Section 3: Knowledge Representation

- Basis of each AI concept or system!
- Representation without processing makes no sense (therefore we started with knowledge processing)
- Same knowledge can be represented very differently:
  - Spectrum: computer friendly - human friendly
  - Levels of abstraction
  - Different views on problem
  - Different processing techniques

  Note: transformations are possible!

Syntax and Semantics

- Similar to programming languages, in knowledge representation we have to look at syntax and semantics of a representation approach
- Syntax: What symbols, data types, etc. are allowed; sorts, number of arguments (multiplicity) and so on? What symbols have special meaning (and therefore have to be used with this meaning in mind)?
- Semantics: What do the symbols mean, what has knowledge processing to accomplish?

☞ we have to look at both

3.1 Logics

- Considered by humans as the knowledge representation (and processing) method of computers
- Clear mathematical foundation: syntax describes formulas; axioms what is considered true; inference rules how to get other true formulas
- Many different kinds of logics
- Meaning of a formula usually not easy to determine by humans (rather formal semantics)

General Definitions (I)

Syntax:

Terms (without sorts): $\mathcal{F}$ = F (function symbols) $\cup$ V (function variables); σ(f) $\in \mathbb{N}$ multiplicity

Term($\mathcal{F}$) recursively defined by

f $\in \mathcal{F}$ with σ(f) = n and $t_1,\ldots,t_n \in$ Term($\mathcal{F}$) then

$f(t_1,\ldots,t_n) \in$ Term($\mathcal{F}$)

Atoms: $\mathcal{P}$ = P (predicate symbols) $\cup$ PI (interpreted predicate symbols) $\cup$ PV (predicate variables); σ(A) $\in \mathbb{N}$ multiplicity

Atom = Atom($\mathcal{P}$,Term($\mathcal{F}$)) = {$A(t_1,\ldots,t_n)$ | A $\in \mathcal{P}$, σ(A) = n, $t_1,\ldots,t_n \in$ Term($\mathcal{F}$) }

General Definitions (II)

Formulas: sets J (Junctors), Q (Quantifiers); σ(★) $\in \mathbb{N}$ multiplicity

Form = Form(J,Q,Atom($\mathcal{P}$,Term($\mathcal{F}$))) recursively def.

- A $\in$ Form, if A $\in$ Atom
- ★ $\in$ J, σ(★) = n, $A_1,\ldots,A_n \in$ Form ☞ ★$(A_1,\ldots,A_n) \in$ Form
- $\square \in$ Q, A $\in$ Form, $x_1,\ldots,x_n \in$ V $\cup$ PV ☞ $\square x_1,\ldots,x_n$.A $\in$ Form

General Definitions (III)

Adding Meaning:

Interpretation: Given Form(J,Q,Atom($\mathcal{P}$,Term($\mathcal{F}$))) , set D of objects (domain), set W of truth values Interpretation I

- Assigns to each f $\in \mathcal{F}$ a function over D and to each A $\in \mathcal{P}$ a predicate over D in the truth values of W
- Assigns to each ★ $\in$ J, σ(★) = n, a function $W^n \to W$

General Definitions (IV)

- Assigns to each □ ∈ Q a combination rule for truth values in W, such that I(□x_1,…,x_n.B) is determined by combining the truth values of all the formulas that are generated by substituting the variables x_1,…,x_n in B by arbitrary (but fitting) combinations of functions and/or predicates over D

General Definitions (V)

All together:

Logic: Form, $\mathcal{I}$ = {I_1,I_2,…} a set of interpretations with

- I_i(★) = I_j(★) for all i,j and ★ ∈ J
- I_i(□) = I_j(□) for all i,j and □ ∈ Q
- I_i(A) = I_j(A) for all i,j and A ∈ PI

☞ (Form, $\mathcal{I}$) logic

Note: there are many different logics!

Working with a Logic

Calculus:

(Form, $\mathcal{I}$) logic to W. Ax ⊆ Form set of Axioms; R set of rules:

(Ax,R) calculus to (Form, $\mathcal{I}$) and w ∈ W, if

B ∈ Form with I(B) = w for all I ∈ $\mathcal{I}$ can be transformed into subset of Ax by applying the rules of R

Note: this still allows for different search models using the calculus rules!

3.1.1 Propositional logic

General idea:

- Formulas describe combinations of statements (propositions) that are either truth or false and this way build statements themselves.
- No parameterized statements!
- Basis of the logics of gates, circuits and micro chips

Basic knowledge structures

- Term($\mathcal{F}$) = Ø
- $\mathcal{P}$ = P and □(A) = 0 for all A ∈ P (elements of P often called propositional variables; very unfortunate naming!)
- J = {¬,∧,∨,→,↔}, Q = Ø
- W = {true, false}
- I = all possible interpretations (Interpretation here is an assignment of truth values to the propositions in P)

Semantics

- Look for tautologies, i.e. formulas that are interpreted to true by all I ∈ $\mathcal{I}$
- I(¬p) = true, if I(p) = false; false else
- I(p∨q) = true, if I(p) or I(q) = true; false else
- I(p∧q) = true, if I(p) and I(q) = true; false else
- I(p→q) = false, if I(p) = true and I(q) = false; true else
- I(p↔q) = true, if I(p) = I(q); false else

How to get knowledge into the representation structure

- assign predicate symbols to simple positive statements
- Connect them to form complicated statements
- But be careful: “tertium non datur”
  - The car is green =: p
  - The car is red =: q
  - We need in addition:
    q □ □p

Discussion

✚ decidable, but NP complete
□ not very expressive
□ knowledge bases get very large

And what about processing data?

- Calculus used in most (best) systems:
  Davis-Putnam (working on clauses; special case of Modelelimination)
- Each formula can be transformed into equivalent set of clauses (remember: formula with J = {□, })
  - "defining" equations for □ and □
  - DeMorgan's laws to move negation inward
- For deciding tautologies, we use and-tree-based search
- For testing for satisfiability, we see clauses as constraints and use or-tree-based search

Example

- Represent the following statements in propositional logic:
  - A Ferrari is a red car.
  - Red cars are fast cars.
  - Bad cars are slow cars.
- Show that the following statement is a logical consequence of the statements above:
  - A Ferrari is a good car.