Comp24412: Symbolic AI
Lecture 0: Course Introduction

Ian Pratt-Hartmann

Room KB2.38: email: ipratt@cs.man.ac.uk

2016–17

- In 2011, the question-answering system Watson, developed by IBM, won a special round of the TV quiz programme *Jeopardy!*:

- The program uses a variety of text-processing and information-retrieval techniques to produce answers to familiar quiz-type questions, such as

> *Kathleen Kenyon's excavation of this city mentioned in Joshua showed the walls had been repaired 17 times.*

- Despite its sophistication, the program has little idea of what it is doing.

- One aspect of understanding that Watson lacks its an appreciation of logical relationships expressed in language.
- Here are some valid arguments:

  Every beekeeper is an artist
  Fred is a beekeeper

  Fred is an artist

  Every beekeeper is an artist
  Some beekeepers are not carpenters

  Some artists are not carpenters

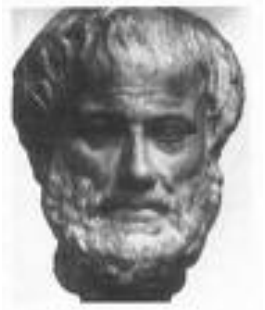
- Here is an invalid argument

  Every beekeeper is an artist
  Some carpenters are not artists

  Some beekeepers are not carpenters

- Question: how do we distinguish between the two?

- The question seems first to have been addressed by Aristotle (384–322 B.C)



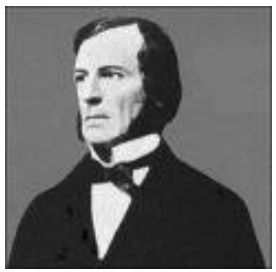
- Aristotle attempted to catalogue all the valid argument forms, or syllogisms
- This is all explained in the (utterly impenetrable) *Prior Analytics*

- The question seems first to have been addressed by Aristotle (384–322 B.C)


- Aristotle attempted to catalogue all the valid argument forms, or syllogisms
- This is all explained in the (utterly impenetrable) *Prior Analytics*

- Aristotle had a rather restricted view of the forms of sentences occurring in arguments.
- Three nineteenth century gentlemen attempted to broaden the scope of this logic:

Augustus De Morgan



George Boole

W.S. Jevons

- De Morgan complained that Aristotle could not account for the evident validity of

Every horse is an animal

Every horse's head is an animal's head

- Likewise, the syllogistic does not help us with

  Every artist admires a beekeeper
  Every beekeeper curses every artist who admires him

  Every artist admires a beekeeper who curses him

- Actually, de Morgan also looked in detail at adding numbers to syllogisms:

  At least 13 artists are beekeepers
  At most 3 beekeepers are carpenters
  At most 4 dentists are not carpenters

  At least 6 artists are not dentists.

- Here is a real killer:

  At most 1 artist admires at most 7 beekeepers
  At most 2 carpenters admire at most 8 dentists
  At most 3 artists admire at least 7 electricians
  At most 4 beekeepers are not electricians
  At most 5 dentists are not electricians
  At most 1 beekeeper is a dentist

- Here is a real killer:

  At most 1 artist admires at most 7 beekeepers
  At most 2 carpenters admire at most 8 dentists
  At most 3 artists admire at least 7 electricians
  At most 4 beekeepers are not electricians
  At most 5 dentists are not electricians
  At most 1 beekeeper is a dentist

  At most 6 artists are carpenters

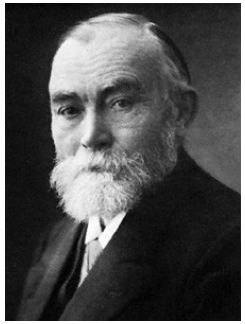
- Here is a real killer:

  At most 1 artist admires at most 7 beekeepers
  At most 2 carpenters admire at most 8 dentists
  At most 3 artists admire at least 7 electricians
  At most 4 beekeepers are not electricians
  At most 5 dentists are not electricians
  At most 1 beekeeper is a dentist

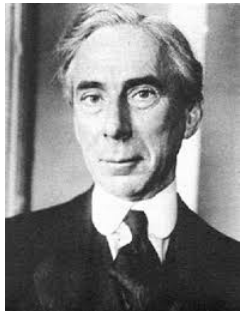
  At most 6 artists are carpenters

- But you knew that, didn't you?

- But the most decisive advance was made at the end of the nineteenth century by Gottlob Frege and Bertrand Russell.



Gottlob Frege



Bertrand Russell

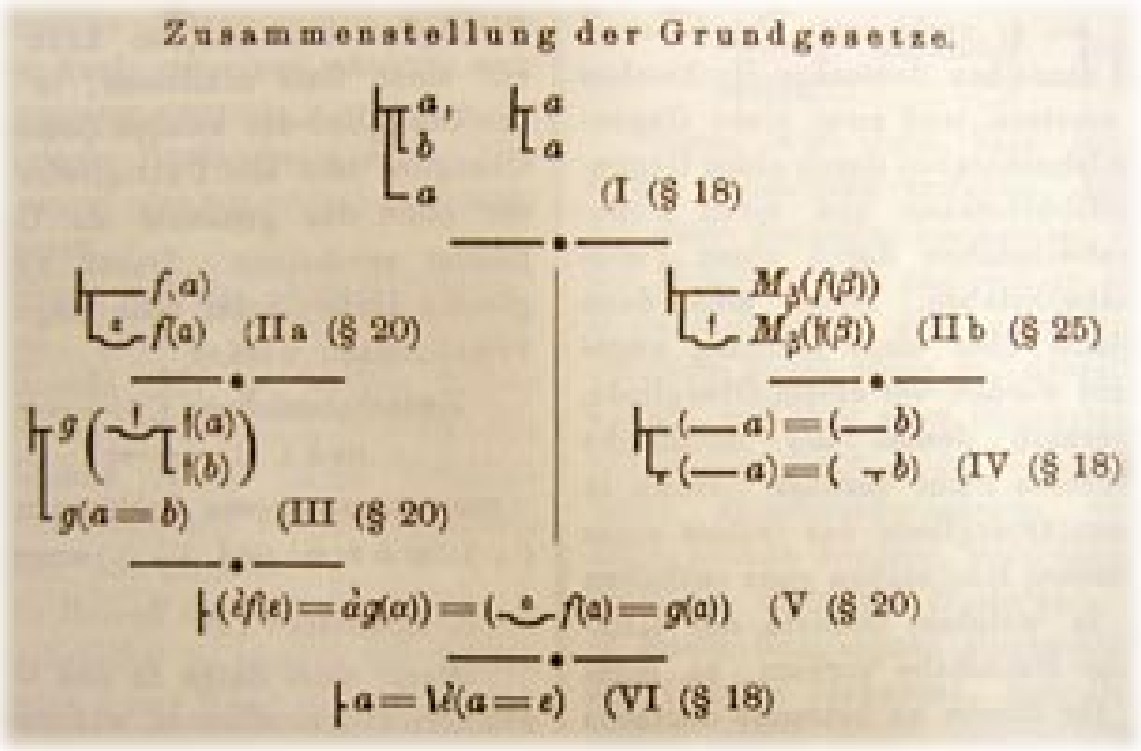
- They developed formal languages to represent information:

Every artist admires a beekeeper
Every beekeeper curses every artist who admires him

Every artist admires a beekeeper who curses him

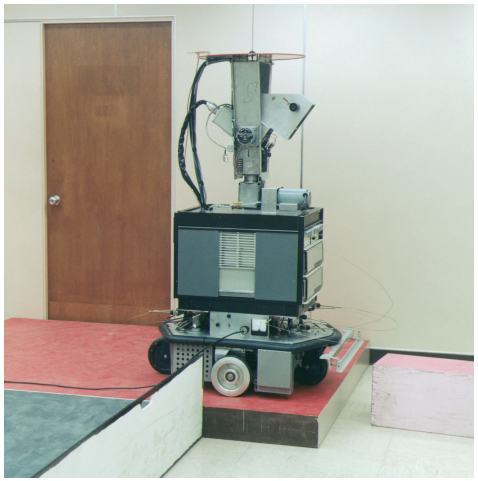
$$\frac{\forall x(\text{artst}(x) \rightarrow \exists y(\text{bkpr}(y) \wedge \text{adm}(x,y))) \quad \forall x(\text{bkpr}(x) \rightarrow \forall y(\text{artst}(y) \wedge \text{adm}(y,x) \rightarrow \text{crs}(x,y)))}{\forall x(\text{artst}(x) \rightarrow \exists y(\text{bkpr}(y) \wedge \text{crs}(y,x) \rightarrow \text{adm}(x,y)))}$$

- Erm, actually, Frege's notation, called the *Begriffsschrift*, was not quite so user-friendly:



But we won't worry about it.

- In the 1980s, serious attempts were made to use logical methods in Artificial Intelligence.
- Here is one early robot, SRI's Shakey:



- The key idea behind logical methods in AI can be summarized as follows
  - The agent's knowledge is represented as a collection formulas in some logic
  - Reasoning tasks (planning, deduction, hypothesis formation, learning) is realized by the manipulation of these formulas.

- Perhaps the most ambitious knowledge-based AI project is CYC.
  - Started 1984 at MCC
  - Contains a huge knowledge-base codifying knowledge of everyday facts.
  - Retrieval of information is by logical inference.
  - Originally estimated that CYC would require 250,000 rules and 350 man-years.
  - Total effort to date is over a man-millennium ...
- It is now widely accepted that this approach will not, by itself, succeed in producing intelligence.

- This course is about logical approaches to AI.
- We shall be concerned with the relationship between logic, natural language and deduction.
- Logic is not the whole of AI, but it is most probably part of it.
- Here is what we have to learn:
  1. Prolog programming
  2. First-order logic
  3. Applications of Logic in AI
  4. Natural language syntax
  5. Lambda calculus
  6. Natural langauge semantics
  7. Resolution theorem-proving

- A student completing this course should:
  1. have a working knowledge of the Prolog programming language
  2. understand the fundamentals of natural language syntax
  3. be able to compute the meanings of a range of natural language sentences
  4. understand the operation and use of automated theorem-provers, and the theoretical reasons for their limitations
  5. Understand how logic may be used to solve problems in AI.

and recommended reading:

- P. Blackburn and J. Bos, *Representation and Inference for Natural Language*, CSLI Press, 2005
- Patrick Blackburn, Johan Bos and Kristina Striegnitz, *Learn Prolog Now!*, `http://www.learnprolognow.org/`
- W.F. Clocksin, C.S. Mellish, *Programming in Prolog* 5th Ed. New York : Springer-Verlag, 2003
- Almost any other Prolog book

Lecture plan

Week	Date	Lecture 1	Lecture 2	Lab
1	30.01.15	L0: Introduction	L1: Prolog I	
2	06.02.15	L2: Prolog II	L3: Prolog III	
3	13.02.15	Prolog Catchup	L4: Search	Intro Prolog (D)
4	20.02.15	L5: Logic	L6: Resolution	
5	27.02.15	L7: Logic and Prolog	L8: Logic and Planning	N Queens (D)
6	06.03.15	L9: Grammar I	L10: Grammar II	
7	13.03.15	L11: Parsing I	L12: Parsing II	Syntax in Prolog (D)
8	20.03.15	L13: Semantics I	L14: Semantics II	
9	27.03.10	L15: Grammar III	L16: Semantics III	Semantics in Prolog (D)
10	24.04.10	L17: Finale	Catch-up	
11	1.05.10	Revision	Revision	Reasoning in NL (D)
12	8.05.10			Marking

- Q: What do I do now?
- A: Several things:
  - Try out Prolog by typing `pl` to the teaching (Linux) system. (You should have `/opt/prolog/bin/` in your `$PATH`)
  - Start reading *Learn Prolog Now!* (or any other book on Prolog). There are also various on-line introductions to Prolog, which are probably just as good.
  - Buy a copy of *Representation and Inference for Natural Language*