Task Specific Knowledge Based Systems and Their Application

Ahmed Kamel
North Dakota State University
Outline

- First Generation Expert Systems
- Task Specific Architectures
- Generic Task Approach:
  - Structured Matching
  - Hierarchical Classification
  - Routine Design
  - Functional Modeling
- Applications:
  - Engineering Design
  - Agricultural Farm Management
  - Adaptive Traffic Control
  - Computer Network Security Management
- Conclusions
First Generation Expert Systems

- Focus is on representation
  - e.g. Rule-based systems
- All knowledge represented in one form
  - e.g. If *condition* then *action*
- Control is separate from knowledge
Strengths

- Uniformity of representation
- Modularity of knowledge
- Change domains by changing rules
- Rules are easily expressed in English
- Rule trace provides explanation facility
Limitations

- No guidance in problem analysis
- No direct mechanism for expressing control knowledge
- Problems with scale-up
- Very hard to maintain: rules are not really independent
Lesson from First Generation

It is difficult to separate knowledge from its use.

- Knowledge Base
- Inference Engine
Task Specific Architectures

A task specific architecture *directly* supports specification of the knowledge needed to carry out a specified mapping of input to output.

*Examples of tasks include design, classification, ...*
Level of Generality

- specialized
  - too specific, lose ability for reuse of the approach/shell

- task specific
  - middle ground

- general purpose
  - too general purpose, lose constraints on knowledge and inference
Task Specific Architectures

- Several task-specific architectures exist to support the development of knowledge-based systems.
- Different architectures share a common philosophy, however they differ on the level of granularity of the tasks and on whether or not representation and use of knowledge should be separate.
- Furthermore, different architectures provide mechanisms for the representation of knowledge or its use or both.
Generic Task Approach

- Human reasoning is based on a set of generic problem solving types out of which complex reasoning is composed.
- An attempt is made to identify these problem solving types and to build software to support these types of problem solving.
- For each identified method, a representation template and a control regime are provided. The embodiment of which amounts to a language tailored for each specific GT that is identified.
Characteristics of a GT

- Information Processing Task (i.e. input/output relationship)
- Organization of Knowledge
- Control principles

e.g. Hierarchical classification, Routine Design, ....
Structured Matching

- A mechanism for imposing a structure on rules
- Organizes rules into small decision tables
- Each decision table produces a result using some ordered metric:
  - 0 … 5
  - Excellent, very good, good, neutral, poor
  - strongly match, match, neutral, against, strongly against
  - etc.
- Decision tables may refer to the results of other decision tables
### Structured Matching Example

<table>
<thead>
<tr>
<th>Var 1</th>
<th>Var 2</th>
<th>Table 2</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 5</td>
<td>= Important</td>
<td>&gt;= Match</td>
<td>Match</td>
</tr>
<tr>
<td>&lt; 2</td>
<td>= Marginal</td>
<td>?</td>
<td>against</td>
</tr>
<tr>
<td>?</td>
<td>?</td>
<td>?</td>
<td>Strongly against</td>
</tr>
</tbody>
</table>
Hierarchical Classification

- **Knowledge Organization**
  - *hierarchy* of specialists
  - each specialist is responsible for establishing a hypothesis
  - a subspecialist represents a more detailed hypothesis than its parent

- **Processing**
  - *establish* the high level hypothesis (typically using a structured matcher)
  - Ask subspecialists to *refine* to a more detailed hypothesis
HC - Example 1

Animal
- Mammal
- Bird
- Reptile
  - Flying
    - Robin
  - Non-flying
    - Pigeon
Classes of Design

- Creative Design
- Innovative Design
- Routine Design
Routine Design

A method for performing “Routine Design” pioneered by David Brown (WPI) for performing design under repetitive well understood situations
Information Processing Task

Design Requirements → Routine Designer → Design Specifications
Routine Design Specialists

Design Performed by a hierarchy of cooperating specialists

- **Specialist S1**
  - Plan
    - Planning Decision
    - Constraint
    - Call Specialist S2
  - Plan
    - Planning Decision
    - Constraint
    - Call Specialist S2, S3

- **Specialist S2**
  - Plan
    - Planning Decision
    - Constraint

- **Specialist S3**
  - Plan
    - Planning Decision
    - Constraint
Routine Design Plans

- Plans represent a step by step method for performing the design
- A plan may include:
  - the use of other specialists
  - tasks to be performed
Plan Sponsors

- Each plan is associated with a sponsor
- A sponsor utilizes a structured matcher to assign an applicability rating to its plan
- Applicability ratings may be:
  - perfect
  - suitable
  - neutral
  - rule-out
Plan Selectors

- Problem-solving agents available to specialists
- Use domain knowledge in consultation with plan sponsors to select a plan for execution
Design Tasks

- A task is an ordered collection of design steps
Design Steps

- A step is the basic problem solving unit in routine design
- A step utilizes a structured matcher to assign a value to a design attribute
Design Constraints

- A relation among design attributes that must be maintained
- Constraints can be applied at any point in the design process
- Failure of a constraint triggers a failure handler
Failure Handlers

Failure handlers may be:

- automatic: systematic backing up to a branching point
- knowledge-directed: use domain knowledge to “fix” the design
Functional Modeling

- A Method for reasoning about the functional behavior of systems (engineered or natural)
- Relies on hierarchically decomposing the system along its major functional components
- Functionality of any component is expressed in terms of the functionality of its subcomponents
Simple Example - Clothes Pin

- Arm
- Pivot
- Teeth
- Spring
Device Decomposition

- Clothes Pin
  - Arms
  - Pivot
  - Spring
  - Spring Assembly
  - Spring Attachment
Device Functions

- Functions of clothes Pin:
  - Open, close, hold

- Functions of Pivot:
  - transmit_force
Representing a Function

Function: Open
of device: Clothes_Pin
to Make: teeth_more_open
provided: applied_force > restoring_force
by: open behavior
Behaviors

- A behavior is a procedural implementation of a function
Behavior Representation

- Applied_force > restoring_force
  - Using Function transmit_force of Pivot
- Force on spring > restoring_force
  - By Knowledge of Newton’s second law
- Teeth_more_open
Applications
Design of Composite Materials

Domain of Problem:
The material design (materials selection and processing protocol design) for thermoset polymer composite materials.
Design Architecture

- Design Requirements
  - Case Library
    - Design Case, Attributes to alter
      - Design Modifier
        - Design Modification failed
          - Output Design
      - Routine Designer
        - No cases
          - Output Design
Problem: Multiple Designs Needed

- Market conditions (e.g. cost and availability) are very dynamic
- needs vary (lower cost vs. faster turnaround time)
- being a relatively new area, experimentation with alternatives is desirable

* A single design is not sufficient
Multiple Design Solution

Routine Designer → Multiple Designer
       ↓
          Family of Designs
       ↓
            Design Selector
Multiple Design

- Allow selection of multiple plans
- Allow assignment of multiple (alternative) values for design attributes
- Output: a tree of design with each path from the root to a leaf node representing a complete design
- Safeguards must be taken to guard against exponential growth
Extensions to Routine Design

- During plan selection: rank the plans and select the highest ranked ones
- During step execution: select all possible values, not just the first one
- Limiters at step and task levels to constrain the generated designs
- Redesign eliminated by explicitly specifying possible redesign values in the main design knowledge-base
Agricultural Systems

**Task:** The management of all aspects of an irrigated wheat farm in Egypt
Top Level Architecture

Farmer’s Circumstances

Farmer’s Preferences

Strategic Planning Module

season
Plan

Farmer’s Preferences
Strategic Planning

Irrigated Wheat Crop Planning Task

- Varietal Selection
- Planting Date
- Strategic Pest Management
- Preplant Tillage
- Planting Parameters
- Harvest
- Fertilizer/Water Regime
Problem Solving Architecture

Strategic Planning Module
Routine Designer
Top Level Controller
Plan: Select Variety, Planting Date, Pest Management, Preplant Tillage, Planting Parameters, Fertilization/Water, Harvest

- Varietal Selection Specialist
- Planting Date Specialist
- Strategic Pest Management Specialist
- Preplant Tillage Specialist
- Planting Parameters Specialist
- Routine Design Module
- HC Module
- Algorithmic Module
- Harvest Specialist
- Fertilizer/Water Regime Specialist
Lessons Learned

- A complex problem solving situation should be decomposed into a set of simpler problems.
- Each of the individual problems can be solved separately using an appropriate generic task method.
- The individual problem solvers should be integrated to accomplish the overall goal.
Weed Identification

- Weed Identification: looks like a classification problem but:
  - Difficult to describe weeds using text
  - Difficult to understand description

- Solution: Picture-Based Hierarchical Classification Tool
Hierarchical Classification with each specialist associated with a picture

Classification accomplished by the user using the pictures

Motivation:
- Some situations are difficult to describe using text
- Describing pictures makes it hard to understand
Example

- Select either broad leaf weeds or grasses
Example (cont.)

- For broad leaf weeds, select appropriate shape for cotyledon
Example (cont.)

- Select appropriate true leaf
Example (cont.)

- Select the matching seedling
Example (cont.)

- The flowering plant
Applications Under Development

- Adaptive Traffic Signal Control
- Computer Network Security Management
Traffic Signal Control

- Simulation-based optimizers are used to optimize the control of traffic signals under “normal” traffic conditions.
- Normal traffic conditions may include periodic variations such as weekday morning and afternoon rush hours.
- What happens when an exception occurs, such as:
  - An accident
  - The end of an event or a popular game at a stadium
  - …etc.
Adaptive Traffic Signal Control

- Using monitoring devices, traffic volumes at all directions are collected.
- Traffic volumes are fed into a hierarchical classification system.
- The observed pattern is classified as one of several pre-compiled patterns described by various increments at the different directions (a null pattern is allowed to allow for no corrective actions to be taken under “normal” conditions).
Corrective Actions

- When a deviation from normal is detected, a routine planner is invoked to modify the current signal pattern for the current traffic signal, and those downstream

- Corrective actions may include modifications to:
  - Signal durations
  - Signal splits
  - Phase shifts relative to upstream signals

- Routine Planners are designed using simulations on a large number of systematically generated traffic patterns
Task: given a new “sensitive” project, assign varying degrees of access to the project files to the different project personnel

A hierarchical classification system based on project sensitivity level, job classification, and role on project
Conclusions

- Extensions are often needed and implemented as new problems are tackled (Will we ever reach a steady state?)
- Complex problems requires collaborative effort of more than one problem solver
- What’s next?
  - Educational Tutorials?
  - Distributed collaboration?
  - Intelligent Agents?