Languages and Algorithms for AI
2nd Module

Maurizio Gabbrielli
DISI (Dipartimento di Informatica – Scienza e Ingegneria)
Università di Bologna

Credits

- The slides of the course contain also material from :

  - Russel Norvig (AI book)
  - Claudio Sacerdoti Cohen (logic course at CS, UniBo, in Italian)
  - Dirk van Dalen’s (book “Logic and structure”
  - E. Clarke (slides)

Organization of the course

- 1st module: Lodi
- 2nd module: This one
- Integrated course: Dal Lago

- Different exams for each module

- Exam for this module: only written

Content for the this module

- Symbolic AI and the need of logic

- Introduction to logic
  - Propositional logic
  - FOL
  - Resolution
  - Unification

- Logic programming
  - Procedural interpretation
  - Declarative interpretation (?)
  - Prolog (1^{st} part, 2^{nd} part in another course)

- Constraint programming
  - CLP
  - MiniZinc (1^{st} part, 2^{nd} part in another course)

Reading

- *Handouts* (These will be uploaded on Virtuale together with the slides and other material).
- *Dirk van Dalen. Logic and structure. 4th edition, Springer (available also online free).*
- *Bratko. Prolog programming for artificial Intelligence*. Available at https://silp.iiita.ac.in/wp-content/uploads/PROLOG.pdf
- Triska the power of Prolog. Available at https://www.metalevel.at/prolog
- Russell, Norvig. Artificial Intelligence: A Modern Approach (any edition). Pearson.
- The MiniZinc Handbook. Available at https://www.minizinc.org/doc-2.5.0/en/index.html

Two motivational examples

1) A combinatorial problem

2) An optimization problem

Example 1

Problem: arrange three 1s, three 2, ... three 9s in sequence so that for all i∈[1,9] there are exactly i numbers between successive occurrences of i

A solution:

1,9,1,2,1,8,2,4,6,2,7,9,4,5,8,6,3,4,7,5,3,9,6,8,3,5,7

A Prolog program which solves previous problem:

```
sequence([_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_,_]).
% Sequence(X) -> X is a lit of 27 variables

question(S):-
    sequence(S),
    sublist([9,_,_,_,_,_,_,_,_,_,9,_,_,_,_,_,_,_,_,_,9],S),
    sublist([8,_,_,_,_,_,_,_,_,8,_,_,_,_,_,_,_,_,8] ,S),
    sublist([7,_,_,_,_,_,_,_,7,_,_,_,_,_,_,_,7] ,S),
    sublist([6,_,_,_,_,_,_,6,_,_,_,_,_,_,6] ,S),
    sublist([5,_,_,_,_,_,5,_,_,_,_,_,5] ,S),
    sublist([4,_,_,_,_,4,_,_,_,_,4] ,S),
    sublist([3,_,_,_,3,_,_,_,3] ,S),
    sublist([2,_,_,2,_,_,2] ,S),
    sublist([1,_,1,_,1] ,S).
% S is a solution
```

Example 2: Cakes problem (a simplified production planning problem)

We need to bake some cakes for a fete at our local school.

We know how to make two sorts of cakes.

1. A banana cake which takes 250g of self-raising flour, 2 mashed bananas, 75g sugar and 100g of butter,
2. A chocolate cake which takes 200g of self-raising flour, 75g of cocoa, 150g sugar and 150g of butter.

We can sell a chocolate cake for $4.50 and a banana cake for $4.00.

We have 4kg self-raising flour, 6 bananas, 2kg of sugar, 500g of butter and 500g of cocoa.

The question is how many of each sort of cake should we bake for the fete in order to maximize the profit.

A MiniZinc program which solves the Cakes problem

CAKES ≡ [DOWNLOAD]

```
% Baking cakes for the school fete

var 0..100: b; % no. of banana cakes
var 0..100: c; % no. of chocolate cakes

% flour
constraint 250*b + 200*c <= 4000;
% bananas
constraint 2*b  <= 6;
% sugar
constraint 75*b + 150*c <= 2000;
% butter
constraint 100*b + 150*c <= 500;
% cocoa
constraint 75*c <= 500;

% maximize our profit
solve maximize 400*b + 450*c;

output ["no. of banana cakes = \(b)\n",
         "no. of chocolate cakes = \(c)\n"];
```

Figure 3: Model for determining how many banana and chocolate cakes to bake for the school fete

Bob Kowalski: "Algorithm = Logic + Control"

In traditional programming:

programmer takes care of both aspects

In declarative programming:

programmer takes care only of Logic:

interpreter of the language takes care of Control

The task of the programmer is to specify the problem to be solved: we have to ``declare'':

what we want to obtain

and not

how we achieve that

Declarative programming: three paradigms

1. Logic Programming (Prolog …)

2. Functional programming (ML, Haskel, OCAML …)

3. Constraint Programming (MinZinc, CLP, ILOG, …)

Constraint programming represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem, the computer solves it. E.C. Freuder

Logic

Good, too, Logic, of course; in itself, but not in fine weather.

Arthur Hugh Clough, 1819-1861

Die Logik muß für sich selber sorgen

Ludwig Wittgenstein, 1889-1951

Logic in Wikipedia

Logic (from the ancient Greek Ancient Greek λογική) is the systematic study of the form of valid inference, and the most general laws of truth. A valid inference is one where there is a specific relation of logical support between the assumptions of the inference and its conclusion. In ordinary discourse, inferences may be signified by words such as therefore, thus, hence, ergo, and so on.

There is no universal agreement as to the exact scope and subject matter of logic but it has traditionally included the classification of arguments, the systematic exposition of the 'logical form' common to all valid arguments, the study of proof and inference, including paradoxes and fallacies, and the study of syntax and semantics.

Historically, logic has been studied in philosophy (since ancient times) and mathematics (since the mid-19th century), and recently logic has been studied in cognitive science (encompasses computer science, linguistics, philosophy and psychology).

Many different Logics …

- Classical logic (propositional, FOL …): considers truth and inference
  If X and Y are true, can I deduce that also W is true?
- Intuitionistic logic: considers constructive proofs
  If I can construct a proof for X and for Y, can I construct a proof also for W ?
- Linear logic: considers resources
  If I have X and Y, can I exchange them for W?
- Epistemic logic: considers knowledge and belief
  If I believe that you know X and Y, can I conclude that I believe W?
- Temporal logic: considers evolution in time
  If X and Y will happen, can I conclude that W could happen?

... with many different uses in Computer Science and AI

Classical logic:

AI, Logic Languages, specification and verification of software

Intuitionistic logic:

construction of software correct by design

Linear logic:

resources control

Epistemic logic:

security

Temporal logic:

specification and verification of software

Logics

- Many different logics, but similar problems and techniques

- An interdisciplinary subject:

  Philosophy

  Mathematics

  computer science

  AI

- Mathematical Logic =

  Application of mathematics methodology to the study of logic

  but also

  logic as the foundation of mathematics

  (different logics => different mathematics)

Why studying logic in Computer Science

1) Computer Science derives from Logic

2) Foundation of computation and of declarative programming languages

1) Foundation of theoretical computer science (computability)

2) Proving correctness of software

3) Logic as a paradox free language

A first question

- Was invented first the computer or the computer science?

A first question

- Was invented first the computer or the computer science?

- The answer seems obvious … however

A first question

- Was invented first the computer or the computer science?
- The answer seems obvious … however
- Computer science exists before and independently of computer (as a study of computability)

1) Computer Science derives from Logic

- Ancient mathematics (apart from geometry) had no strict formal basis

  These are searched in logic and arithmetic

- Mathematical methods are applied to the study of logic

  The logical derivation is identified with a computation process (calculemus!)

- Computation processes are studied with surprising results

  There exist non computable functions (actually, the majority of functions)

  There exists a class of computable functions that can be defined in many, completely different ways (Lambda calculus, Turing machine, logic programs, RAM

- Computer science is born (about 1930)

- Computer science exists before and independently of computer (as a study of computability)

Computer Science is no more about computers than Astronomy is about telescopes. -- E.W. Dijkstra (1930 – 2002); Turing award 1972.

2) Foundation of computation and of declarative programming languages

Two main classes of programming languages: Declarative (functional, logic, constraint) and imperative

Functional languages (LISP, ML, Haskell, OCAML etc.)

- Based on (composition of) functions
- Strong relations with logic: programming corresponds to proving (in a suitable logic); typing correspond to enunciate theorems.

Logic Languages (Prolog, CLP, CHR, CCP …)

- Based on the notion of relation
- Even stronger relations with logic: computation = deduction
- Algorithm = Logic + Control

Constraint Languages (CLP, CHR, CCP, Minzinc, …)

- Based on the notion of relation and on specific domain solvers
- Both logic based languages and imperative ones
- Used for many practical applications

Two more questions

- Given a statement which is either true or false can we always prove it?
- Given a generic function can we compute it using a computer? (I.e., given a problem, can we solve it using a computer)

3) Foundation of theoretical computer science (computability)

Study of the limits of human knowledge

- Not everything can be proved to be true or false
- Not everything can be calculated
- Limits unavoidable to computer science and artificial intelligence

Another question

- Can we be sure that a generic program written in a common programming language is correct?

4) Proving correctness of software

A fundamental result:
Most properties of programs are not decidable

For example, it cannot exist an automatic tool that, given a generic program (in a Turing complete language), decide whether the program is correct (e.g. terminates)

A possible approach: Given a program and a specification, use logic to prove that the program satisfies the specification.

Automatic proofs: only in some specific cases or approximation of properties.

Assisted proofs: the human does the proof (using logic), the computer verifies it.

Three important limitations of functions computed by a computer which affect correctness.

Important note: the following results hodl when considering a Turing complete language. i.e. a programming language which has the maximal (known) expressive power).

Limitation 1: An apparent paradox

Define (in a programming language) a function f as follows:

$$f(g) = \text{not } (g(g))$$

Then apply f to itself. We have

$$f(f) = \text{not } (f(f))$$

Fact 1) Functions computed by programs are not only total !

Limitation 2: Termination is not decidable

We prove, by contradiction, that it cannot exists a program (in any programming language) f such that, given as input a program g and an argument x:

f(g,x) = true iff g(x) terminate

f(g,x) = false   iff g(x) does not terminate

Fact 2) Termination is not decidable

2: Termination is not decidable, proof

Proof by contradiction. Assume that a program f exists such that, given as input a program g and an argument x:

f(g,x) = true iff g(x) terminate

f(g,x) = false iff g(x) does not terminate

Define

```
h(g) = if f(g,g) = true then
              while true do skip
              else return 0
```

Then

h(h) terminates iff f(h,h) = false iff h(h) does not terminate

So f cannot exist!

Limitation 3: Programming languages can express only a (very) limited number of functions

1) The set of programs which can be expressed in any programming language is countable (easy proof, considering the definition of program and its possible enumeration)
2) Define F = {f | f: N –> {0,1}} (the set of boolean functions}. Then F is not countable (proof, see next slide)

3) Consequence, any programming language cannot express all boolean functions (and hence, a fortiori, all mathematical functions).

The set of Boolean functions is not countable

Define F = {f | f: N –> {0,1}}. Then F is not countable.

Proof (reductio ad absurdum).

Assume that F is countable and let us consider an enumeration f_1, f_2, f_3, ... of all the functions in F.

Define the function f* as follows

$f^*(n) = 1$ iff $f_n(n) = 0$

$f^*(n) = 0$ iff $f_n(n) = 1$

Clearly f* differs from each function f_i (since, for each i, f*(i) is different from $f_i(i)$). This contradicts the fact that f_1, f_2, f_3, ... is an enumeration of all the functions in F and completes the proof.

Alternative proof

Consider the set T of all possible values of a programming language.

T contains at least booleans and functions.

Assume also that a function in our programing language is a mathematical function. Then

$T = \{0,1\} \cup T^T$

Absurd: $| T | < 2 + | T^T |$ (see next slide)

Consequence:

Fact 3) any programming language cannot express all mathematical functions!

Cantor diagonalization

Theorem. Assume that T contains at least two elements.

Then $|T| < |T^T|$.

Proof.

Obviously is cannot be $|T| > |T^T|$ hence it suffices to prove that $|T| \neq |T^T|$.

Reduction ad absurdum. Assume that $|T| = |T^T|$, then there exists a bijection $g: T \to T^T$.

Define $f(t) = t'$ such that $t' \neq g(t)(t)$, with $t \in T$ and $f \in T^T$.

Then for each $t \in T$ we have $f \neq g(t)$, since $f \in T^T$ this means that g is not a bijection.

5) Logic as a paradox free language

- Paradox: apparently correct reasoning (w.r.t. given rules and axioms) from apparently correct premises which allow to derive an apparently contradictory conclusion

- Natural language allows to express paradoxes, hence not appropriate as a basis for mathematics or CS

- Logic provides languages paradox free

Paradoxes

- False Paradox (find the error):

  $x=1 \Rightarrow x^2 = x \Rightarrow x^2 - 1 = x - 1 \Rightarrow (x-1)(x+1) = x-1 \Rightarrow$

  $\Rightarrow x+1 = 1 \Rightarrow x = 0$

- Natural language allows paradoxes

  (1)

  I am a liar

  I am liar iff what I am saying is not true

  I am liar iff “I am a liar” is not true

Paradoxes

- Natural language allows paradoxes (2)

  Definition. Le x be the smallest number which cannot be defined with less than 1000 words

  x is defined iff x is the smallest number which cannot be defined with less than 1000 words

  x is defined (with less than 1000 words) iff x is not defined (with less than 1000 words)

Why paradoxes

Why these paradoxes?

1. Metalinguistic use of natural language
2. Self-application of a meta linguistic concept

Meta-linguistic: of or relating to a metalanguage

Metalanguage: a language used to talk about language, something about the language

How to avoid paradoxes

How to solve these problem

1. Abandon natural language
2. Use suitable language from mathematical logic

Paradoxes in mathematics

Also mathematic language allows paradoxes

Russel Paradox: Let $X = \{Y \mid Y \notin Y\}$

Then $X \in X$ iff $X \notin X$

Solving the Russel paradox

How to solve Russel paradox:

$\{Y \mid Y \notin Y\}$ is not a set

- Not allowed to use the notion of set in a meta-linguistic way, i.e., the collection of all sets is not a set.

- Sets can be constructed only selecting elements from sets (so $\{Y \in \text{class of all sets} \mid Y \notin Y\}$ is not a set, but a class, hence it does not contain itself.

Paradoxes in computer science

Many programming languages allow higher order functions (A function that takes as input another function)

Meta-linguistic features of programming languages

Auto-application possible

Paradoxes not avoidable, actually these paradoxes show limitations of our computational models (see later)

Logic to avoid paradoxes

Curry's paradox

If this sentence is true, then Santa Claus exists *

Call the above sentence A and "then Santa Claus exists" B

To "prove" that A is true:

Assume that the premise of A is true, that is " this sentence is true" is true. We must prove that B is true.

Assuming that the premise of A is true, means to assume that A->B is true and then, by modus ponens, B is true. Hence A is true.

Since A is true, and A is the premise in the implication *

Curry’s paradox in a logic setting

1. $X := (X \rightarrow Y)$ assumption, the starting point, equivalent to "If this sentence is true, then Y"

2. $X \rightarrow X$ (law of identity_

3. $X \rightarrow (X \rightarrow Y)$ substitute right side of 2, since X is equivalent to

$X \rightarrow Y$ by 1

4. $X \rightarrow Y$ from 3 by contraction

5. X substitute 4, by definition in 1

6. Y from 5 and 4 by modus ponens

So a paradox again …?

Depends on 1, in many logics if Y is false 1 is not possible!

Why studying logic in AI

a) A tool for expressing human reasoning

b) Foundation of a classic approach to AI (see later, and see also the Foundations course)

c) Foundation of (declarative) languages used in AI (see this course)

d) Direct application in some programing languages used in AI (e.g. PROLOG and CLP, see this course)

Btw, what is AI …

… and when all started?

A ultra short history

Analogously to Computer Science, AI existed before the Computers!

An historical perspective

Ramon Llull (1232-1316): Ars combinatoria: a physical machine (paper discs) which allowed to combine the concepts symbolically represented on the disks

Every single letter, from *B* to *K*, represents not merely itself, but several strictly defined and placed meanings. By writing the letters from B to K as key-terms heading a table, a series of different sentences can be easily constructed. For example, B=Bonitas, C=Magnitudo, D=Duratio, E=Potestas, F=Sapientia, G=Voluntas, H=Virtus, I=Veritas and K=Gloria. This is, initially, the paper-circle called the *Prima Figura*. The next strictly defined table of words can be produced on the next circle, perhaps as seen on the *Secunda Figura* (shown below), where we find categories and relations of thinking.

Hence the machine allows all the words to be combined by turning the circles step by step. In this manner, it is possible to connect every word with every other word placed in a position of a table—depending only on the construction of the individual tables. Imagine how Llull could play this out: bearing in mind the inscribed words of the Prima Figura. These nine words are none other than the attributes of God. Combined with a table of nine questions, it is possible to construct the skeleton of the "Proofs of God." The machine shows all possible statements and declarations about God.

What is AI? An historical perspective

- René Descartes (1596-1610): in a sense a forereunner of cogntive science and AI for his studies on the cognitive aspects of automatons
- Leibniz (1646-1716): Calculus raziocinator which should have allowed to solve all problems (Calculemus!)

René Descartes 1596-1650

Influenced by the automatons on display throughout the city of Paris, began to investigate the connection between the mind and body. Dualism:

- body works like a machine, that it has material properties.
- mind is not material and does not follow the laws of nature.
- mind interacts with the body at the pineal gland

Cartesio, can be considered a precursor of AI because he considered the constructions of automaton an important cognitive process. [1]

1) Bruno G. Bara, Scienza cognitiva. Un approccio evolutivo alla simulazione della mente, Bollati Boringhieri, Torino 1982

AI (and CS) ante litteram in the XVIII century

- G.W. Leibniz (1646-1716) Calculus raziocinator:

" I believe I can devise a certain universal language, by which we can perform calculations on all kinds of subjects and find demostrations, as in algebra and arithmetics"

- Quo facto, quando orientur controversiae, non magis disputatione opus erit inter duos philosophos, quam inter duos computistas. Sufficiet enim calamos in manus sumere sedereque ad abacos, et sibi mutuo (accito si placet amico) dicere: calculemus!

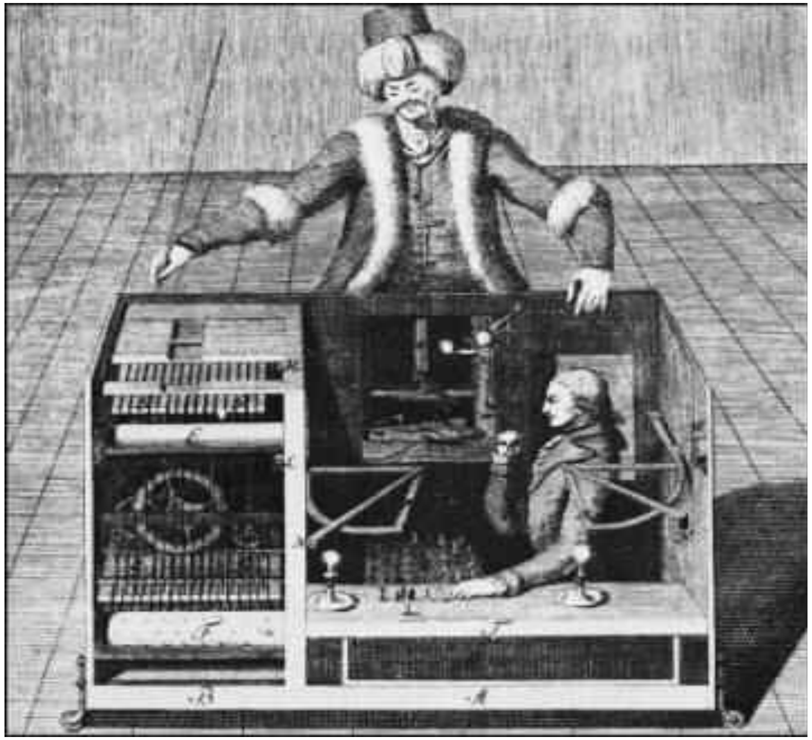
(Leibniz: De scientia universalis seu calculo philosophico)

AI (and CS) ante litteram in the XVIII century

- Jacques de Vaucanson(1709-1782) built an artificial duck that could swim and “eat” grain (400 parts in each wing)
- Pierre Jacquet-Droz (1721–1790) swiss-born watchmaker who designed and built animated dolls and automata including a writer and a musician
  - Movements controlled by an input coded on a metallic disc !

AI ante litteram … fake !

An image from
18th century



AI before AI term was coined, i.e. Turing

- 1935 Universal Turing Machine

- 1947 Talk by A. Turing in London: "What we want is a machine that can learn from experience … the possibility of letting the machine alter its own instructions provides the mechanism for this".

- 1948. *Intelligent Machinery*. Never published during his life, a real AI manifesto contain several ideas later re-invented (including symbolic systems and neural netwroks).

- 1950. *Computing Machinery and Intelligence*. Including the imitation game, i.e. Turing test (we will see this later)

SUMMARY.

The possible ways in which machinery might be made to show intelligent behaviour are discussed. The analogy with the human brain is used as a guiding principle. It is pointed out that the potentialities of the human intelligence can only be realised if suitable education is provided. The investigation mainly centres round an analogous teaching process applied to machines. The idea of an unorganised machine is defined, and it is suggested that the infant human cortex is of this nature. Simple examples of such machines are given, and their education by means of rewards and punishments is discussed. In one case the education process is carried through until the organisation is similar to that of an ACE.

An excerpt from Turing: answer to the question "Can machines think?"

According to the most extreme form of this view [that thought is impossible without consciousness] the only way by which one could be sure that a machine thinks is to be the machine and to feel oneself thinking. One could then describe these feelings to the world, but of course no one would be justified in taking any notice.

Likewise according to this view the only way to know that a man thinks is to be that particular man. It is in fact the solipsist point of view. It may be the most logical view to hold but it makes communication of ideas difficult. A is liable to believe "A thinks but B does not" whilst B believes "B thinks but A does not".

Instead of arguing continually over this question it is usual to have the polite convention that everyone thinks.

A. Turing. Mind. 1950.

Neurons and learning

- W. S. Mc-Culloch and W. H. Pitts (1943). First computational model of the neuron (threshold logic)

  At the basis of the research on neural networks

- D.O. Hebb (1949). Hebbian learning.

  A theory in neuroscience that proposes an explanation for the adaptation of neurons in the brain during the learning process, describing a basic mechanism for synaptic plasticity neuronal basis of unsupervised learning

- F Rosenblatt (1958). Perceptron.

  An algorithm for pattern recognition based on a two-layer computer learning network using simple addition and subtraction

The official start of AI: Dartmouth **conference** 1956

John McCarty coined the term AI: “The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it”.

McCarty also invented LISP!

Trenchard More, John McCarthy, Marvin Minsky,
Oliver Selfridge, Ray Solomonoff

So, what is AI today?

The four main current view of AI …

What is AI?

Systems that think like humans	Systems that think rationally
Systems that act like humans	Systems that act rationally

Some definitions of AI (Russel and Norvig)

Thinking Humanly	**Thinking Rationally**
"The exciting new effort to make computers think … *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning …" (Bellman, 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Acting Humanly	**Acting Rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998) "AI …is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Thinking humanly: imitate the brain

Why not imitate the human brain ? We do not know enough:

- Broca (1861) relation between language and specific areas in the brain
  - Golgi (1873) first visualization of neurons
  - Berger (1929) EEG
  - Ogawa et al (1990) FMRI: measurement of cognitive processes.

Today

- We know some relations between areas in the brain and specific functions
  - do not know how a cognitive process works
  - do not know how memory works

Thinking humanly: cognitive science

1960s "cognitive revolution": information-processing psychology replaced prevailing orthodoxy of behaviorism (this was concerned with "measures" of stimula/answers only)

Requires scientific theories of internal activities of the brain

– What level of abstraction? "Knowledge" or "circuits"?
– How to validate? Requires
  1) Predicting and testing behavior of human subjects (top-down)
  or 2) Direct identification from neurological data (bottom-up)

Both approaches (roughly, Cognitive Science and Cognitive Neuroscience) are now distinct from AI

Both share with AI the following characteristic:
the available theories do not explain (or engender) anything resembling human-level general intelligence

Thinking rationally: Laws of thought, that is Logic

Normative (or prescriptive) rather than descriptive

Aristotle: what are correct arguments/thought processes?

Several Greek schools developed various forms of logic:
notation and **rules of derivation** for thoughts;
may or may not have proceeded to the idea of mechanization

Direct line through mathematics and philosophy to modern AI

Problems:
1) Not all intelligent behavior is mediated by logical deliberation
2) What is the purpose of thinking? What thoughts **should** I have out of all the thoughts (logical or otherwise) that I **could** have?

Acting rationally

Rational behavior: doing the right thing

The right thing: that which is expected to maximize goal achievement, given the available information

Doesn't necessarily involve thinking—e.g., blinking reflex—but thinking should be in the service of rational action

Aristotle (Nicomachean Ethics):

> **Every art and every inquiry, and similarly every action and pursuit, is thought to aim at some good**

Acting Humanly: An excerpt from Turing

According to the most extreme form of this view [that thought is impossible without consciousness] the only way by which one could be sure that a machine thinks is to be the machine and to feel oneself thinking. One could then describe these feelings to the world, but of course no one would be justified in taking any notice.

Likewise according to this view the only way to know that a man thinks is to be that particular man. It is in fact the solipsist point of view. It may be the most logical view to hold but it makes communication of ideas difficult. A is liable to believe "A thinks but B does not" whilst B believes "B thinks but A does not".

Instead of arguing continually over this question it is usual to have the polite convention that everyone thinks.

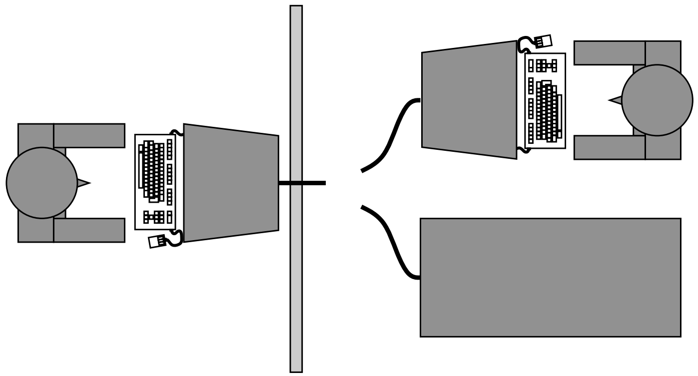
A. Turing. Mind. 1950.

Acting humanly: The Turing Test

Turing (1950) "Computing machinery and intelligence":

◊ "Can machines think?" $\longrightarrow$ "Can machines behave intelligently?"

◊ Operational test for intelligent behavior: the Imitation Game



◊ Predicted that by 2000, a machine might have a 30% chance of fooling a lay person for 5 minutes

◊ Anticipated all major arguments against AI in following 50 years

◊ Suggested major components of AI: knowledge, reasoning, language understanding, learning

Problem: Turing test is not **reproducible**, **constructive**, or amenable to **mathematical analysis**

So, four possible definition of AI:

1)Thinking humanly
2)Acting humanly
3)Thinking rationally
4)Acting rationally

We, as computer scientists (and AI researchers) take the "simple" view point of A.M. Turing and consider AI systems those which act humanly (or, at least, ratiuonally)

So, the official definition of AI

The ability of a computer to perform tasks commonly associated with intelligent beings

Two main approaches

Symbolic AI

Sub-symbolic AI (including ML).

This motivates the definition by Norvig!

A more complex definition?

Peter Norvig (Director of Research, Google):

AI is "Knowing what to do when you don't know what to do"

Two main AI categories

1. Symbolic computation

2. Subsymbolic (connessionstic) computation

1^{st} Example : A System which identify severe infections caused by bacteria and recommend antibiotics

Sys: The culture is Gram positive?

User: NO.

Sys: The culture is Gram negative?

User: YES.

Sys: The patient is allergic to antibiotics?

User: NO.

Sys: I would suggest a therapy based on antibiotics.

Is this ChatGPT ?

No, indeed it can provide explanations for recommendations:

Explaining session:

Sys: The culture is Gram positive

User: WHY? < why you need this information?

Sys: This information is needed to determine whether (R1) the organisim is a streptococcus

User: WHY?

Sys: This information is helpful because for streptococcus one cas use penicilina (R3).

Sys: ... : I would suggest a therapy based on antibiotics.

User: HOW? <How can you infer this conclusion ?

Sys: I used rule R4: I suggested to use antibiotics because they are appropriate for this infection (prem1) and the patient is not allergic (prem2).

User: HOW Premise 1 ?

<How can you derive that antibiotics are appropriate?

Sys: I used rule R5: that antibiotics are appropriate because the bacterium causing the infection is ecoli.

Symbolic computation (expert systems)

I. Logic based reasoning
II. Can provide explanations for decisions
III. Not appropriate for problems such as image recognition (cannot be solved algorithmically)

Expert systems and decision support systems (1970 -)

Needed a domain knowledge provided by humans (typically rules expressed in logic)

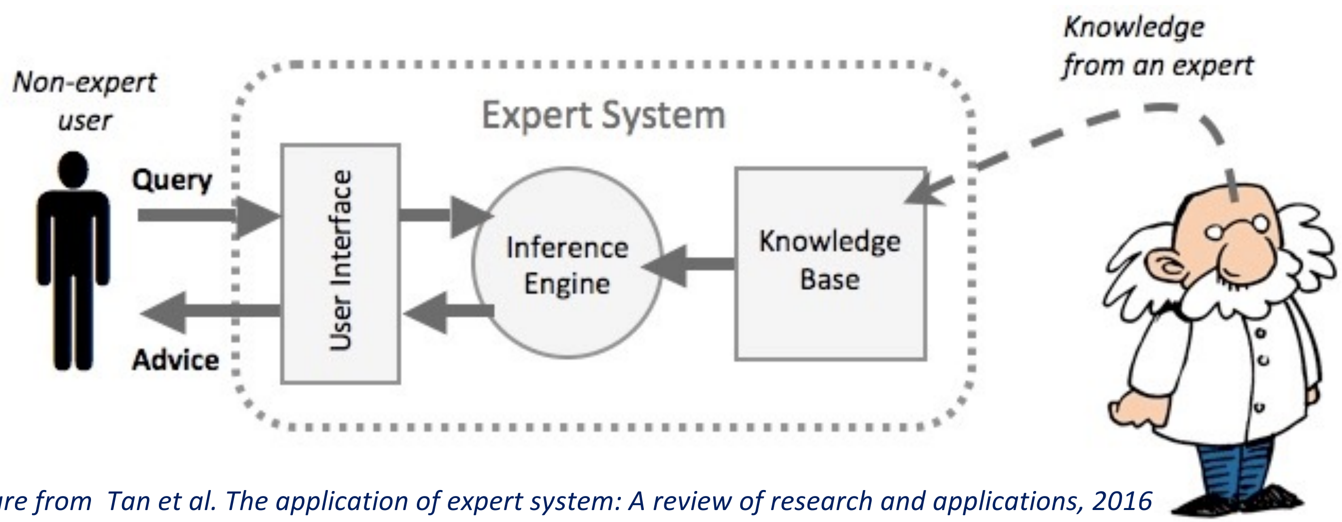
Solve problems in a limited domain

Have performance similar to an expert of the domain

Uses knowledge representation and search in large spaces

Build dynamically the solution

Search and solution generation by means of an inference engine which elaborates rules



Picture from Tan et al. The application of expert system: A review of research and applications, 2016

Logic and AI

Many different logics

- Propositional, first-order, modal (epistemic, temporal, deontic) …

Many techniques for representing knowledge and reasoning

- Deduction, Induction, Abduction …

Many languages and tools

- PROLOG, Constraint programming

As the base of many Expert Systems

Formalizing the knowledge by using logic for the diagnosis problem

Goal: suggest a Drug which is appropriate for a patient on the basis of some clynical analysis
`prescribe(Drug).`

Knowledge base

Facts:

`gram(neg).`

`not(allergic(antb)).`

Rules:

`R1: gram (neg) → id (ecoli).`

If the result of the analysis is *gram-negativ* then the identity is *enterium-coli*

`R2: gram (pos) → id (strep).`

If the result of the analysis is *gram-positivo* then the identity is *streptococcus*

`R3: id(strep) OR id(bact) → ind(pen).`

If the identity is streptococcus or bacterium then it is appropriate to indicate penicillin

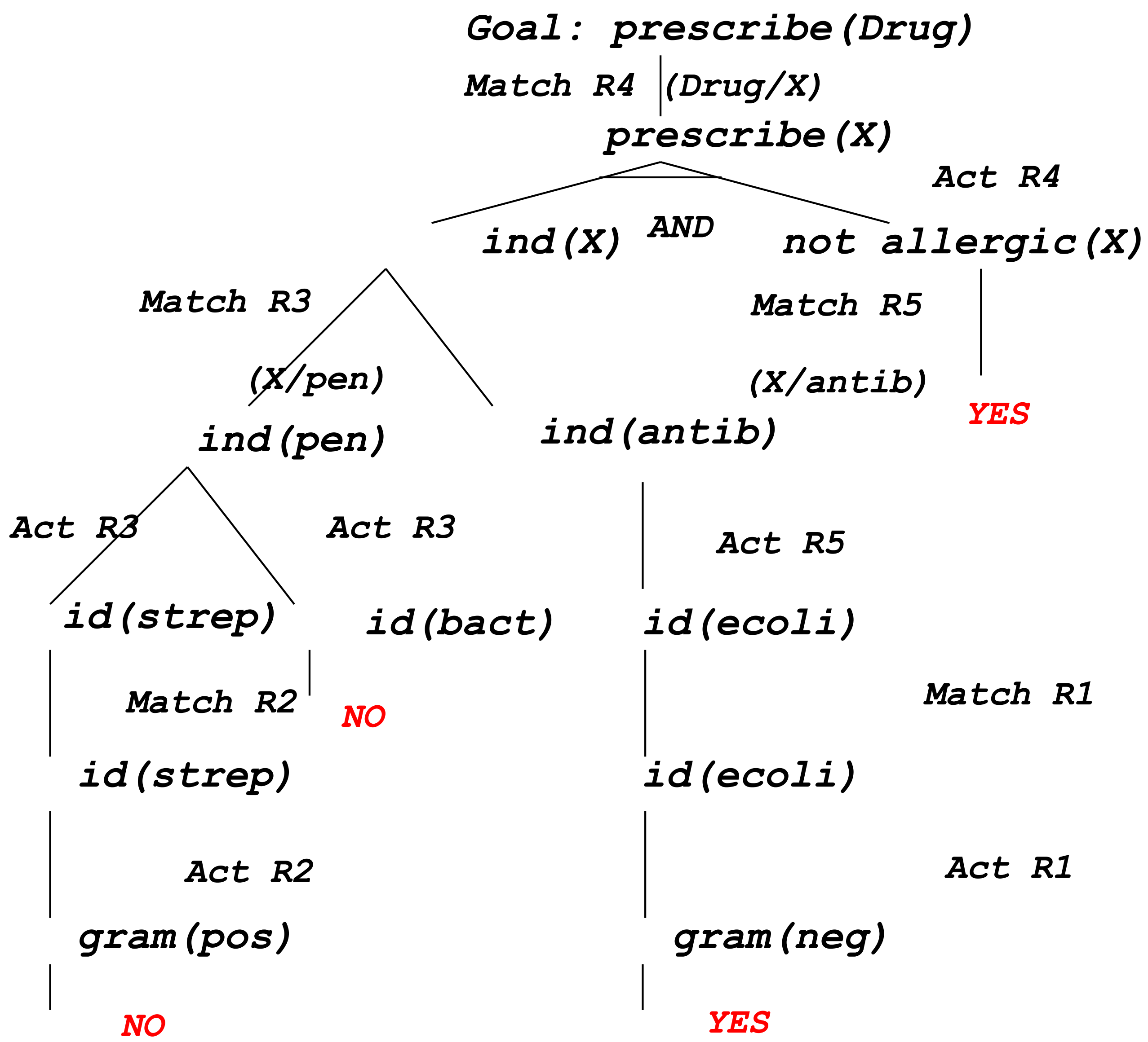
`R4: ind(X) AND not (allergic(X)) → prescribe(X)`

If it is appropriate to indicate the drug X and the patient is not allergic to X then we can prescribe X

`R5: id(ecoli) → ind(antb).`

If the identity is ***enterium-coli*** then it isappropriate to indicate antibiotics.

Inference engine



Sub-symbolic computation

I. Based on (artificial) neural networks and ML
II. Very good for image recognition and many problems which cannot be solved algorithmically
III. Cannot provide explanations for decisions (thus far away from passing the Turing test!)

Sub-symbolic computation