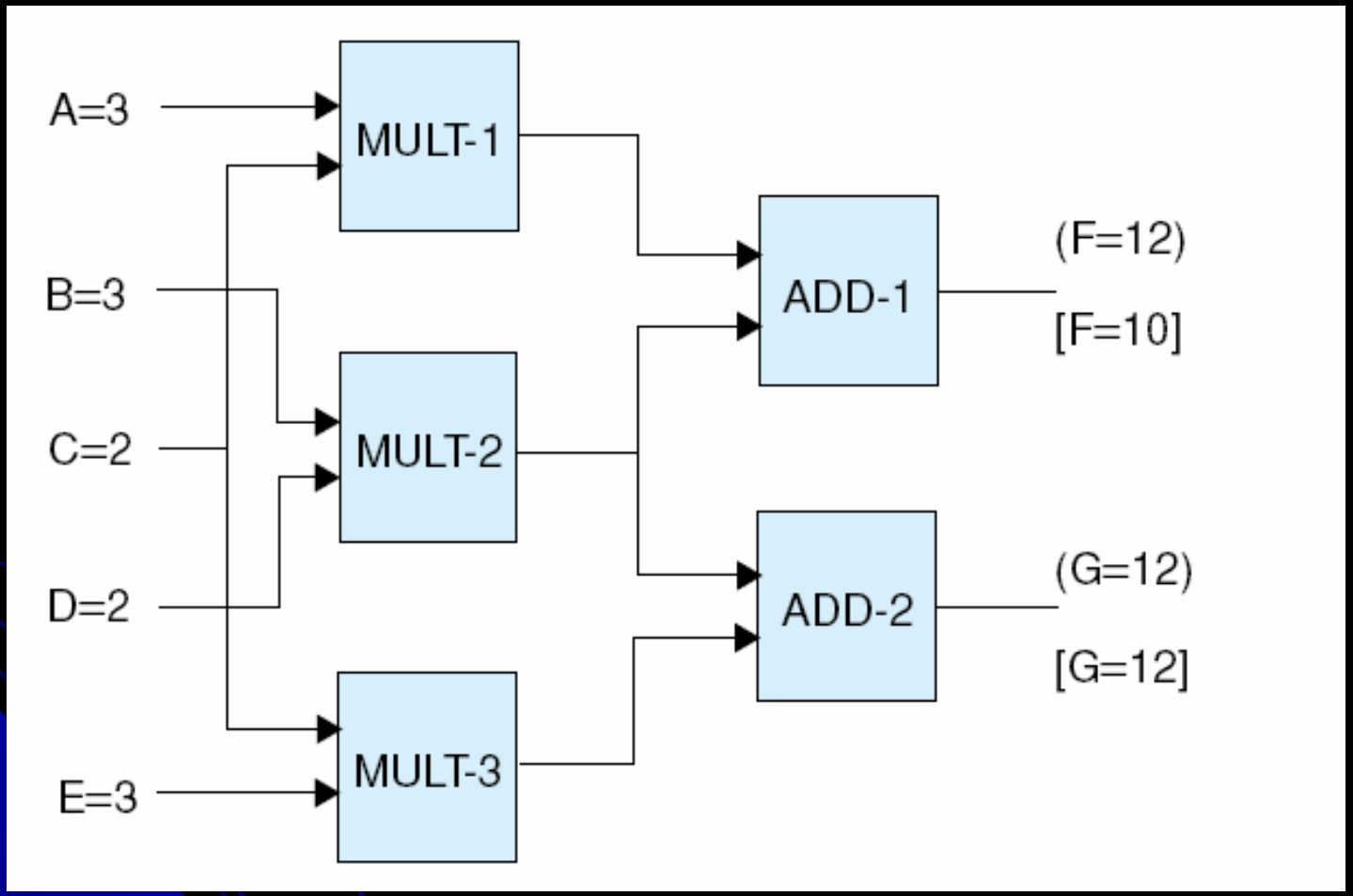


Fig 8.15 A schematic of the simplified Livingstone propulsion system, from Williams and Nayak (1996b).

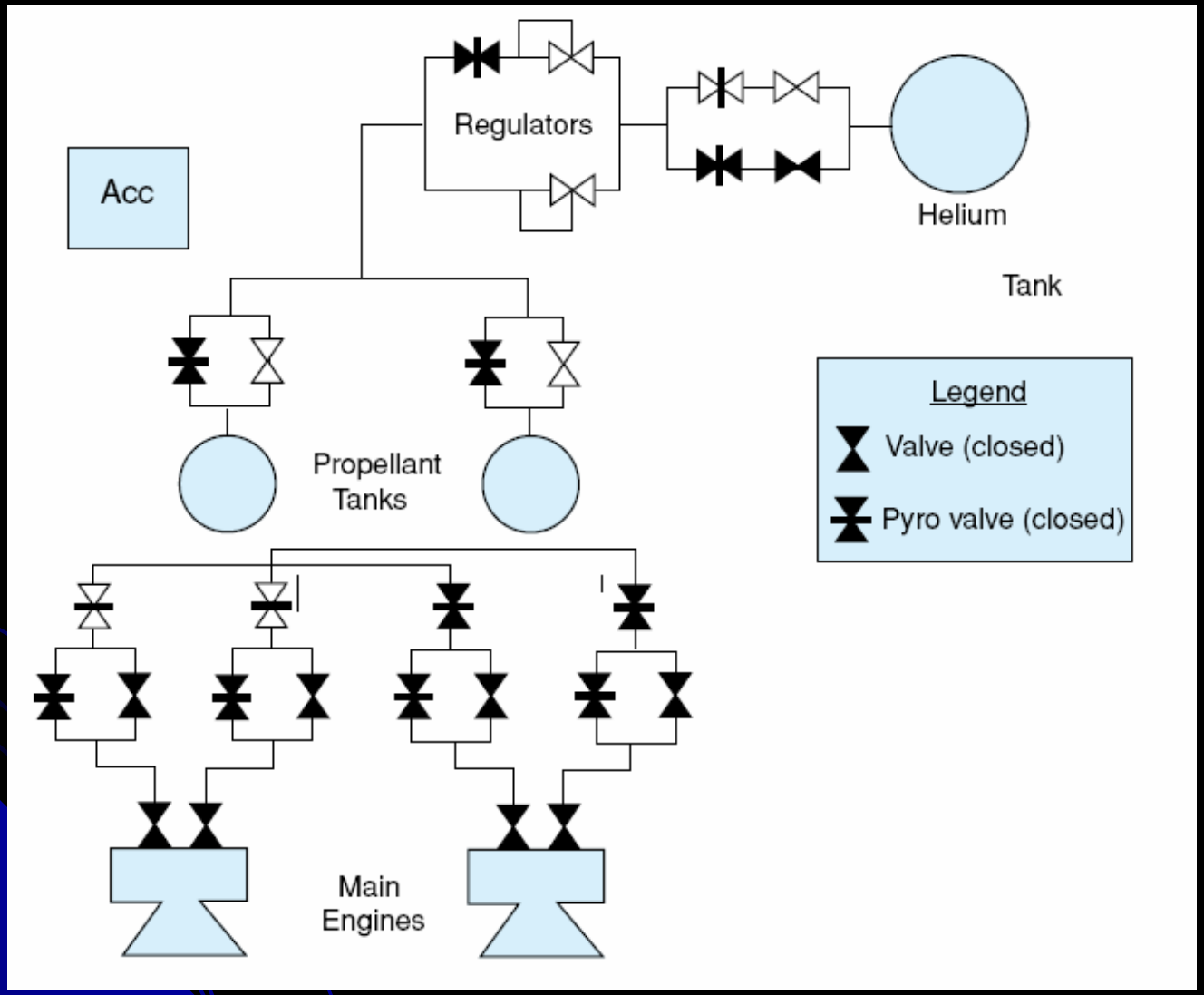
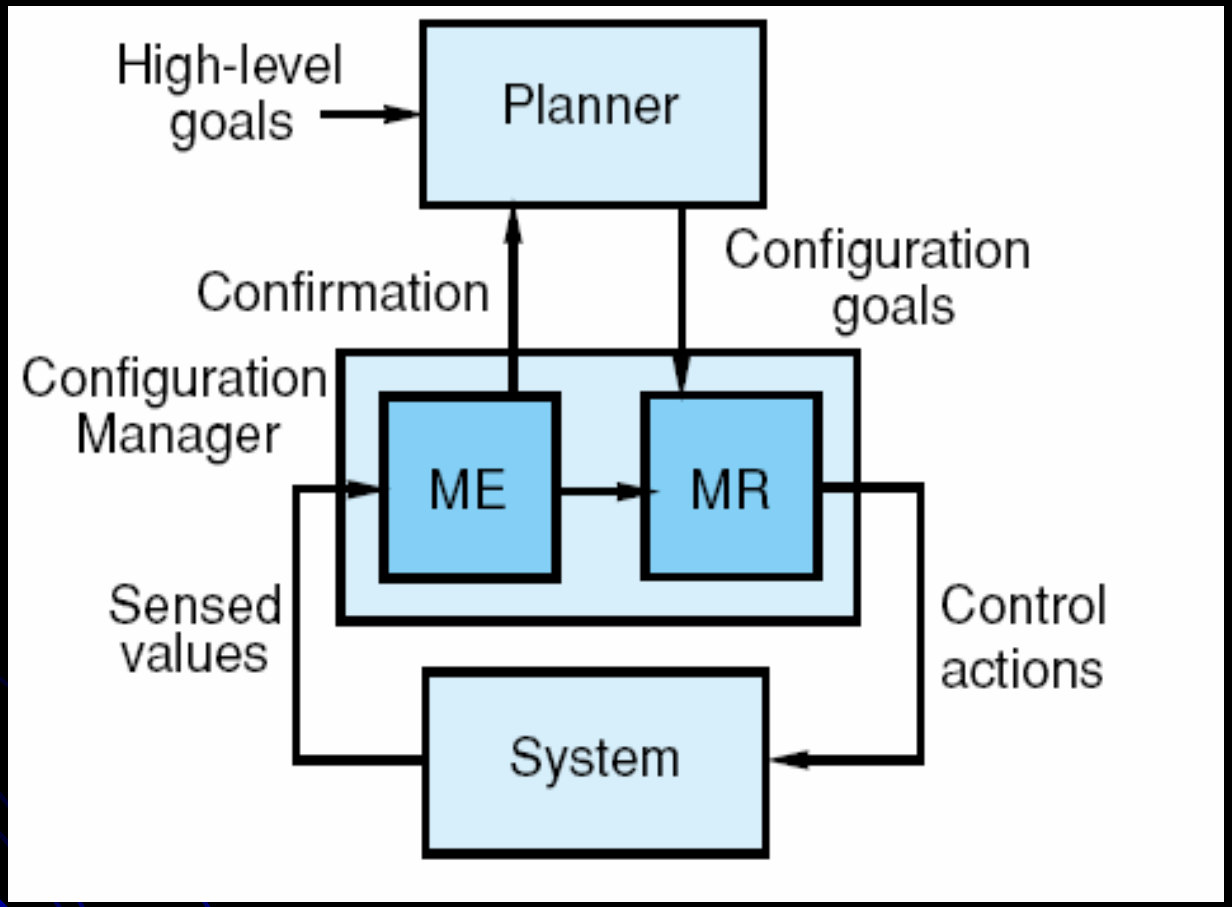


Fig 8.16 a model-based configuration management system, from Williams and Nayak (1996b).



Case-based reasoners share a common structure

For each new problem they:

1. Retrieve appropriate cases from memory.

2. Modify a retrieved case so that it will apply to the current situation.

3. Apply the transformed case.

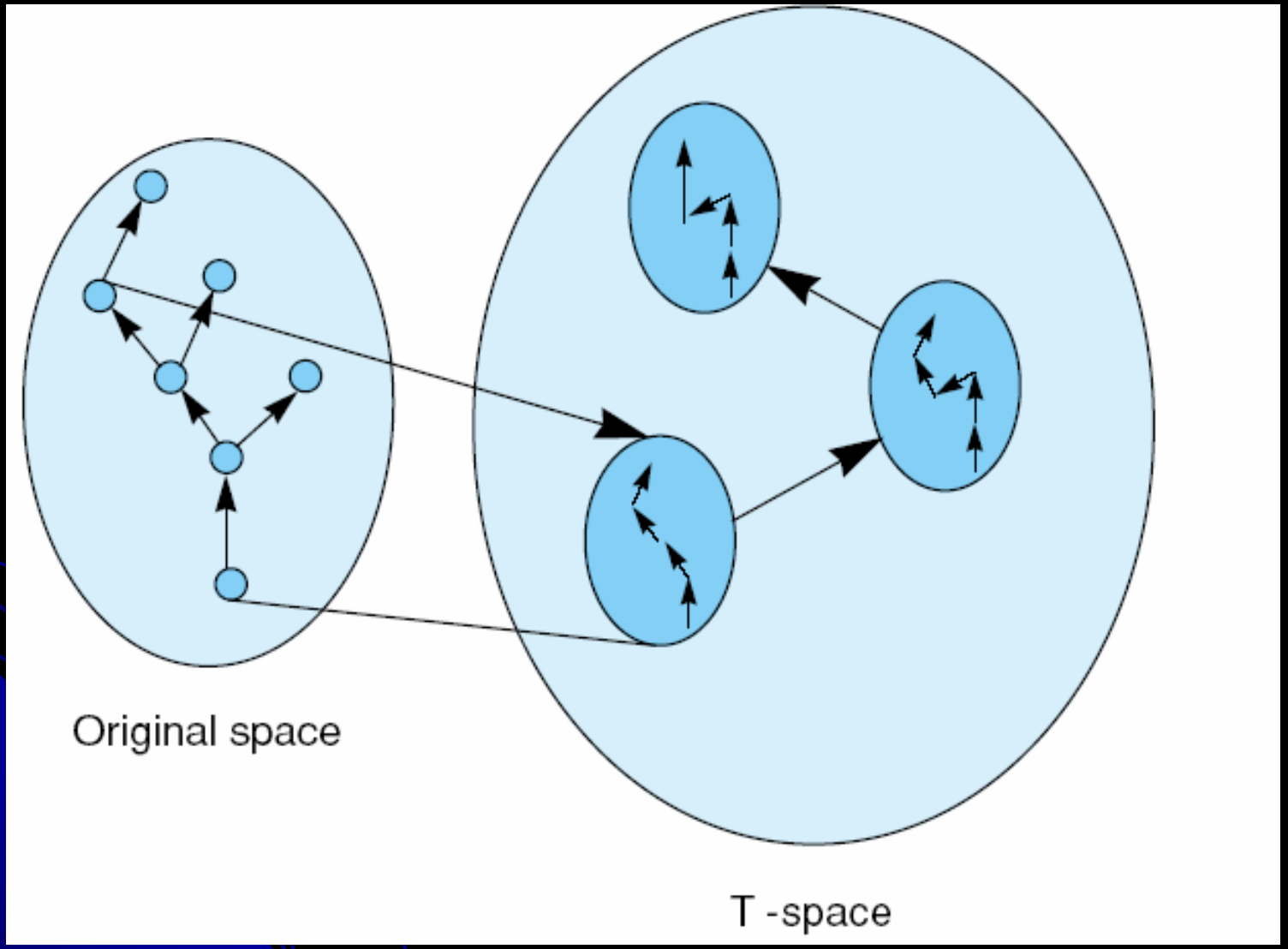
4. Save the solution, with a record of success or failure, for future use.

Kolodner (1993) offers a set of possible preference heuristics to help organize the storage and retrieval of cases.

These include:

1. *Goal-directed preference.* Organize cases, at least in part, by goal descriptions. Retrieve cases that have the same goal as the current situation.
2. *Salient-feature preference.* Prefer cases that match the most important features or those matching the largest number of important features.
3. *Specify preference.* Look for as exact as possible matches of features before considering more general matches.
4. *Frequency preference.* Check first the most frequently matched cases.
5. *Recency preference.* Prefer cases used most recently.
6. *Ease of adaptation preference.* Use first cases most easily adapted to the current situation.

Fig 8.17 Transformational analogy, adapted from Carbonell (1983).



The advantages of a rule-based approach include:

1. The ability to use, in a very direct fashion, experiential knowledge acquired from human experts. This is particularly important in domains that rely heavily on heuristics to manage complexity and/or missing information.
2. Rules map into state space search. Explanation facilities support debugging.
3. The separation of knowledge from control simplifies development of expert systems by enabling an iterative development process where the engineer acquires, implements, and tests individual rules.
4. Good performance is possible in limited domains. Because of the large amounts of knowledge required for intelligent problem solving, expert systems are limited to narrow domains. However, there are many domains where design of an appropriate system has proven extremely useful.
5. Good explanation facilities. Although the basic rule-based framework supports flexible, problem-specific explanations, it must be mentioned that the ultimate quality of these explanations depends upon the structure and content of the rules. Explanation facilities differ widely between data- and goal-driven systems.

Disadvantages of rule-based reasoning include:

1. Often the rules obtained from human experts are highly heuristic in nature, and do not capture, functional or model-based knowledge of the domain.
2. Heuristic rules tend to be "brittle" and cannot handle missing information or unexpected data values.
3. Another aspect of the brittleness of rules is a tendency to degrade rapidly near the "edges" of the domain knowledge. Unlike humans, rule-based systems are usually unable to fall back on first principles of reasoning when confronted with novel problems.
4. Explanations function at the descriptive level only, omitting theoretical explanations. This follows from the fact that heuristic rules gain much of their power by directly associating problem symptoms with solutions, without requiring (or enabling) deeper reasoning.
5. The knowledge tends to be very task dependent. Formalized domain knowledge tends to be very specific in its applicability. Currently, knowledge representation languages do not approach human flexibility.

The advantages of case-based reasoning include:

1. The ability to encode historical knowledge directly. In many domains, cases can be obtained from existing case histories, repair logs, or other sources, eliminating the need for intensive knowledge acquisition with a human expert.

2. Allows shortcuts in reasoning. If an appropriate case can be found, new problems can often be solved in much less time than it would take to generate a solution from rules or models.

3. It allows a system to avoid past errors and exploit past successes. CBR provides a model of learning that is both theoretically interesting and practical enough to apply to complex problems.

4. Extensive analysis of domain knowledge is not required. Unlike a rule-based system, where the knowledge engineer must anticipate rule interactions, CBR allows a simple additive model for knowledge acquisition. This requires an appropriate representation for cases, a useful retrieval index, and a case adaptation strategy.

5. Appropriate indexing strategies add insight and problem-solving power. The ability to distinguish differences in target problems and select an appropriate case is an important source of a case-based reasoner's power; often, indexing algorithms can provide this functionality automatically.

The disadvantages of case-based reasoning include:

1. Cases do not often include deeper knowledge of the domain. This handicaps explanation facilities, and in many situations it allows the possibility that cases may be misapplied, leading to wrong or poor quality advice.
2. A large case base can suffer problems from store/compute trade-offs.
3. It is difficult to determine good criteria for indexing and matching cases. Currently, retrieval vocabularies and similarity matching algorithms must be carefully hand crafted; this can offset many of the advantages CBR offers for knowledge acquisition.

The disadvantages of model-based reasoning include:

1. A lack of experiential (descriptive) knowledge of the domain. The heuristic methods used by rule-based approaches reflect a valuable class of expertise.

2. It requires an explicit domain model. Many domains, such as the diagnosis of failures in electronic circuits, have a strong scientific basis that supports model-based approaches. However, many domains, such as some medical specialties, most design problems, or many financial applications, lack a well-defined scientific theory. Model-based approaches cannot be used in such cases.

3. High complexity. Model-based reasoning generally operates at a level of detail that leads to significant complexity; this is, after all, one of the main reasons human experts develop heuristics in the first place.

4. Exceptional situations. Unusual circumstances, for example, bridging faults or the interaction of multiple failures in electronic components, can alter the functionality of a system in ways difficult to predict a priori.

For example, the combination of rule-based and case-based systems can:

1. Offer a natural first check against known cases before undertaking rule-based reasoning and the associated search costs.
2. Provide a record of examples and exceptions to solutions through retention in the case base.
3. Record search-based results as cases for future use. By saving appropriate cases, a reasoner can avoid duplicating costly search.

The combination of rule-based and model-based systems can:

1. Enhance explanations with functional knowledge. This can be particularly useful in tutorial applications.

2. Improve robustness when rules fail. If there are no heuristic rules that apply to a given problem instance, the reasoner can resort to reasoning from first principles.

3. Add heuristic search to model-based search. This can help manage the complexity of model-based reasoning and allow the reasoner to choose intelligently between possible alternatives.

The combination of model-based and case-based systems can:

1. Give more mature explanations to the situations recorded in cases.
2. Offer a natural first check against stored cases before beginning the more extensive search required by model-based reasoning.
3. Provide a record of examples and exceptions in a case base that can be used to guide model-based inference.
4. Record results of model-based inference for future use.

Heuristic and control in expert systems

- the inference engine
- the structure of the rules in the knowledge base, e.g., the order of the premises
- the contents of the rules

What good are KB systems?

Comparing human and artificial expertise

The Good News

Human Expertise

- Perishable
- Difficult to transfer
- Difficult to document
- Unpredictable
- Expensive

Artificial Expertise

- Permanent
- Easy to transfer
- Easy to document
- Consistent
- Affordable

Why do not entirely eliminate Human Experts?

Comparing human and artificial expertise

The Bad News

Human Expertise

- Creative
- Adaptive
- Sensory experience
- Broad focus
- Commonsense knowledge

Artificial Expertise

- Uninspired
- Needs to be told
- Symbolic input
- Narrow focus
- Technical knowledge

Fig 6.13 Blackboard architecture