Artificial Intelligence 1

Roman Barták
Department of Theoretical Computer Science and Mathematical Logic

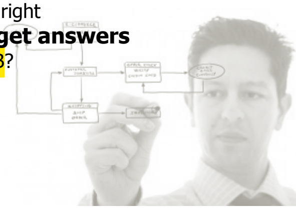
Knowledge Engineering

Knowledge engineering

- **Knowledge engineering** deals with the process of knowledge-base construction.
- A **knowledge engineer** is someone who:
  – **investigates** a particular domain
    - How the things work?
    - This is usually done in co-operation with a problem expert.
  – **learns** what **concepts** are important in that domain
    - Which will be the queries asked and what do we need to find answers?
  – **creates a** formal **representation** of the objects and relations in the domain
    - How to encode facts and axioms so the computer can do inference?

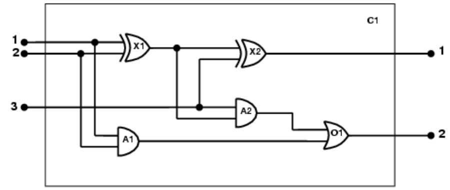
Knowledge-engineering process

1. **identify the task**
   – What is the range of questions?
   – Wumpus: action selection or asking about the contents of the environment?
2. **assemble the relevant knowledge** **(knowledge acquisition)**
   – How does the domain actually work?
   – Wumpus: what does it mean to feel stench and breeze?
3. **decide on a vocabulary of predicates, functions, and constants**
   – How to translate domain-level concepts to logic-level names?
   – Wumpus: is a pit an object or a function of the square?
   – The result is an **ontology** of the domain (vocabulary of notions).
4. **encode general knowledge about the domain**
   – Which axioms hold in the domain?
   – Wumpus: breeze means a pit in the neighbourhood square
5. **encode a description of the specific problem instance**
   – What is the current state of the world?
   – Wumpus: the agent is at square (1,1) looking to the right
6. **pose queries to the inference procedure and get answers**
   – How does the inference procedure operate on our KB?
   – Wumpus: is cell (2,2) really safe?
7. **debug the knowledge base**
   – What is missing in the knowledge base?
   – Wumpus: there is a single wumpus in the cave



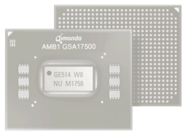
KE process: identify the task

Digital circuits
– 1 and 2 are input bits, 3 is a carry bit
– 1 is output bit for sum, 2 is output bit for carry



What is important in the domain?
– Does the circuits add properly?
– If the inputs are know, what is the output?
– If desired output is given, what should be the input?

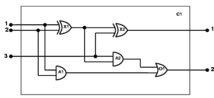
Different queries may require different knowledge!
– What is cost of the circuit?
– What is the size of circuit?
– How much energy does the circuit consume?



KE process: knowledge acquisition

What do we know about digital circuits?

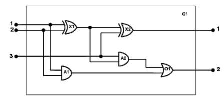
- circuits are composed from wires and gates
- signals 0 and 1 flow along wires
- signals flow to the input terminals of gates
- each gate produces signal on the output terminal
- there are four types of gates: AND, OR, XOR, NOT
- circuits have input and output terminals
- wires are used just as connections between terminals
- signal delay, energy consumption, shape of gates are not assumed



KE process: vocabulary

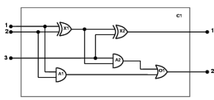
What constants, predicates, and functions?

- we describe circuits, gates, terminals, signals, and gates
  - **gates** are denoted by **constants X_1, X_2, A_1, …**
  - the behaviour of each **gate** is determined by its **type**
    - we will used constants AND, OR, XOR, NOT
    - types of gates are described by **functions Type(X_1) = XOR**
    - We can also predicates Type(X_1,XOR) or XOR(X_1)
      - Beware! We will also need axioms to describe uniqueness of the gate type.
  - **terminals** of gates can also be names by constants (X_1In_1, …), but then we need to connect them to gates
    - it is better to use **functions In(1, X_1), …**
  - **wires** can be described by **predicates**
    - **Connected(Out(1, X_1),In(1, X_2)), …**
    - Beware! We connect the terminals not the gates.
  - **signals** at terminals are determined by a **function**
    - **Signal(g) = 1**

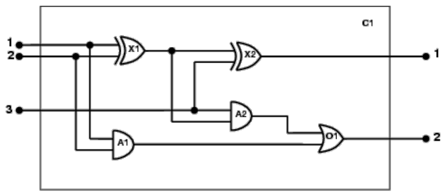


KE example: general knowledge

- **If two terminals are connected, then they have the same signal.**
  - $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Signal(t_1) = Signal(t_2)
- **The signal at every terminal is either 1 or 0.**
  - $\forall t$ Signal(t) = 1 $\vee$ Signal(t) = 0
  - $1 \neq 0$
- **Connected predicate is commutative.**
  - $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Connected(t_2, t_1)
- **The gate behaviour is determined by its type.**
  - $\forall g$ Type(g) = OR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 1
  - $\forall g$ Type(g) = AND $\Rightarrow$ Signal(Out(1,g)) = 0 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 0
  - $\forall g$ Type(g) = XOR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ Signal(In(1,g)) $\neq$ Signal(In(2,g))
  - $\forall g$ Type(g) = NOT $\Rightarrow$ Signal(Out(1,g)) $\neq$ Signal(In(1,g))



KE process: specific problem instance



Type(X_1) = XOR
Type(X_2) = XOR
Type(A_1) = AND
Type(A_2) = AND
Type(O_1) = OR

Connected(Out(1,X_1),In(1,X_2))
Connected(Out(1,X_1),In(2,A_2))
Connected(Out(1,A_2),In(1,O_1))
Connected(Out(1,A_1),In(2,O_1))
Connected(Out(1,X_2),Out(1,C_1))
Connected(Out(1,O_1),Out(2,C_1))

Connected(In(1,C_1),In(1,X_1))
Connected(In(1,C_1),In(1,A_1))
Connected(In(2,C_1),In(2,X_1))
Connected(In(2,C_1),In(2,A_1))
Connected(In(3,C_1),In(2,X_2))
Connected(In(3,C_1),In(1,A_2))

KE process: querying and debugging

Query is a **logical formula**.

- What combination of inputs would cause the sum output to be 0 and carry-bit output to be 1?
  - $\exists i_1,i_2,i_3$ Signal(In(1,C_1)) = i_1 $\wedge$ Signal(In(2,C_1)) = i_2 $\wedge$ Signal(In(3,C_1)) = i_3 $\wedge$ Signal(Out(1,C_1)) = 0 $\wedge$ Signal(Out(2,C_1)) = 1

Answer is obtained as **substitutions of variables** i_1,i_2,i_3.

- $\{i_1/1, i_2/1, i_3/0\}$, $\{i_1/1, i_2/0, i_3/1\}$, $\{i_1/0, i_2/1, i_3/1\}$

Debug the knowledge base

- Some queries may give an unexpected (wrong) answer that indicates a problem in knowledge base (wrong/ missing axiom, …).
  - A typical problem is a missing axiom claiming that constants identify different objects.
    - $1 \neq 0$

Objects and categories

- Let us notice that
  - agents **manipulate** with real **objects**
  - but **reasoning** is done at the level of **categories**
  - An agent uses observations to find properties of objects that are used to assign objects to categories. Reasoning on category then reveals useful information about the object itself.
- **Category**
  - = a set of its members
  - = a complex object with relations
    - MemberOf
    - SubsetOf

Categories in FOL

How to represent a category in FOL?

- an object is a **member** of a category
  - MemberOf(BB_{12},Basketballs)
- a category is **subset** of another category
  - SubsetOf(Basketballs,Balls)
- all members of the category have some **property**
  - $\forall$ x (MemberOf(x,Basketballs) $\Rightarrow$ Round(x))
- all members of the category can be **recognized** using common properities
  - $\forall$ x (Orange(x) $\wedge$ Round(x) $\wedge$ Diameter(x)=9.5in $\wedge$ MemberOf(x,Balls) $\Rightarrow$ MemberOf(x,BasketBalls))
- category may also have some property
  - MemberOf(Dogs,DomesticatedSpecies)

Taxonomy

- Categories organize and simplify knowledge base by using **inheritance of properties**.
  - properties are defined for a category but they are inherited to all members of the category
  - food is eatable, fruits are food, apples are fruits, and hence apples are eatable
- Subclasses organize categories to a **taxonomy**
  - a hierarchical structure that is used to categorize objects
  - originally proposed for classifying living organisms (alfa taxonomy)
  - categories for all knowledge
    - Used in libraries
    - Dewey Decimal Classification
    - 330.94 European economy