Knowledge Representation, Reasoning and Planning

CPSC 533

Christian Jacob
Dept. of Computer Science
Dept. of Biochemistry & Molecular Biology
University of Calgary
Knowledge Engineering

- Theory vs. Rich
- Poor vs. Rich

- Symbolic AI Systems
- Fuzzy Systems
- Neural Networks
- Genetic Algorithms
- Statistical Methods

Any of the methods; Best - Hybrid Systems
Knowledge Engineering

What we look at now
Knowledge Representation, Reasoning and Planning

1. Common Sense Knowledge
2. Knowledge Representation Techniques
3. Reasoning with Predicate Logic
4. Planning
Common Sense Knowledge
CSK&

“It is perhaps paradoxical that AI scientists find the very subjects that are easy for ten year old humans more refractory to AI methods than are subjects that experts must study for years.”

Nilsson, 1998, p. 301
CSK Examples

• A liquid fills the shape of a cup.
• If you drop an object, it will fall.
• People don’t exist before they are born.
• Fish live in water and will die if taken out.
• People typically sleep at night.
CSK Examples 2&

- Get under shelter when it’s raining.
  - No need for knowledge about low pressure systems.
- Don’t spill coffee out of your cup.
  - No need for advanced hydrodynamics.
- When a battery gets low, charge it.
  - No need for electrochemical theory.
CSK vs. Scientific Knowledge

• Common Sense Knowledge
  • Adequate for many of the things that humans do
  • How many rules are needed by a system capable of general human level intelligence?
  • “... there is probably no elegant, effortless way to obtain this immense knowledge base. Rather, the bulk of the effort must at least initially be manual entry of assertion after assertion.” Doug Lenat, CYC
CSK vs. Scientific Knowledge

• SK gradually separated itself from CSK, seeking more precise descriptions of our world.

• Expert knowledge can be compartmentalized.

• CSK is needed to make expert systems more useful.
CSK vs. Scientific Knowledge

• CSK is required to understand natural language.

• Analogies and metaphors are part of our reasoning.
  • Example: spacial metaphors
    Cleanliness is next to godliness.
    Mary’s salary is above John’s.
Importance of Common Sense Knowledge

“An expert system possessing a basic commonsens conceptualization of the world might already be well equipped to expand its knowledge base with littl augmentation or revision.”

Knowledge Representation, Reasoning and Planning

1. Common Sense Knowledge
2. Knowledge Representation Techniques
3. Reasoning with Predicate Logic
4. Planning
Knowledge Representation Techniques

a. Taxonomic Knowledge
b. Semantic Networks
c. Frames
Taxonomic Knowledge

- Entities of both common sense and expert knowledge domains can be arranged in hierarchical structures
  - to organize and
  - to simplify reasoning.
Taxonomic Knowledge

- A hierarchical representation implicitly encodes facts of the form
  - “X is a P.”
  - “All Ps are Qs.”
  - “All Qs are Rs.”
Taxonomic Knowledge Example

• Facts and their formal representations predicate calculus:

  • “Snoopy is a laser printer.” \( \text{Laser\_printer}(\text{Snoopy}) \)

  • “All laser printers are printers.”

    \( (\forall x)\text{[Laser\_printer}(x) \land \text{Printer}(x)] \)

  • “All printers are machines.”

    \( (\forall x)\text{[Printer}(x) \land \text{Office\_machine}(x)] \)
Knowledge Representation Techniques

a. Taxonomic Knowledge

b. Semantic Networks

c. Frames
Three kinds of arcs:

- **Subset arcs** *isa* links
- **Set membership arcs** *instance* links
- **Function arcs**
Semantic Network

"R2D2 is an office machine."
Semantic Network

“Snoopy is connected to the wall outlet.”
Knowledge Representation Techniques

a. Taxonomic Knowledge
b. Semantic Networks
c. Frames
Frames

Printers
  subset_of: Office_machines
  superset_of:
    { Laser_printers, Ink_jet_printers }
  slots
  energy_source: Wall_outlet
  creator: John_Jones
  date: 16_Aug_91

Frame name
Slot name
Meta information
Slot fillers
Knowledge Representation, Reasoning and Planning

1. Common Sense Knowledge
2. Knowledge Representation Techniques
3. Reasoning with Predicate Logic
4. Planning
Logic System

• A logic system consists of four parts:
  • Alphabet: a set of basic symbols from which more complex sentences are made.
  • Syntax: a set of rules or operators for constructing expressions sentences&
  • Semantics: for defining the meaning of the sentences
  • Inference rules: for constructing semantically equivalent but syntactically different sentences
Predicate Logic

• The following types of symbols are allowed in *predicate logic*:
  • Terms
  • Predicates
  • Connectives
  • Quantifiers
Predicate Logic

• Terms:
  • Constant symbols: symbols, expressions, or entities which do not change during execution e.g., true / false
  • Variable symbols: represent entities that can change during execution
  • Function symbols: represent functions which process input values on a predefined list of parameters and obtain resulting values
Predicate Logic

• Predicates:
  • *Predicate* symbols: represent true/false type relations between objects. Objects are represented by constant symbols.
Predicate Logic

• Connectives:
  • Conjunction
  • Disjunction
  • Negation
  • Implication
  • Equivalence
  • ... same as for propositional calculus
Predicate Logic

- **Quantifiers:**
  - valid for variable symbols
  - **Existential quantifier:** “There exists at least one value for $x$ from its domain.”
  - **Universal quantifier:** “For all $x$ in its domain.”
First Order Logic

• First order logic allows quantified variables to refer to objects, but not to predicates or functions.

• For applying an inference to a set of predicate expressions, the system has to process matches of expressions.

• The process of matching is called unification.
In PROLOG, a quantifier free Horn(clausal) notation is adopted.

\( \forall x, \text{Human}(x) \Rightarrow \text{Mortal}(x) \)

- \( \text{mortal}(X) :- \text{human}(X). \)

- \( \text{Human}(\text{Socrates}) \)
  - \( \text{human}(\text{Socrates}). \)
PROLOG: Horn Clauses

- A rule
  - \( A_1, A_2, ..., A_n \Rightarrow B \)
- as a Horn clause has the following form:
  - \( B \leftarrow A_1, A_2, ..., A_n \)
- The goal \( B \) is true if all the sub goals \( A_1, A_2, ..., A_n \) are true.
- \( A_1, A_2, ..., A_n \) are predicate expressions.
PROLOG: Facts

• A fact is represented as a literal clause with the right side being the constant \textit{true}.

• Therefore, a clause representing a fact is always true.

• Example:
  • \texttt{human(Socrates) :- true.}
  • \texttt{human(Socrates).}
Knowledge Representation

• In PROLOG knowledge is represented as a set of clauses with
  • premises right side of the clause and
  • one conclusion left side

• AND is denoted by the character “,”.

• OR is denoted by the character “;”.

• NOT is denoted by “¬”.
PROLOG Architecture

Standard Inference

Goal

User defined
Knowledge base
(Clauses)

The program

FACTS

(yes, fail, solutions...)

CPSC 533   Artificial Intelligence: An Introduction
Christian Jacob, University of Calgary
Family Example: *Facts*

- father(John, Mary).
- father(Jack, Andy).
- father(Jack, Tom).
- mother(Helen, Jack).
- mother(Mary, Jilly).
- grandfather(Barry, Jim).
Family Example: Rules

grandfather(X,Y)
   :- father(X,Z), parent(Z,Y).

grandmother(X,Y)
   :- mother(X,Z), parent(Z,Y).

parent(X,Y) :- mother(X,Y).

parent(X,Y) :- father(X,Y).
PROLOG Sample Dialogue

Goal?– grandfather(John, Jilly).
  yes

Goal?– grandmother(Helen, X).
  2 solutions
    X = Andy; X = Tom

Goal?– grandmother(X,Y), X = Y.
  fail
Goal: \text{grandfather}(John, Jilly). \\

\text{yes}
PROLOG Sample Inference

Goal?-- grandmother(Helen, X).

2 solutions: X = Andy or X = Tom
Knowledge Representation, Reasoning and Planning

1. Common Sense Knowledge
2. Knowledge Representation Techniques
3. Reasoning with Predicate Logic
4. Planning
Planning

a. Simple Planning Agents
b. From Searching to Planning
c. Planning through Logical Inference
d. Representations for Planners
e. Example: Partial Order Planner POP&
Simple Planning Agent

• A **planning agent** uses its percepts of the world state to form a **model** of the world.

  • The agent first generates a **goal** to achieve.

  • Then the agent constructs a **plan** to achieve it from its current state.

  • Once it has a plan, it keeps executing it until the plan is finished.

  • Then the agent begins again with a new goal.
A Simple Example Plan

• Suppose we are at home and we want to go to school, and we must take the bus to do so.

• A simple plan would likely be:
  • **Start state**: We are at home.
  • **Actions**:
    - Go to the bus stop.
    - Get on the bus. ...
A Simple Example Plan

- **Actions**
  
  Go to the bus stop.
  
  Get on the bus.
  
  Ride the bus to school.
  
  Get off the bus.
  
  Walk from the bus stop to your classroom.

- **Goal state:** We are at school.
Planning

a. Simple Planning Agents

b. From Searching to Planning

c. Planning through Logical Inference

d. Representations for Planners

e. Example: Partial Order Planner POP&
From Searching to Planning

• **Actions**
  
  • Actions are described by programs that generate successor state descriptions.

• **States**
  
  • State representations are used for
    
    successor generation,

    heuristic function evaluation, and

    goal testing.
From Searching to Planning

• **Goals**
  - The goal test and heuristic function are the only way to test for a goal state.

• **Plans**
  - A solution is a special plan—an unbroken sequence of actions from an initial state to a goal state.
Utilizing First Order Logic

• First order Logic
  • **States** and **goals**: sets of sentences
  • **Actions**: logical descriptions of preconditions and effects.

• Consequently, we get direct connections between **states** and **actions**.
States and Actions: Example

• Use first order logic to select an appropriate action for a plan:
  • Goal: \( \text{Have Milk} \& \)
  • Database: \( \text{Buy}\ x \Leftrightarrow \text{Have}\ x \& \)
  • Chosen action by inference \( \& \text{Buy Milk} \& \)
Planning Order

• A planner is free to add actions to the plan wherever they are needed.

  • **Example**: Choose *Buy Milk* even before knowing where and how to get *Milk*.

  • There is no necessary connection between the order of planning and the order of execution.

  • Representing states as sets of logical sentences is essential for making this freedom possible.
Planning

a. Simple Planning Agents

b. From Searching to Planning

c. Planning through Logical Inference

d. Representations for Planners

e. Example: Partial Order Planner  POP&
Logical Inference Planning

• Situation Calculus

  • **Initial state**: an arbitrary logical sentence about a situation $S$.

    At $Home$, $S \land \neg$Have $Milk$, $S \land \neg$Have $Bananas$, $S \land \neg$Have $Drill$, $S$

  • **Goal state**: a logical query asking for suitable situations.

    $\exists s$ At $Home$, $s \land$ Have $Milk$, $s \land$ Have $Bananas$, $s \land$ Have $Drill$, $s$
Logical Inference Planning

- **Operators**: a set of descriptions of actions.

\[ \forall a, s \text{ Have } Milk, \textbf{Result } a, s \Leftrightarrow \]

\[ a = \text{Buy Milk} \land \text{At Supermarket}, s \land \]

Have Milk, s \land a \neq \text{Drop Milk} \land

**Result** a, s \& the situation resulting from executing action a in situation s.
Logical Inference Planning

- **ResultSeq** $k$, $s$: the situation resulting from executing the sequence of actions $k$, starting in situation $s$.

$$V \; s \; \text{ResultSeq} \quad s \not= s$$

$$V \; a, p, s \; \text{ResultSeq} \quad a \mid p \; s \not= \text{ResultSeq} \; p, \; \text{Result} \quad a, s \not= s$$
Shopping Example

• A solution to the shopping problem is a plan \( p \) that, when applied to the start state \( S \), yields a situation satisfying the goal query.

• Therefore, we search for a \( p \) such that

  • At \( \text{Home} \), ResultSeq \( p, S \& \land \)\n
  Have \( \text{Milk} \), ResultSeq \( p, S \& \land \)

  Have \( \text{Bananas} \), ResultSeq \( p, S \& \land \)

  Have \( \text{Drill} \), ResultSeq \( p, S \& \)
Shopping Example

The plan $p$

- Go Supermarket&Buy Milk&Buy Bananas&
  Go HardwareStore&Buy Drill&Go Home&

is a solution for the goal

- At Home, ResultSeq $p$, $S\land$
  Have Milk, ResultSeq $p$, $S\land$
  Have Bananas, ResultSeq $p$, $S\land$
  Have Drill, ResultSeq $p$, $S\land$
Planning

a. Simple Planning Agents
b. From Searching to Planning
c. Planning through Logical Inference
d. **Representations for Planners**
e. Example: Partial Order Planner POP&
Representing Actions

• An action, embedded in a plan
  • **Action description**: name for a possible action
  • **Precondition** for an operator: a conjunction of atoms positive literals stating what must be true before the operator can be applied.
  • **Effect** of an operator: a conjunction of literals positive or negative that describe how the situation changes when the operator is applied.
Operator Example

- \( Op \) ACTION: Go \textit{there}\&

PRECOND: \( \text{At} \textit{here}\& \land \text{Path} \textit{here, there}\& \)

EFFECT: \( \text{At} \textit{there}\& \land \neg \text{At} \textit{here}\& \)

\[ \text{At}(\textit{here}), \text{Path}(\textit{here, there}) \]

\[ \text{Go}(\textit{there}) \]

\[ \text{At}(\textit{there}), \neg \text{At}(\textit{here}) \]
Total / Partial Order Plans

• Least Commitment Principle  As a planner, don’t make a decision until you have to.
  
  • If a decision is not important at a given stage, leave it undecided.

    Allows for greater flexibility in the planning process.

    Results in less backtracking.

  • This leads to a partial order of the action sequences within a plan.
Example: Putting Shoes On

Partial Order Plan:

Start

Left Sock

Right Sock

LeftSockOn, RightSockOn

Left Shoe

Right Shoe

LeftShoeOn, RightShoeOn

Finish

Total Order Plans:

Start

Start

Start

Start

Start

Start

Start

Start

Right Sock

Right Sock

Left Sock

Left Sock

Right Sock

Right Sock

Right Shoe

Left Shoe

Right Shoe

Left Shoe

Left Shoe

Left Shoe

Right Shoe

Right Shoe

Left Shoe

Right Shoe

Left Shoe

Left Shoe

Right Shoe

Right Shoe

Left Shoe

Right Shoe

Left Shoe

Right Shoe

Finish

Finish

Finish

Finish

Finish

Finish

Finish

Finish

Finish
Data Structure for Plans

• A set of plan steps. Each step is one of the operators.

• A set of step ordering constraints.
  • Each constraint is of the form $Si < Sk$.
  • “Step $Si$ must occur some time before step $Sk$.”

• A set of variable binding constraints of the form $v = x$. $v$: variable, $x$: constant / variable.

• A set of causal links.
Data Structure: Shoe Example

- **Plan STEPS:**

  $S_1$: *Op ACTION: Start*,

  $S_2$: *Op ACTION: Finish*,

  **PRECOND:** $RightShoeOn \land LeShoeOn$

  **ORDERINGS:** $S_1 < S_2$ 

  ```
  Start
  \arrow{Initial State}
  \arrow{Goal State}
  \arrow{Finish}
  ```

  ```
  Start
  \arrow{LeftShoeOn}
  \arrow{RightShoeOn}
  \arrow{Finish}
  ```
Causal Links

• Causal links are formed based on the fulfilling of preconditions.

• A causal link is created when a given state $S_1$ contains a precondition $c$ for state $S_2$.
  
  - $S_1 \xrightarrow{c} S_2$
  
  - “$S_1$ achieves $c$ for $S_2$.”

• Causal links are protected.
Protecting Threatened Causal Links

$S_1$ $c$ $S_2$ $\neg c$ $S_3$

When linearizing a plan, threats that may delete a precondition are ordered either
- before the first state (demoted) or
- after the second (promoted) when linearizing the plan.

This prevents removing the link.

$S_3$ threatens a condition $c$ established by $S_1$ and protected by the causal link from $S_1$ to $S_2$. 
Protecting Threatened Causal Links

$S_3$ threatens a condition $c$ established by $S_1$ and protected by the causal link from $S_1$ to $S_2$. 
Planning

a. Simple Planning Agents

b. From Searching to Planning

c. Planning through Logical Inference

d. Representations for Planners

e. Example: Partial Order Planner (POP)
Partial Order Planner POP&

- POP uses *regressive* planning to create a solution.

  1. Start with a minimal partial plan and extend it at each step by achieving the precondition of a state $s$.

  2. Set up causal links and eliminate threats.

  3. If an applicable operator cannot be found, backtrack to previous choice point.

- POP is *sound* truth preserving and *complete*, i.e., it always finds a solution if one exists.
POP Shopping Example

At(Home) Sells(SM,Banana) Sells(SM,Milk) Sells(HWS,Drill)

Have(Drill) Have(Milk) Have(Banana) At(Home)

Start

Finish
POP Shopping Example

Op(ACTION: Buy(x), PRECOND: At(store) \land Sells(store, x), EFFECT: Have(x))
POP Shopping Example

\[ Op(\text{ACTION}: \text{Buy}(x), \text{PRECOND}: \text{At(store)} \land \text{Sells(store, x)}, \text{EFFECT}: \text{Have}(x)) \]
POP Shopping Example

$Op(\text{ACTION: Buy}(x), \text{PRECOND: } \text{At}(\text{store}) \land \text{Sells}(\text{store}, x), \text{EFFECT: Have}(x))$
POP Shopping Example

\[\text{Op(ACTION: Start, EFFECT: At(Home) \land Sells(HWS, Drill) \land Sells(SM, Milk) \land Sells(SM, Banana))}\]
POP Shopping Example

Op(ACTION: Start, EFFECT: At(Home) ∨ Sells(HWS, Drill) ∨ Sells(SM, Milk) ∨ Sells(SM, Banana))
POP Shopping Example

A partial plan achieving $At$ preconditions of the three $Buy$ actions.
POP Shopping Example

A flawed plan, getting the agent to the HWS and SM.
⇒ Add causal link protection
POP Shopping Example

Causal link protection.
POP Shopping Example

Again: Causal link protections are needed!
POP Shopping Example

Causal link protection.
POP Shopping Example

A Solution Plan

→: causal link protection
References

