Overview of Knowledge Representation and Reasoning

Tim Finin
University of Maryland
Baltimore County

January 2004

Some material adapted from Richard Fikes, Stanford.
Questions

- What’s the difference between data, information and knowledge?
  - Intensional vs. extensional information?
  - Particular vs. general information?
- What does it mean to know something?
  - Philosophers often define knowledge as “justified, true belief”
  - Early AI scientists considered appropriate use of knowledge to be a key
- How is knowledge created?
  - Via learning? By being told? By reasoning from exiting knowledge?
- How does our way of conceptualizing the world influence the way we think and act.
Knowledge Representation

“We base ourselves on the idea that in order for a program to be capable of learning something it must first be capable of being told it.

We shall therefore say that a program has common sense if it automatically deduces for itself a sufficiently wide class of immediate consequences of anything it is told and what it already knows.”

http://www-formal.stanford.edu/jmc/mcc59.html
Knowledge and reasoning

**Knowledge**
- The psychological result of cognition, i.e., of perception, learning and reasoning
- That which is or can be understood
- The wing wherewith we fly to heaven (Shakespeare)
- Knowledge differs from data or information in that new knowledge may be created from existing knowledge using inference

**Reasoning**
- Thinking that is coherent and logical
- Logical inference
- The process of creating new knowledge from existing knowledge
Knowledge Representation

- **Representation of knowledge**
  - Description of world of interest
  - Usable by machine to make conclusions about that world

- **Intelligent System**
  - Computational system
  - Uses an explicitly represented store of knowledge
  - To reason about its goals, environment, other agents, itself

- **Reasoning based on explicitly represented knowledge**

- **Working hypothesis**
  - Knowledge of the world -
    - Can be articulated
    - Used as needed
Sample Issues in KR

- What form is the knowledge to be expressed?
- How can a reasoning mechanism generate implicit knowledge?
- How can knowledge be used to influence system behavior?
- How is incomplete or noisy information handled?
- How can we represent and reason?
- How can practical results be obtained when reasoning is intractable?
KR&R - Knowledge Representation

- How information can be appropriately encoded and utilized in computational models of cognition

- Two primary areas of activity
  - Designing formats for expressing information
    - Mostly "general purpose" representation languages (e.g., first-order logic)
  - Encoding knowledge (knowledge engineering)
    - Mostly identifying and describing conceptual vocabularies (ontologies)

- Declarative representations are the focus of KR technology

- Knowledge that is domain-specific but task-independent
KR&R - Reasoning

- Computations methods for creating new knowledge and information from exiting knowledge
  - Very general methods, e.g., modus ponens
  - Task-specific methods, e.g., algorithms for planning, scheduling, diagnosis, constraint satisfaction
  - Methods for managing reasoning, e.g., hybrid reasoning, parallel processing

- Analysis of reasoning
  - Soundness, completeness, complexity, ...

- Methods for creating explanations of reasoning results
Expressiveness vs. tractability tradeoff

- How to express what we know
- How to reason with what we express
- “A Fundamental Tradeoff in Knowledge Representation and Reasoning”
  - H. Levesque, R. Brachman; in Readings in Knowledge Representation; R. Brachman and H. Levesque (eds); Morgan Kaufman; 1985.
KR and Data Base Research

- Both “represent” knowledge
- Data bases contain only “ground literals”
  - No disjunctions
  - No quantifiers
- Data base schema provide quantified information
- Deductive data bases include implications
- Data base concerns -
  - Efficient access and management of large data bases
  - Concurrent updating
- KR concerns -
  - Expressivity
  - Effective reasoning
Early History of KR (‘60’s - ‘70’s)

- Origins
  - Problem solving work at CMU and MIT
  - Natural language understanding
- Many ad hoc formalisms
- “Procedural” vs. “declarative” knowledge
- No formal semantics
  - Problems:
    - How do we assign “meaning”
    - How can we say that a computer “understands”?
Emerging Paradigms (‘70’s - ‘80’s)

- **Semantic nets**
  - Unstructured node-link graphs
  - No semantics to support interpretation
  - No axioms to support reasoning
  - “What’s in a Link: Foundations for Semantic Nets”

- **Frames**

- **Production rules**

- **Predicate logic**
Nodes and Arcs

- arcs define binary relationships which hold between objects denoted by the nodes.

Diagram:

- Sue
- John
- Max

Relationships:
- mother
- age
- wife
- husband

Examples:
- mother(john, sue)
- age(john, 5)
- wife(sue, max)
- age(max, 34)

...
Semantic Networks

- The ISA (is a) or AKO (a kind of) relation is often used to link a class and its superclass.
- And sometimes an instance and it’s class.
- Some links (e.g. haspart) are inherited along ISA paths.
- The semantics of a semantic net can be relatively informal or very formal
  - often defined at the implementation level

![Semantic Network Diagram]

- Animal
- Bird
- Robin
- Rusty
- Red
- Wing
Reification

- Non-binary relationships can be represented by “turning the relationship into an object”
- This is an example of what logicians call “reification”
  - reify v : consider an abstract concept to be real
- We might want to represent the generic give event as a relation involving three things: a giver, a recipient and an object,
  give(john, mary, book32)
Many semantic networks distinguish nodes representing individuals and those representing classes.

The "subclass" relation from the "instance-of" relation.
Emerging Paradigms (‘70’s - ‘80’s)

- Semantic nets
- Frames
  - Structured semantic nets
  - Object-oriented descriptions
  - Prototypes
  - Class-subclass taxonomies
- "A Framework for Representing Knowledge"
- Production rules
- Predicate logic
Semantic networks morphed into Frame Representation Languages in the 70’s and 80’s.

A Frame is a lot like the notion of an object in OOP, but has more meta-data.

A **frame** has a set of **slots**.

A **slot** represents a relation to another frame (or value).

A slot has one or more **facets**.

A **facet** represents some aspect of the relation.
Facets

- A slot in a frame holds more than a value.
- Other facets might include:
  - current fillers (e.g., values)
  - default fillers
  - minimum and maximum number of fillers
  - type restriction on fillers (usually expressed as another frame object)
  - attached procedures (if-needed, if-added, if-removed)
  - salience measure
  - attached constraints or axioms
- In some systems, the slots themselves are instances of frames
Example Class-Subclass Taxonomy

Author
Document
   Book
      Edited-Book
   Miscellaneous-Publication
      Artwork
      Cartographic-Map
      Computer-Program
      Multimedia-Document
      Technical-Manual
   Periodical-Publication
      Journal
      Magazine
      Newspaper
   Proceedings
   Technical-Report
   Thesis
      Doctoral-Thesis
      Masters-Thesis
Title
Class Edited-Book

- Defined in Ontology: Documents
- Source code: documents.lisp

Arity: 1
Documentation: An edited book is a book whose authors are known as editors.
Instance-Of: Class, Relation, Set
Subclass-Of: Book, Document, Individual, Individual-Thing, Thing ... 

Slots:

Has-Author:
Minimum-Slot-Cardinality: ≥ 1

Has-Editor:
Minimum-Slot-Cardinality: 1
Same-Slot-Values: Has-Author

Publication-Date-Of:
Slot-Cardinality: ≥ 1

Publisher-Of:
Slot-Cardinality: ≥ 1

Title-Of:
Slot-Cardinality: ≥ 1
Example Instance Frame

Instance Solving-Frame-Problem

- Defined in Ontology: My-documents
- Source code: my-documents.lisp

Documentation: Not supplied yet.

Has-Author:
  Minimum-Slot-Cardinality: Go !

Has-Editor: Murray-Shanahan
  Minimum-Slot-Cardinality: Go !
  Same-Slot-Values: Go Has-Author

Instance-Of: Edited-Book, Go Book, Go Bounded, Go Document, Go Individual, Go Individual-Thing, Go Thing

Number-Of-Pages-Of: 407

Publication-Date-Of: Year-1997
  Slot-Cardinality: Go !

Publisher-Of: Mit-Press
  Slot-Cardinality: Go !

Title-Of: Solving-The-Frame-Problem
  Slot-Cardinality: Go !
Description Logic

- There is a family of Frame-like KR systems with a formal semantics.
  - E.g., KL-ONE, LOOM, Classic, ...

- An additional kind of inference done by these systems is automatic classification
  - finding the right place in a hierarchy of objects for a new description

- Many current systems take care to keep the language simple, so that all inference can be done in polynomial time (in the number of objects)
  - ensuring tractability of inference
Emerging Paradigms ('70’s - ‘80’s)

- Semantic nets
- Frames
- Production rule systems
  - Situation-action rules
    - If (warning-light on) then (turn-off engine)
  - If-then inference rules
    - If (warning-light on) then (engine overheating)
    - If (warning-light on) then ((engine overheating) 0.95)
- Hybrid procedural-declarative representation
- Basis for expert systems
- Predicate logic
Production Systems

- The notion of a “production system” was invented in 1943 by Post.
- Used as the basis for many rule-based expert systems.
- Used as a model of human cognition in psychology.
- A production is a rule of the form:

  \[ C_1, C_2, \ldots, C_n \Rightarrow A_1 A_2 \ldots A_m \]

  **Left hand side (LHS)**

  Condition which must hold before the rule can be applied.

  **Right hand side (RHS)**

  Actions to be performed or conclusions to be drawn when the rule is applied.
Basic Components

- **Rules:** Unordered set of user-defined "if-then" rules.
  - Form: \( \text{if } P_1 \land \ldots \land P_m \text{ then } A_1, \ldots, A_n \)
  - The \( P_i \)s are facts that determine conditions when rule is applicable.
  - Actions can add or delete facts from the Working Memory.

- **Working Memory** -- A set of "facts" consisting of positive literals defining what's known to be true about the world.
  - Usually “flat tuples” like (age finin 45)

- **Inference Engine** -- Procedure for inferring changes (additions and deletions) to Working Memory.
  - Typically forward chaining
Typical CLIPS Rule

(defrule determine-gas-level ""
  (working-state engine does-not-start)
  (rotation-state engine rotates)
  (not (repair ?))
  =>
  (if (not (yes-or-no-p "Gas in tank?"))
   then (assert (repair "Add gas.")))
)

(defrule normal-engine-state-conclusions ""
  (declare (salience 10))
  (working-state engine normal)
  =>
  (assert (repair "No repair needed."))
)

(defrule print-repair ""
  (declare (salience 10))
  (repair ?item)
  =>
  (printout t crlf crlf)
  (printout t "Suggested Repair:")
  (printout t crlf crlf)
  (format t " %s%n%n%n" ?item))
Typical CLIPS facts

- Facts in most production systems are basically flat tuples.

- A simple extension supported by many is to allow simple templates using “slot-filler” pairs.

  (deftemplate engine
    (slot horsepower)
    (slot displacement)
    (slot manufacturer)
    (slot year))

- Matching slots in a template is order insensitive, as in:

  (engine (year 1998) (horsepower ?x))

  (engine (horsepower 250)
    (displacement 500) (year 1998))

(initial-fact)
(working-state engine unsatisfactory)
(charge-state battery charged)
(rotation-state engine rotates)
(repair "Clean the fuel line.")
(engine (horsepower 250)
  (displacement 409)
  (manufacturer ford))
Basic Procedure

While changes are made to Working Memory do:

- **Match** -- Construct the Conflict Set -- the set of all possible (rule, list-of-facts) pairs such that rule is one of the rules and list-of-facts is a subset of facts in WM that unify with the antecedent part (i.e., Left-hand side) of the given rule.

- **Conflict Resolution** -- Select one pair from the Conflict Set for execution.

- **Act** -- Execute the actions associated with the consequent part of the selected rule, after making the substitutions used during unification of the antecedent part with the list-of-facts.
**Rete Algorithm**

- The **Rete Algorithm** (Greek for “net”) is the most widely efficient algorithm for the implementation of production systems.
- Rete is the only algorithm for production systems whose efficiency is asymptotically independent of the number of rules.
- The basis for a whole generation of fast expert system shells: OPS5, ART, CLIPS and Jess.
Soar

- Soar is a production system developed initially at CMU and now used in many places.
- Soar stood for State, Operator And Result because all problem solving in Soar is regarded as a search through a problem space in which you apply an operator to a state to get a result.
- It’s also a general cognitive architecture for developing systems that exhibit intelligent behavior.
- See [http://ai.eecs.umich.edu/soar/](http://ai.eecs.umich.edu/soar/)
- Example:
  ```
  sp {hello-world
      (state <s> ^type state)
      -->
      (write |Hello World|) (halt)}
  ```
Emerging Paradigms (‘70’s - ‘80’s)

- Semantic Nets
- Frames
- Production rule systems
- Predicate calculus
  - Primarily first order logic
    "Everybody loves somebody sometime."
    \[(\text{forall } \text{p} \hspace{1cm} (\text{forall } \text{p}1) \hspace{1cm} (\text{exists} \, (\text{p2 } \text{t}) \hspace{1cm} (\text{and} \, (\text{Person } \text{p2}) \hspace{1cm} (\text{Time } \text{t}) \hspace{1cm} (\text{Loves } \text{p1 } \text{p2 } \text{t})))\]

- Resolution theorem proving
KR in the ‘90’s

- Declarative representations
  - Easier to change
  - Multi-use
  - Extendable by reasoning
  - Accessible for introspection

- Formal semantics
  - Defines what the representation means
  - Specifies correct reasoning
  - Allows comparison of representations/algorithms

- KR rooted in the study of logics
  - temporal, context, modal, default, nonmonotonic, ...

- Rigorous theoretical analysis
Description Logic

- There is a family of Frame-like KR systems with a formal semantics.
  - E.g., KL-ONE, LOOM, Classic, …

- An additional kind of inference done by these systems is automatic classification
  - Finding the right place in a hierarchy of objects for a new description

- Current systems take care to keep the language simple, so that all inference can be done in polynomial time (in the number of objects)
  - Ensuring tractability of inference
KR in the ‘00’s ??

- Web based systems
  - embedding knowledge on web pages
  - languages based on XML: OIL, RDF, DAML, OWL

- Driven by new classes of applications
  - Electronic commerce (e.g., product catalogues)
  - Information retrieval on the web
  - Web services

- Integration with conventional software
  - e.g., OO modeling tools like UML
  - e.g., reflection in Java

- Business rules?

- Ontologies !

- ??
Summary

- Real knowledge representation and reasoning systems come in several major varieties.
- These differ in their intended use, degree of formal semantics, expressive power, practical considerations, features, limitations, etc.
- Some major families are:
  - Logic programming languages
  - Theorem provers
  - Rule-based or production systems
  - Semantic networks
  - Frame-based representation languages
  - Databases (deductive, relational, object-oriented, etc.)
  - Constraint reasoning systems
  - Description logics