

CHAPTER 1

Generic Tasks and Classificatory Diagnosis

Introduction

This paper reviews the development of a framework, using the Java programming language, for building expert systems that perform classification. The framework's foundation is provided by an architecture and philosophy for building knowledge-based systems called generic tasks [SG,CSRL,SM,AB]. Generic tasks are well-defined types of expert problem solving - classification is one such task.

The creators of this methodology developed several tools and expert systems related to the generic tasks including the Conceptual Structures Representation Language (CSRL) [CSRL]. This LISP tool is used for building problem solvers that perform hierarchical classification. For the project described in this paper, CSRL provided the basis for developing the object design for a framework of objects that collaborate to perform classification. This framework was implemented in Java using the Beans software component model to facilitate access by both integrators and end-users.

Chapter Overview

Chapter One introduces generic tasks, an approach to building knowledge-based systems. It describes a general architecture for the implementation of problem solvers for performing the generic tasks. It decomposes a complex task, diagnosis, into its constituent subtasks: classification, structured matching, and abductive assembly. It provides details about each subtask's specific architecture, knowledge representation, and

control strategy. It presents the strengths and weaknesses of the generic tasks approach and, finally, reviews an evolutionary enhancement called the task structure perspective.

1.1 What is a Generic Task?

The Laboratory for Artificial Intelligence Research (LAIR) at the Ohio State University identified several well-defined types of expert problem solving called generic tasks [SG]. This approach decomposes complex problem solving tasks into modular subtasks like classification, data retrieval, plan selection and refinement, state abstraction, and abductive assembly. It recognizes that particular ways of representing and using knowledge are ideally suited to the performance of a given task and, therefore, each generic task requires a specific organizational and problem solving structure [CSRL].

To build a knowledge-based system, two types of knowledge must be captured from human experts: domain knowledge and control knowledge [SM]. Domain knowledge includes the objects, relationships between them, and operators in a given problem area. Control knowledge refers to the determination and ordering of operations necessary to perform the task.

Generic tasks are considered task-specific knowledge-based techniques. Each solves a narrowly defined class of problem through the application of specialized control knowledge to specialized domain knowledge.

This focus on organizational structure and control strategy facilitated the development of a general architecture for building shells implementing the generic tasks [SG]. Each shell is built by deriving a task-specific architecture from the general one and contains the organizational and problem solving structure germane to the task. Populating the shell with domain knowledge provides a problem solver for the

corresponding task in that domain. For example, the task of classification could be used in the field of medical diagnosis to classify a patient's symptoms into one or more disease categories. The abductive task might then be used to select the hypothesis from the set of all plausible hypotheses that best explains the case data.

1.2 The General Architecture for Generic Tasks

The domain knowledge for a given task is spread amongst a community of specialists, each corresponding to particular concepts in the problem domain [SG]. This division of knowledge makes the problem solving process tractable – a specialist's knowledge is not relevant until the corresponding concepts are relevant. To facilitate their coordination during the problem solving process, the specialists are organized hierarchically. A specialist's children are said to be its subspecialists. A particular task may define additional relationships amongst its specialists.

A particular task utilizes a collection of strategies to control the problem solving process by utilizing the problem solving capabilities of individual specialists and the relationships between them [CSRL]. Each specialist decides locally which strategy to employ during the problem solving process. This leads to a more dynamic system than if that decision were made globally.

1.3 The Diagnostic Task

The diagnostic task involves identifying a set of plausible hypotheses for a set of data, and then selecting the best subset of these hypotheses based on which explained the data most satisfactorily [SG]. This task is considered a composition of four generic tasks:

classification, structured matching, data retrieval, and abductive assembly. Classification identifies the plausible hypotheses. Data retrieval supports the other subtasks by making inferences from incomplete or denormalized data in the case description. Abductive assembly selects the best subset of hypotheses.

The following sections discuss the three generic tasks comprising the framework developed for this project and discussed later in the paper: classification, structured matching, and abduction. A high-level task description and architectural overview are provided for each.

1.3.1 The Classification Task

In the classification task, a set of data describing a case is classified into one or more categories, each of which corresponds to a node in a pre-determined hierarchy [SG]. These nodes represent hypotheses about the case under consideration. Upper nodes correspond to the most general hypotheses while lower ones are more specific. Frequently, a classification hierarchy is used to classify malfunctions of some object [CSRL]. The case description contains detailed information about the object.

The classification hierarchy is implemented as a community of interacting specialists, each corresponding to a single hypothesis or concept [CSRL]. A specialist contains procedural knowledge and uses it to assess the likelihood of its hypothesis. This assessment produces a confidence value quantifying the specialists belief in the hypothesis. The hypothesis is established when the confidence value exceeds a specified threshold.

The strategy for the classification task is called establish-refine [CSRL]. When a specialist establishes, it refines the hypothesis by asking its subspecialists to establish. This process continues through the classification hierarchy. When a specialist rejects its hypothesis because the confidence value is too low, the search space below that node is effectively pruned.

Figure 1.1 shows a fragment of a classification hierarchy that classifies fuel problems [CSRL]. The following scenario might occur. The specialist “Fuel System Problems” is invoked. This specialist establishes and calls its subspecialists “Bad Fuel Problems” and “Fuel Mixture Problems”. “Bad Fuel Problems” rejects, eliminating its three subspecialists from consideration. Finally, the specialist “Fuel Mixture Problems” establishes and invokes its subspecialists.

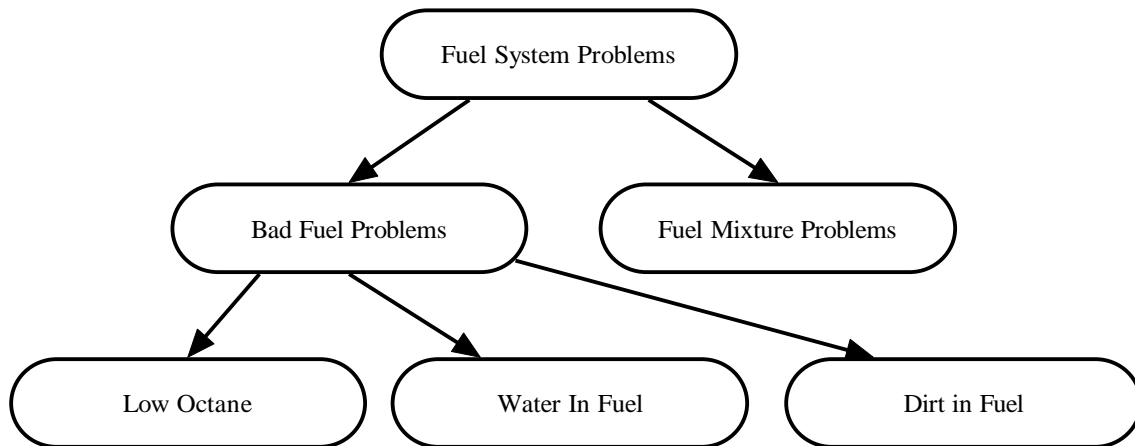


Figure 1.1 Fragment of a Diagnostic Hierarchy

1.3.2 The Structured Matching Task

Structured Matching is a subtask of the classification task that supports hypothesis assessment. The specialists in the classification hierarchy use structured matching to

evaluate the likelihood of their hypothesis based on the case description [SM]. The primary purpose of the structured matching task is to select a single choice from a small number of choices based on a given set of parameters by hierarchically matching features [SG].

Each node in a structured matching hierarchy represents a unique aspect in the decision. An intermediate node in the hierarchy contains the knowledge required to combine the results of its child nodes to produce a subdecision [SM]. The root node determines the final decision. Leaf nodes correspond to the decision parameters.

Figure 1.2 shows a structured matching hierarchy for evaluating the presence of biliary stones, taken from the MDX system described in Chapter Two [SM]. The presence of stones is based on the strengths of x-ray, clinical, and historical evidence. Each specialist contains knowledge to map parameters to its hypothesis. For example, the top specialist “Biliary Stone” contains rules like “if confidence value of Clinical Evidence specialist is +3 and Historical Evidence is T, then confidence in Biliary Stone is +3”. The “Clinical Evidence” specialist contains rules such as “if cholangitis is true, colicky pain is true, and nausea is true, confidence in clinical evidence is +3”.

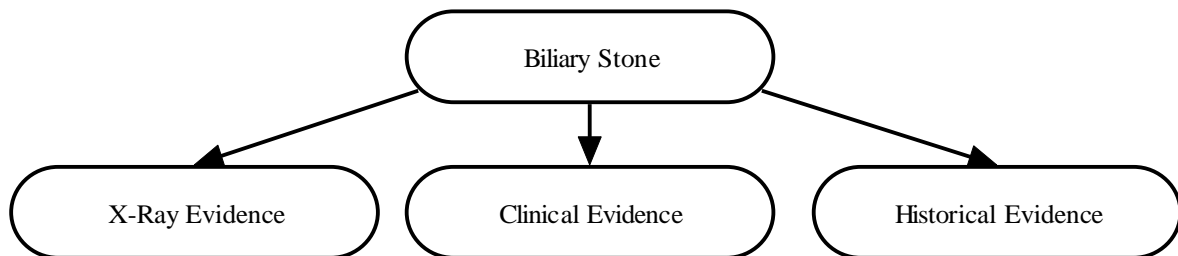


Figure 1.2 Structured Matching Hierarchy for Biliary Stone

1.3.3 The Abductive Assembly Task

The abductive task uses the classification task to generate a set of plausible hypotheses and an account of what each can explain. It then selects the best subset of these hypotheses by reasoning about which subset explains the data most satisfactorily [SG]. The abductive task supports a broader range of diagnostic problems than can be handled with classification alone.

Each plausible hypothesis identified by the classifier includes the following information: a case-specific description of the findings it explains; the confidence value produced by the specialist; a description of the fact obtained if the hypothesis is found to be true; and interactions between specialists [AB]. A specialist's plausibility confidence value is not an estimate of the probability of the truth of its hypothesis because this cannot be determined independently of interactions with other hypotheses. Rather, the confidence value says that the hypothesis might be true and is worth pursuing.

The strategy used for assembly is called means-ends search. Its goal is a complete explanation for the set of findings [AB]. A basic assembler handles mutually compatible hypotheses with overlapping explanatory capabilities. It can be enhanced to handle other types of hypothesis interaction, such as when one hypothesis is a subhypothesis of another; two hypotheses are mutually incompatible; or two hypotheses cooperate additively and overlap in what they can account for.

The means-ends search proceeds as follows [AB]. The process loops until there is nothing left to explain or nothing left that can be explained. Select an unexplained finding and utilize the classifier to identify the most plausible explanation for it, or mark it unexplainable if none is found. Add the selected finding to the growing composite

hypothesis, compute what the composite explains, compare that with what still needs to be explained, and determine the remainder.

Finally, an overview critic removes explanatorily superfluous parts from the tentative best explanation generated by the assembler [AB]. It identifies indispensable hypotheses, those that explain a finding that has no other plausible explanation, in the composite and removes those that are non-indispensable. The critic then invokes the assembler to rebuild a complete explanation and removes any remaining unnecessary information. At the end of this process the composite explains as much as possible with maximum plausibility and parsimony having used as a foundation the most likely hypotheses.

1.4 Strengths of the Generic Tasks Approach

Traditional knowledge-based systems employ task-independent knowledge representation schemes, such as predicates, sets, and subset relations. The task-specific representation schemes used by the generic tasks produce vocabularies of task related terms and relate the knowledge to how it will be used [SG]. Both of these qualities of task-specific representations facilitate knowledge acquisition.

Some of the generic task implementations exhibit computational advantages. Hypothesis generation by classification is linear in the number of hypotheses [SG]. Though the general problem of abductive assembly is NP-complete, knowledge in the correct form allows abductive problems to be solved in acceptable time.

Finally, the approach's focus on a general-purpose architecture for generic tasks provides numerous advantages. It facilitates the development of task-specific

architectures by specializing the general one. The emphasis on architecture allows system builders to focus primarily on content issues [SG].

1.5 Weaknesses of the Generic Tasks Approach

In order to perform diagnosis and design on a device, structural knowledge is required by both tasks. A common criticism of the generic tasks approach is that strict adherence to the task-specific architectures results in multiple representations of the structural knowledge [SG]. This problem is addressed by making structural knowledge representation via structure to behavior simulation a common subtask of the other two tasks.

Implementing systems as communities of interacting specialists simplifies system development – specialists are the “glue” which holds the system together via message-passing. The close tie between the task-level view and specialist-based implementation style is overly restrictive [SG]. LAIR addressed this problem by defining generic tasks using input-output task descriptions, appropriate strategies, and required knowledge.

Similarly, the strategies themselves were overly rigid [SG]. For example, establish-refine is a good general strategy for classification, but occasional a different control strategy is needed. Opportunistic run-time method selection provides more flexibility in integrating different generic task problem solvers.

Researchers at LAIR identified other difficulties with the approach [SG]. There are problems identifying the criteria for what qualifies a given task to be a generic task. The broad scope of some generic task problem solvers caused difficulties. For example, hypothesis assessment was originally part of the classification tool but eventually was

broken out to become the structured matching tool. Finally, confusion resulted from the ability to solve the same problem by combining the various generic tasks in different ways.

1.6 The Task Structure Perspective

LAIR enhanced the generic task approach with some philosophical changes embodied in their task-structure perspective [SG]. The primary change is that a task does not require a fixed method. For example, a diagnostic system might employ physiological model simulation in one problem instance and the heuristic match method in another. Furthermore, the task-structure perspective states that there is not a finite set of distinct methods for a task or, for that matter, a finite number of tasks.

1.7 Paper Overview

Chapter One introduced generic tasks and the task structure perspective. Chapter Two surveys tools and expert systems related to generic tasks. Chapter Three presents an object design for the classification framework. Chapter Four reviews the implementation of the framework using Java and Beans. Chapter Five demonstrates the framework's use for building and executing an expert classification system. Finally, Chapter Six considers potential research directions for the framework, including integrating the classification tool with tools for structured matching and abduction.