

INTRODUCTION

What is **Artificial Intelligence**?
(chapter 1)

Cse352

Lecture Notes (1)

Professor Anita Wasilewska

Introduction

- AI is a broad field. It means different things to different people.
- AI is concerned with getting computers to do tasks that require human intelligence.
 - Example 1 : Complex Arithmetic –Computers can do this very easily.
 - Example 2: Recognizing a face – People do easily, but it was very difficult to automate.

Definition Attempt

- **AI** is concerned with difficult tasks , which require complex and sophisticated **reasoning process** and **knowledge**

Why to automate Human Intelligence?

(and to which degree is it possible?)

Why to automate Human Intelligence ?

- Reason 1: To understand human intelligence better: We may be able to test and refine theories of **Human Intelligence** by writing programs which attempt to simulate aspects of **human behavior**
- Reason 2: To have smarter programs and machines; by studying human reasoning we may develop useful techniques for solving difficult problems.

Science Fiction

Science Fiction Human-like robots –
whether such a **goal** is **possible**
or even **desirable** – belongs to science
fiction

But it **does have impact** on the practical
work of writing smarter programs and
developing **better models** of **human**
reasoning

The progress in **modern day ROBOTICS**

AI as a branch of Science and Engineering

- AI – for us is a **technical subject**; we put emphasis on **computational techniques** and less on **psychological modeling** and **philosophical issues**
- AI is both a **branch of science** and a branch of **engineering**
As **engineering**, **AI** is concerned with the concepts, **theory** and **practice** of building **intelligent machines**

AI as a branch of Science and Engineering

Examples:

1. **Expert Systems** that give advice about specialized subjects; e.g., medicine, mineral exploration, etc....
2. **Question-Answering Systems** for answering queries posed in restricted, but large subset of English and other natural languages.
3. **Theorem Proving Systems.**
4. **Systems for program verifications.** It is a very important field of CS.

Knowledge in Intelligent Entities

“**Intelligent entities** seem to **anticipate their environments** and the consequences of **their actions**”

We **assume** that the **Intelligent entities** possess **knowledge** of their environments

▪

Knowledge in Intelligent Entities

Basic QUESTIONS

- What is **knowledge**?
- What **forms** can it take?
- How do **entities use** knowledge?
- How is **knowledge acquired**?

Knowledge in Intelligent Entities

We have:

- Procedural Knowledge.
- Declarative Knowledge

We talk about and define:

- Knowledge Representation
- Knowledge Base

Forms of Knowledge

There are **two major ways** we can think about machine having knowledge about its world:

- **IMPLICIT – Procedural**
- **EXPLICIT – Declarative**

Forms of Knowledge

The **knowledge** represented by the actual running or execution of a **program** is **procedural**;

Spider knowledge about spinning the web and **tennis** knowledge used by a **player** are both **procedural**

Tennis knowledge as **TAUGHT** by the instructor is a **declarative**

Intelligent Machines need both:
procedural and **declarative** knowledge

Reasons for preferring Declarative Knowledge

- Here are some reasons for **AI researchers** to prefer **declaratively represented** knowledge :
 - Can be **changed** easily.
 - Can be used for several **different purposes**.
 - The knowledge base itself does not have to be repeated or designed for different applications
 - Can be **extended** by **reasoning process** that **derive additional knowledge**

Requirements for Knowledge Representation Languages

- **Representational adequacy:**
It should allow to represent all knowledge that one needs to reason with
- **Inferential Adequacy:**
It should allow **new knowledge** to be **inferred** from basic set of facts
- **Inferential Efficiency:**
Inferences should be made **efficiently**
- **Naturalness:**
The language should be **reasonably natural** and easy to use

Declarative Knowledge

- **AI** focuses strongly on the **declarative knowledge**

- One of classic books

Logical Foundations of Artificial Intelligence

Michael R. Genesereth, Nils J. Nilsson (Stanford University)

is concerned with and based on **declarative knowledge**

Conceptualization

The **formalization** of knowledge in **declarative** form begins with a notion of **conceptualization**

- The **language** of conceptualization is often **predicate calculus**
- Definition presented here is from Nils Nilsson's book

Logical Foundations of AI

Conceptualization

- **Conceptualization** – step one of formalization of knowledge in declarative form.
- $C = (\mathcal{U}, F, R)$
- \mathcal{U} – Universe of discourse; it is a **FINITE** set of objects.
- **F** – Functional Basis Set; Set of functions (defined on \mathcal{U}). Functions may be partial.
- **R** – Relational Basis Set; Set of relations defined on \mathcal{U} .
- Remark: sets **R**, **F** are **FINITE**.

Conceptualization

- **R** – Relational Basis Set; Set of relations defined on \mathcal{U} .
- $\mathbf{R} \in \mathbf{R}$, $\mathbf{R} \subseteq \mathcal{U}^n$, $\# \mathbf{R} = n$

This is like in predicate logic:

$M = (\mathcal{U}, \mathbf{F}, \mathbf{R})$ is a Model. Where $\mathcal{U} \neq \emptyset$,
 $\mathbf{F} \in \mathbf{F}$, $f \in \text{FUN}$, $f_1 = \mathbf{F}$, $f_1 : \mathcal{U}^n \rightarrow \mathcal{U}$, etc.,
Satisfiability Model, etc., in Predicate Logic.

Example: Block World



(Example is continued on next slide.)

Example: Block World

- $\mathcal{U} = \{ a, b, c, d, e \}$
- \mathbf{F} – set of functions; here $\mathbf{F} = \{h\}$
- **Intuitively:** h maps a block into a block on the top of it
- We use **intended interpretation** and write $h = \text{Top}$
- **Formally:** $h = \{(b,a), (c,b), (e,d)\}$, i.e.
- $h(b) = a; h(c) = b; h(e) = d$
- h is a **partial function** and $h : \mathcal{U} \rightarrow \mathcal{U}$
- Domain of $h = \{b,c,e\} \subseteq \mathcal{U}$

Example: Block World

R – Set of Relations (always finite)

- We **define** here 4 relations, i.e.

$$\mathbf{R} = \{\text{Above, On, Table, Clear}\}$$

where

$$\text{Above} \subseteq \mathcal{U} \times \mathcal{U} \quad , \quad \text{On} \subseteq \mathcal{U} \times \mathcal{U}$$

- $\text{Table} \subseteq \mathcal{U} \quad , \quad \text{Clear} \subseteq \mathcal{U}$

- Observe that **Above, On** are **two** argument relations and **Table, Clear** are **one** argument relations

Example: Block World

We define intuitively:

Above (x,y) iff x is anywhere above y

We define formally:

Above = {(a,b), (b,c), (a,c), (d,e)}

Above is a two argument relation

We define intuitively:

On (x,y) iff x is immediately above y

We define formally:

On = {(a,b), (b,c), (d,e)} **On** $\subseteq \mathcal{U} \times \mathcal{U}$

On is a two argument **partial function**

Example: Block World

We define intuitively:

Clear(x) iff there is no block on top of x

• **We define formally:**

Clear = {a, d} $\subseteq \mathcal{U}$

•

Clear is one argument **relation**

• **We define intuitively:**

Table(x) iff x is resting directly on the table

We define formally:

Table = {c,e} $\subseteq \mathcal{U}$

• **Table** is one **argument relation**

Example: Block World

- Observe that

$\text{On} \subseteq \text{Above}; \quad \text{Clear} \cap \text{Table} = \emptyset$

We have chosen in our **conceptualization** to define some particular **relations** and **functions**
But depending on what we want **to tell about our world** – we can define less or more of them, or some **totally different sets** of **relations** and **functions**

Intended Interpretation

- We defined

$$\text{On} = \{(a,b), (b,c), (d,e)\}$$

- We can also use other symbols, e.g. :

$$\blacksquare = \{(a,b), (b,c), (d,e)\} \text{ (Math. model)}$$

- This is the same as:

$$\blacksquare(a,b) , \blacksquare(b,c) \text{ and } \blacksquare(d,e)$$

- **Intended Interpretation** of the symbol \blacksquare is as is the intuitive meaning of the word **On** in our definition, i.e. “**x is immediately above y**”

Block World in Prolog

- $On \subseteq \mathcal{U} \times \mathcal{U}$
 $On = \{(a,b), (b,c), (d,e)\}$ (Math. Definition)

- This is **Prolog** like statements:

$On(a,b)$, $On(b,c)$ and $On(d,e)$

Facts in Prolog

It is equivalent to your definition as a **declaration** of what “**On**” means, i.e.

- We write $On(a,b)$ for $(a,b) \in On$
- **Prolog** is called a **declarative programming language**

Representation in Predicate Logic

- **Facts** about our **Universe**:

On(a,b)	Above(a,b)	Clear(a)
On(b,c)	Above(b,c)	Clear(d)
On(d,e)	Above(a,c)	Table(c)
Top(b,a)	Above(d,e)	Table(e)
Top(c,b)	Top(e,d)	

Representation in Predicate Logic

- Remark: We use **intended Interpretation** in the **Conceptualization**
It means that we make all statements **True** in the **intended interpretation**
- We can then **ADD** some **rules** describing
- general properties of our **Universe**
- **Rules : Axioms of our Universe**
- $\forall x \forall y (\text{On}(x,y) \Rightarrow \text{Above}(x,y)) .$
 - $\forall x \forall y ((\text{Above}(x,y) \wedge \text{Above}(y,z)) \Rightarrow \text{Above}(x,z)) .$
 - etc

Reasoning in Prolog : Resolution

- To be able to use **Prolog** we have to convert all statement into a “**non quantifier**” form
-
- This process is called **Skolemization**
- Good **Prolog compiler** does it for us
- **Resolution** is the **Inference Engine** of **Prolog**

Plan for Logic Part

1. Short **Introduction** and **Overview** to **Predicate Logic**
2. Laws of Quantifiers
3. **Propositional Resolution**
4. **Resolution Strategies** (to go faster!)
5. **Skolemization** -reduction predicate logic to propositional logic
6. **Predicate Resolution**- introduction

Example

- **Conceptualize** the following situation using **Nilsson's definition**
- *In a room there are 2 cats, 3 dogs, and 2 kind of food– one for cats and one for dogs.*
- **The following properties must be true.**
 1. *One cat likes all dogs.*
 2. *One dog hates all cats.*
 3. *Everybody (cats and dogs) like all food.*
 4. *One dog hates cat food.*
 5. *All cats hate dog food.*

Example: Notation

- **We use the following notation**

- **U** – Universe of discourse is the set

$$U = \{ o1, o2, o3, o4, o5, o6, o7 \}$$

- **R** – Relational Basis Set; Set of relations

$$R = \{ \text{CAT, DOG, FOOD, CFOOD, DFOOD, LIKE, HATE} \}$$

- **WE USE INTENDED Interpretation, i.e.**

- Relation **CAT** is defined intuitively by a property **x is a cat**
- Relation **DOG** is defined intuitively by a property **x is a dog**
- Relation **FOOD** is defined intuitively by a property **x is food**
- Relation **CFOOD** is defined intuitively by a property **x is cat food**
- Relation **DFOOD** is defined intuitively by a property **x is dog food**
- Relation **LIKE** is defined intuitively by a property **x likes y**
- Relation **LIKE** is defined intuitively by a property **x likes y**
- Relation **HATE** is defined intuitively by a property **x hates y**

Example: Relations

Remark that the relations

CAT, DOG, FOOD, CFOOD, DFOOD

are *one argument relations* and

the relations

LIKE, HATE

are *two argument relation* and

all of them are **defined on** the Universe **U**

Example: Relations Definition

- We define, for example the relation **CAT** $\subseteq U$ (one argument relation) as
 - **CAT** = { o1, o2 }
 -
- We define, for example the relation **DOG** $\subseteq U$
- (one argument relation) as
 - **DOG** = { o3, o4, o5 }
- Observe that the sets **CAT** and **DOG** must be **disjoint**- as we use the **intended interpretation**

Example: Relations Definition

- Observe that the sets **CAT**, **DOG** and **FOOD** must also be **disjoint-** as we use the **intended interpretation**
- We must define now the relation **FOOD** \subseteq **U**
- (one argument relation) as
- **FOOD** = { o6, o7 }
- We define, for example the one argument relations
- **CFOOD** \subseteq **FOOD** \subseteq **U**, **DFOOD** \subseteq **FOOD** \subseteq **U**, as
- **CFOOD** = { o7 }, **DFOOD** = { o6 }
- Observe that the sets **CFOOD** and **DFOOD** must be **disjoint-** as we use the **intended interpretation**

DEFINITION of the relations LIKE, HATE

- Relations **LIKE, HATE** are defined intuitively by respective properties: *x likes y* and *x hates y*
- Both are 2 argument relation defined on **U**, i.e.
- **LIKE** \subseteq **UxU** and **HATE** \subseteq **UxU**

and must fulfill the following properties:

- 1. One cat likes all dogs.*
- 2. One dog hates all cats.*
- 3. Everybody (cats and dogs) like all FOOD.*
- 4. One dog hates cat food.*
- 5. All cats hate dog food*

Definitions of the relations **LIKE**, **HATE**

- Observe that the relations **LIKE** and **HATE** in order to fulfill the conditions **1.-5.** must be defined differently on different subsets of **U**.
- We define first appropriate parts
- **LIKE1, LIKE2** of the relation **LIKE** that correspond to properties **1., 3.** and define **LIKE** as set union of all of them, i.e. we put
- $$\mathbf{LIKE = LIKE1 \vee LIKE2}$$

Definition of the relation **LIKE**

- **PROPERTIES**

- *1. One cat likes all dogs*

- We define **LIKE1** as follows

- **$LIKE1 \subseteq CAT \times DOG \subseteq U \times U$**

- **$LIKE1 \subseteq \{o1, o2\} \times \{o3, o4, o5\} \subseteq U \times U$**

- We put

- **$LIKE1 = \{(o2, o3), (o2, o4), (o2, o5)\}$**

- Observe that there are many ways of defining **LIKE1** – this is just my choice

Definition of the relation **LIKE**

- **PROPERTIES**

- *3. Everybody (cats and dogs) like all FOOD*

We define **LIKE2** as follows

- **$LIKE2 \subseteq (CAT \vee DOG) \times FOOD \subseteq U \times U$**
- **$LIKE1 \subseteq \{o1, o2, o3, o4, o5\} \times \{o6, o7\} \subseteq U \times U$**
- We put
- **$LIKE2 = \{o1, o2, o3, o4, o5\} \times \{o6, o7\}$**
 $LIKE = LIKE1 \vee LIKE2$

Definition of the relation **HATE**

- We define first appropriate parts
- **HATE1, HATE2, HATE3** of the relation **HATE** that correspond to properties **2., 4., 5.** and define **HATE** as set union of all of them, i.e. we put
- **$HATE = HATE1 \vee HATE2 \vee HATE3$**

Definition of the relation **HATE**

- **PROPERTIES**
- *2. One dog hates all cats.*
- We define **HATE1** as follows
- **$HATE1 \subseteq DOG \times CAT \subseteq U \times U$**
- **$HATE1 \subseteq \{o3, o4, o5\} \times \{o1, o2\} \subseteq U \times U$**
- We put, for example
- **$HATE1 = \{(o5, o1), (o5, o2)\}$**
- Observe that there are *many ways* of defining **HATE1** – this is just my choice

Definition of the relation **HATE**

- **PROPERTIES**
- *4. One dog hates cat food.*
- We define **HATE2** as follows
- **$HATE2 \subseteq DOG \times CFOOD \subseteq U \times U$**
- **$HATE2 \subseteq \{o3, o4, o5\} \times \{o7\} \subseteq U \times U$**
- We put, for example
- **$HATE2 = \{(o3, o7)\}$**
- Observe that there are many ways of defining **HATE2** – this is just my choice

Definition of the relation **HATE**

- **PROPERTIES**
- *5. All cats hate dog food*
- We define **HATE3** as follows
- **$\text{HATE3} \subseteq \text{CAT} \times \text{DFOOD} \subseteq U \times U$**
- **$\text{HATE3} \subseteq \{o1, o2\} \times \{o6\} \subseteq U \times U$**
- We put **$\text{HATE3} = \{(o1, o7), (o2, o7)\}$**
and
- **$\text{HATE} = \text{HATE1} \vee \text{HATE2} \vee \text{HATE3}$**
- Observe that there is only *one way* of defining **HATE3**

Exercise

- Write all definitions from the **Example** as **Prolog** like **Facts** about our **Universe**

Add few **Rules** governing the **Universe**

History: Major AI Areas

1. Game Playing:

In early 1950 **Claude Shannon (1950)** and **Alan Turing (1953)** were writing **chess programs** for von Neumann computers

But, in fact **Shannon** had no real computer to work with, and

Turing was denied access to his own team's computers by the British government on the grounds that

research into AI was frivolous !

History: Search as AI

- Search as a Major AI Technique:
Search is a **problem solving** technique that systematically explores a space of **problem states**, i.e., stages of problem solving process.
 - Example:
Different board configurations in a game form a space of alternative solutions. The space is then searched to find a final answer.

History: Search as AI

- Much of early research in State Space Search was done using common board games: checkers, chess, 16-puzzle
- Games have well defined rules, and hence it is easy to generate the search space
- Large space – Heuristic Search
- 1984 book by Pearl , “Heuristics” – First Comprehensive Mathematical treatment of heuristic search
- **Heuristic Search** is widely used now in Theorem Proving, Machine Learning, Data Mining and Big Data

Heuristic Search became now a newly vibrant area of research

History: Major AI Areas

2. Automated Reasoning and Theorem Proving:

Origin: Foundations of Mathematics.

Mathematics can be considered as “axiomatic theory.”

- Hilbert Program (1910) – to formalize **all** of mathematics in such a way that a proof of **any** theorem can be found automatically.
- Gentzen(1934) – **positive answer** for **Propositional Logic**
Partial (semi-decidable) answer for **First Order Logic**

History: Major AI Areas

Automated Reasoning and Theorem Proving

- Gödel (1933) – negative answer for **arithmetic**; incompleteness theorem
- Robinson (1965) – Resolution
- Program Verification – uses theorem proving techniques

History: Major AI Areas

3. Expert Systems:

- Obtaining knowledge from human experts, or databases (automated rules generators) and representing it in a form that computer may apply to similar problems
- **Rule Based Systems.**
- **Expert Systems** grew into information systems
- **Expert Systems** are always developed for a specific domain

History: Expert Systems

- First Examples:

- Dendral, Stanford 1960:

- built to infer the structure of organic molecules from their chemical formulas.

- MYCIN, Stanford 1970:

- diagnostic system, plus prescribes treatment for Spinal Meningitis and bacterial infection in the blood. It was the first program to address the problem of reasoning with **uncertain** and/or **incomplete** information.

- Still on the Web ! (Medical Information Systems.)

Expert Systems

(Our Handout #1 – Modern Approach)

[Jerzy Busse, Managing Uncertainty in E.S., Kluwer, NY](#)

1. Knowledge acquisition by using **Machine Learning**
2. **Rule Induction** from databases. (Rough Sets approach)
3. **Uncertainties in Quantitative approach:**
 - Bayes rules and network (probabilistic approach)
 - Belief networks. (probabilistic)
 - Dempster – Shafer Theory:
Dempster Rules.

Managing Uncertainty in E.S.

3. Uncertainties – Quantitative Approaches

- Fuzzy Sets (Zadek, 1965)
- Rough Sets (Pawlak, 1985)
- Machine learning / data mining techniques.

4. Uncertainties – Qualitative Approaches

- Modal Logics.
- Non-monotonic logics.
- Default logic
- Plausible Reasoning.

Early Expert Systems

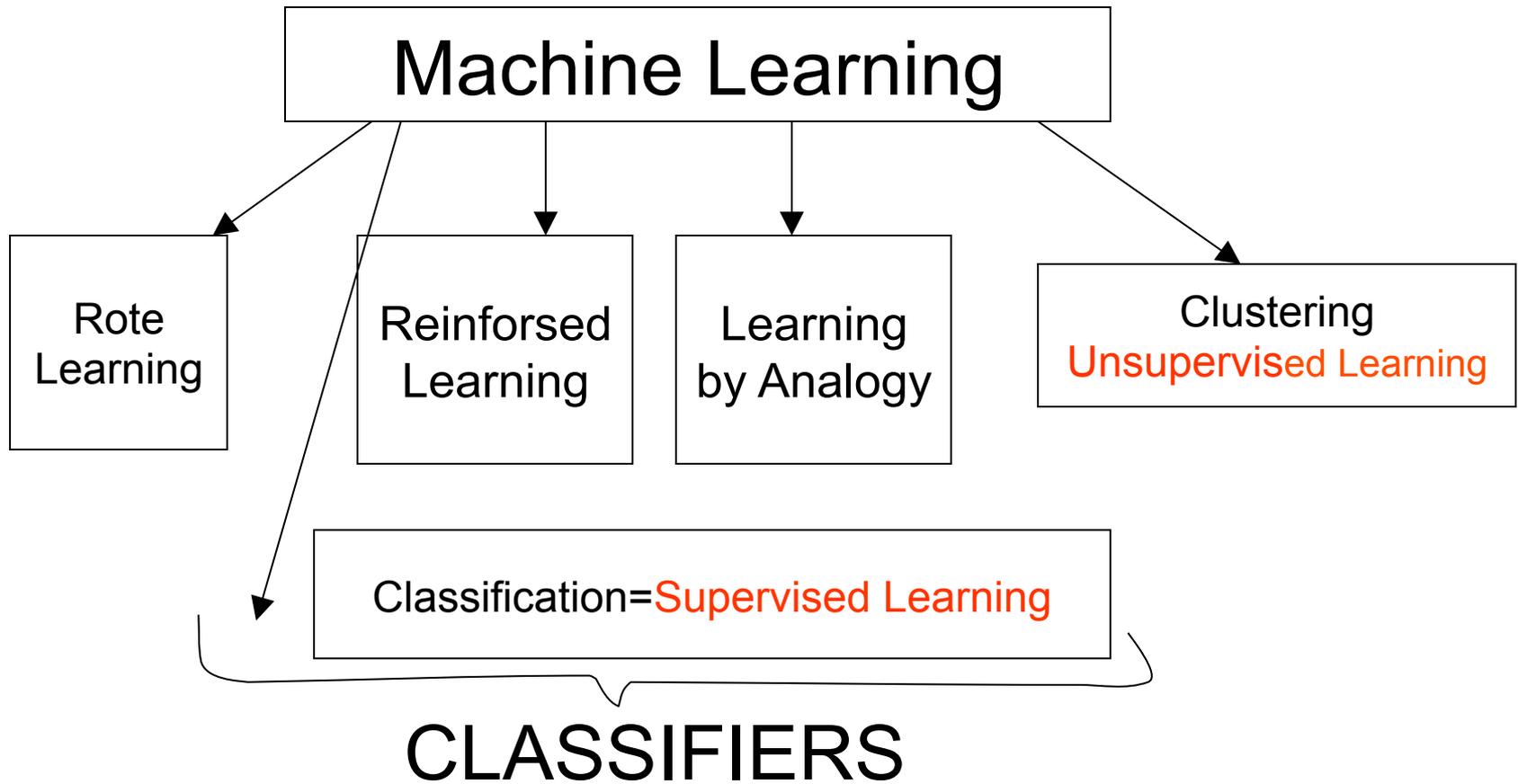
MYCIN Story:

MYCIN asked if the patient was **pregnant** even though it has been told that the patient was **male**.

Modern Expert Systems

- **Modern Expert Systems** always have Machine Learning components.
- **Supervised (Classification) Learning** in large databases is called **Data Mining**.
- **Supervised Learning Techniques are:**
 - 1) Genetic Algorithms. (Evolutionary)
 - 2) Neural Networks
 - 3) Decision Tree
 - 4) Rough Sets
 - 5) Classification by Association

Some Types of Machine Learning



Other AI Areas

- Natural Language Processing.
- Natural Language Understanding
- Robotics
- Intelligent Visualization.

AI: Very Short History

- **The name, “AI”** , was suggested in 1956 by McCarthy (at Dartmouth at that time, and then at Stanford, Yale) during a **two month long** workshop at Dartmouth
- The Workshop was devoted to programs that could perform:
 - Elementary Reasoning Tasks
 - Proving Simple Theorems.
 - Answering Simple Questions.
 - Playing Board Games.
 - ALL Non computational (in a sense of numbers) tasks
 - -revolutionary at that time

Short History

- All together there were **10 people**. For the next 20+ years the field would be dominated by them, their students and colleagues at MIT, CMU (Carnegie-Mellon University) , Stanford and IBM.
- **Allen Newell** and **Herbert Simon** from **CMU** stole the show with **Logic Theorist (LT)** – first program to think non-numerically.

Very Short History

- LT proved most of the theorems in Chapter 2 of Russell and Whitehead's "Principia Mathematica"
- Herb Gelernter (Stony Brook) constructed **first (1959)** Geometry Theorem Prover
-
- Anita Wasilewska (now Stony Brook) invented and wrote
- **first theorem prover** (in LISP-ALGOL) for **MODAL LOGIC** in 1967 at Warsaw University, Poland
- Now **Theorem Proving** is a separate field of Computer Science
-

Very Short History

- 1952-1969 : Time of **Early Enthusiasm** and **Great Expectations**
- 1952 :
Arthur Samuel wrote a tournament level checkers program
- In **February 1956** the program was demonstrated on **National TV**
- **A. Samuel**, like **Alan Turing** had a hard time to obtain computer time; worked only at night

Short History

- 1958 :
McCarthy moved from Dartmouth to MIT and invented LISP - Second **oldest** programming language still in use; Which is the Oldest?
- LISP is now being replaced by Prolog as a dominant AI language (in many areas)
- McCarthy and his group also invented Timesharing and formed Digital Equipment Corporation (DEC) to produce time sharing computers

Very Short History

- 1958 :
 - Marvin Minsky moved to MIT - hee represented Anti-logic outlook.
 - McCarthy was Pro-logic and moved to Stanford
 - McCarthy's **Logic agenda** was busted by Robinson's discovery of Resolution and Kowalski's work on Prolog - Logic Programming“
 - McCarthy founded SRI - Stanford Research Institution – still main place for research in **general purpose methods** for logical reasoning

Very Short History

- 1963:
 - J. Slagle's program SAINT was able to solve closed form integration problems. (first year calculus)
- 1969:
 - Green's Question – Answering and Planning Systems.
 - Shakey's Robotics Projects; first integration of logical reasoning and physical activity
- 1971:
 - D. Huffman's "vision" project - rearrangement of the blocks, put on top of the table, using a robot hand that picked one block at a time
- 1970:
 - P. Winston – first learning theory

Very Short History

- 1972:
T. Winograd – first natural language understanding theory
- 1974:
Planner of Scott Fahlman
- 1966 – 1974:
A Dose of Reality
- 1966:
All **American** Governmental funding for **machine translations** were **cancelled**
- 1973:
British Government **stopped AI support** to all but 2 universities

Knowledge-Based Systems the (1969-79)

- Narrow the area of expertise and then solve.
- Dendral (1969):
 - Buchanan, a philosopher turned Computer Scientist, and Joshua Lederberg (a nobel geneticist) at Stanford, brought forward the first successful knowledge-intensive system, “Dendral”.
 - Knowledge base is a large number of special purpose rules.
 - With Dendral, there is a clean separation of the knowledge base (Rules) and the reasoning component. (following McCarthy.)

Very Short History

- **Genetic Algorithms** were formulated in 1958-59, but computers were not yet up to it
- The same happened to **Neural Networks** – mathematical model and theoretical research was rampant, but for years computers were not strong and fast enough to give meaningful results
- 1980 – back propagation (NN) algorithm was invented and **first applications** followed

AI becomes an Industry

1982:

First successful **Expert System RI** at **Digital Equipment Corporation (DEC)** was made (**McDermot**)

RI helped configure orders for new Computer Systems and by **1986** was **saving** the company **\$40 million a year**

1988:

DEC's AI group had **40 Expert systems**

Du Pont had **100 E.S.** in use and **500 in development** **saving \$10 million a year**

Information Systems Departments were crated in Industries and at **Universities**

Industry went from a **few million** in **sales** in **1980** to **2 Billion** in **1988**

History: AI becomes an Industry

- 1981:

Japan announced **Fifth Generation** project

The **Fifth Generation Project** was created to use **Prolog** to achieve full-scale **natural language understanding**

USA formed a company **MCC** (Microelectic and Computer Technology Corporation) **to compete** with **Japan**
ALSO: **Cornegie Group, Inference, Intellicop, Lisp Machines**

- **Fifth Generation Project** generated a progress but the **project failed**
- **Prolog** is just one of many programming languages
- **Prolog** is still prominent in **Linguistics** and **Natural Language processing** and **translation**

PROLOG – Logic Programming

Short History

- 1964–65 :
Robinson, (Syracuse University),
introduced Resolution.
- 1968–70:
Kowalski, University of Edinburgh, England,
created first version of Prolog.
- David Warren (British) made the prolog machine.
- Stony Brook's D. Warren was a president of
Association for Logic Programming. Prominent !

Philosophical Issues

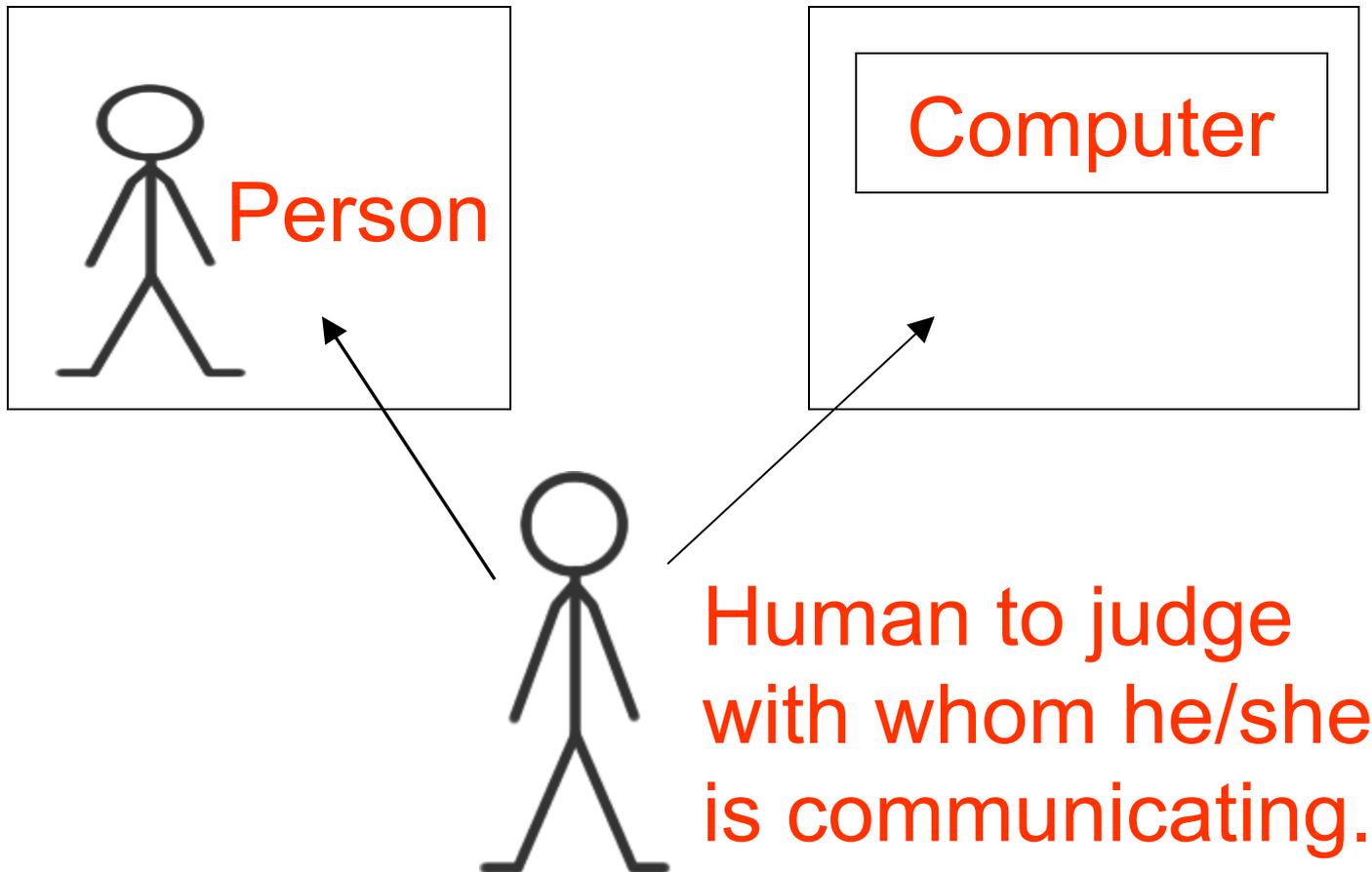
- AI research makes the assumption that human intelligence can :
 - 1) be reduced to the (complex) manipulations of symbols, and
 - 2) It does not matter what Medium is used to manipulate these symbols. (It does not have to be a biological brain.)

Philosophical Issues

- Turing's test (1950) and Charles Searle's Chinese room (1980) are the most known contributions to the philosophical debate.
- **Loebner Prize Contest** – competition started in 1991 & inspired by Turing test.
 - Best program – \$2,000.
 - Grand Prize - \$100,000. (No one claimed yet.)

History: Philosophical Issues

- Turing Test:



History: Philosophical Issues

- Turing test :
 - The human can ask the other person/ computer any question; including very subjective questions, like:
 - “What is your favorite color?” and
 - “What do you think of the privatization of the British Rail” ?
 - **Computer is Intelligent** – If it answers so well that the judge cannot distinguish it from a person.

History: Philosophical Issues

- Searle “Chinese Room” critique:

If you have a huge rule book on Chinese you can “talk” Chinese even if you do not understand it;

so, **Turing Test** **does not** measure the Intelligence

Questions – Answers Systems. The Loebner Prize Contest

- 10 judges type questions at a computer which are responded by a person or a program.
- The best program receives \$2000.
- The Winning program : \$100,000.
 - No one won yet, but one program fooled 5 judges out of 10.

Questions – Answers Systems. The Loebner Prize Contest

- 1996 Winning Fragment:

P: No worries ! Hey! I did not come here to be subjected to bad language! How rude.

J2: Sorry, sorry. Did not mean to offend. How old are you?

The Loebner Prize Contest

1996 Winning Fragment

P: That is ok. Just don't do it again.
Well, should my age really matter?
Just for the record, I am 24. How old
are you then?

J2: I am 21.

P: If you are 21, it is your business! Would
you like to hear a joke? Knock, knock.

Questions – Answers Systems.

- Programs like these don't really understand the conversation. They use various **stores of amusing replies and use keywords in the question to select the reply.**
- Example: 1960 a program, "ELIZA" attempted to imitate a psychotherapist.