Chapter Overview

Knowledge-Based Systems

Knowledge-Based Systems Structure
User’s / Developer’s / Tool Builder’s View

Knowledge-Based Tools
Shells

Productions
Production Rules, Grammars, Languages

Semantic Nets
Nodes / Objects, Arcs / Relationships

Frames
Slots, Fillers

Logic
Propositional and Predicate Logic

Chapter Review
Important Issues

End User’s View

individual using the final product

intelligent program
frequently black box, no comprehension of its internal functioning

user interface

• allows questions about the problem to be solved
• queries about particular decisions
• provides explanations of decisions
• displays the derived results, possibly graphically
• saves or prints the results
• usually based on menus, graphic displays, natural language
must be carefully matched to the target end users

data / knowledge base

• contains relevant problem-specific information
• describes all currently known facts
• some facts may have been derived internally

Structure

of Knowledge-Based Systems

internal structure
components and their interaction

perspectives
different points of view
Knowledge Engineer’s View

interaction with the domain expert

intelligent program

- knowledge base
  contains relevant, domain-specific problem solving knowledge
  may be of heuristic or algorithmic nature
  internal representation format (rules, semantic nets, frames, objects, …)
- inference engine
  derives additional knowledge from existing rules and facts
  must be suited for the chosen representation scheme and the nature of the problem
  constitutes the final product for the end user

development shell

- knowledge acquisition tool
  construction of the knowledge base editor, syntax check, consistency check, …
- test case data base collection of data from sample problems
  verifies the knowledge base after modifications
- developer’s interface similar to the end user's interface, but with additional features
  display of intermediate steps, breakpoints, traces, access to test procedures, …

Tool Builder’s View

set of tools for the knowledge engineer

intelligent program

- knowledge base
  choice of internal representation formats (rules, semantic nets, frames, objects, …)
- inference engine
  various inference mechanisms for the supported representation formats (forward / backward / bidirectional chaining, inexact reasoning, single / multiple solutions, correctness vs. performance, …)
  constitutes the final product for the knowledge engineer

development shell

- knowledge acquisition tool
  tools for the construction of the knowledge base editor, syntax check, consistency check, …
- test case data base tools for constructing and using test cases
- developer’s interface enhancements for development purposes
  display of intermediate steps, breakpoints, traces, access to test procedures, …

similar to the knowledge engineer’s view
Knowledge-Based Tools

shells: everything but domain knowledge

inductive shells
decision trees or rule sets
derived from example cases

rule-based shells
knowledge is expressed as If-Then rules
tools for editing, structuring, checking rules
Examples: CLIPS, Personal Consultant Plus

hybrid shells
integrate multiple knowledge representation paradigms and various reasoning methods
very powerful, but also quite complex

special purpose shells
designed for particular problem classes
dedicated representation and reasoning methods
Examples: G2, RTworks (real-time processes)

Knowledge

and its meaning

epistemology
study of knowledge:
nature, structure, origins

a priori knowledge
known to be true in advance of experience
does not require evidence for its validation

a posteriori knowledge
empirical, open to revision
requires evidence for its validation

Types of Knowledge

procedural
knowing how to do something
algorithm

declarative
statements that can be true or false
specification

tacit also: unconscious
can’t be expressed in language
skills

also other classifications of knowledge

knowledge + inference = expert system
analogous to N. Wirth’s expression
algorithms + data structures = program
Knowledge Hierarchy

**meta-knowledge**
knowledge about knowledge
selects applicable knowledge

**knowledge**
information items and their relationships
usually loosely structured

**information**
processed data

**data**
items of potential interest
usually rigidly structured

**noise**
irrelevant items, of no interest
often obscure data

---

Backus-Naur Form

**meta-language**
defines the syntax of a language
does not say anything about semantics
(meaning)

**grammar**
complete set of production rules defining a
language unambiguously

**parse tree** also: derivation tree
graphic representation of a sentence and its
derivation

very popular for the specification of the syntax of
(computer) languages

---

Semantic Nets

also: propositional net, associative nets
labeled, directed graph

**nodes**
stand for physical objects, concepts,
situations

**arcs** (links, edges)
represent relationships between nodes
classic AI representation technique
originally proposed by [Quillian, 1968] for the
description of human memory and language
understanding

Example:

```<sentence> ::= <subject> <verb> <end-mark>```
in semantic nets

**purpose**
basic structure for organizing knowledge
formal basis for inferences

**format**
basically unrestricted, any type of link can be defined

**common types**
- is-a an individual is an instance of a class
- a-kind-of relates an individual class to a parent class
- is defines the value of an attribute
- cause expresses causal knowledge

**inheritance**
characteristics of a node are duplicated in a descendent

structure for representing typical knowledge about objects

**extension of semantic nets**
nodes can have an internal structure

**purpose**
a frame represents related knowledge for a narrow topic

**slots and fillers**
slots define attributes, fillers contain values

**procedural attachments** to slots
procedures invoked in certain situations
if-needed, if-added, if-removal

**commonsense knowledge**
frames are very useful for causal and commonsense knowledge
very powerful and flexible, but sometimes inefficient and incorrect

and knowledge

**knowledge representation**
formal method to describe knowledge via logical sentences

**inference mechanism**
generally accepted rules of reasoning often with strict formal properties, e.g. correctness, completeness

manipulation of propositions

**knowledge representation**
logical variables represent propositions propositions can be either true or false logical connectives for constructing compound sentences

**inference**
specified by a calculus allows the evaluation of a sentence to true or false
limited ability to express knowledge not adequate for many statements about the world
Predicate Logic

manipulation of predicates and terms

predicates
express relationships between objects
terms
used for the specification of objects
- constants stand for one specific object
- variables represent currently unspecified objects
- functions map arguments (terms) from one domain to another

quantifiers
restrict the scope of variables

unification
computes proper substitutions for matching predicate logic expressions
much more powerful than propositional logic
still some restrictions in its basic form (first order predicate logic)

Chapter Review

Knowledge-Based Systems

Knowledge-Based Systems Structure
User’s / Developer’s / Tool Builder’s View

Knowledge-Based Tools
Shells: everything but domain knowledge

Productions
Production Rules, Grammars, Languages

Semantic Nets
Nodes / Objects, Arcs / Relationships

Frames
Slots, Fillers, Commonsense Knowledge

Logic
Propositional and Predicate Logic