21PCA101: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

UNIT- V

- Overview of AI application areas. AI as Representation and search

✓ THE PREDICATE CALCULUS:

- In propositional calculus, each atomic symbol (P, Q, etc.) denotes a single proposition. There is no way to access the components of an individual assertion. Predicate calculus provides this ability. For example, instead of letting a single propositional symbol, P, denote the entire sentence "it rained on Tuesday," we can create a predicate weather that describes a relationship between a date and the weather: weather (tuesday, rain).

- Through inference rules we can manipulate predicate calculus expressions, accessing their individual components and inferring new sentences.
- Predicate calculus also allows expressions to contain variables. Variables let us create general assertions about classes of entities. For example, we could state that for all values of X, where X is a day of the week, the statement weather (X, rain) is true; i.e., it rains every day. As we did with the propositional calculus, we will first define the syntax of the language and then discuss its semantics.

✓ The Syntax of Predicates and Sentences

- Before defining the syntax of correct expressions in the predicate calculus, we define an alphabet and grammar for creating the *symbols* of the language. This corresponds to the lexical aspect of a programming language definition.
- Predicate calculus symbols, like the *tokens* in a programming language, are irreducible syntactic elements: they cannot be broken into their component parts by the operations of the language.
- In our presentation we represent predicate calculus symbols as strings of letters and digits beginning with a letter. Blanks and nonalphanumeric characters cannot appear within the string, although the underscore, _, may be used to improve readability.

D E F I N I T I O N

PREDICATE CALCULUS SYMBOLS

The alphabet that makes up the symbols of the predicate calculus consists of:

1. The set of letters, both upper- and lowercase, of the English alphabet.
2. The set of digits, 0, 1, , 9.
3. The underscore, _.

Symbols in the predicate calculus begin with a letter and are followed by any sequence of these legal characters.

Legitimate characters in the alphabet of predicate calculus symbols include

a R 6 9 p _ z

Examples of characters not in the alphabet include

% @ / &

Legitimate predicate calculus symbols include

George fire3 tom_and_jerry bill XXXX friends_of

Examples of strings that are not legal symbols are

3jack no blanks allowed ab%cd ***71 duck!!!

- Symbols, are used to denote objects, properties, or relations in a world of discourse. As with most programming languages, the use of "words" that suggest the symbol's intended meaning assists us in understanding program code. Thus, even though l (g, k) and likes (george, kate) are formally equivalent (i.e., they have the same structure), the second can be of great help (for human readers) in indicating what relationship the expression represents.
- It must be stressed that these descriptive names are intended solely to improve the readability of expressions. The only "meaning" that predicate calculus expressions have is given through their formal semantics.

- Parentheses "()", commas ",", and periods "." are used solely to construct well-formed expressions and do not denote objects or relations in the world. These are called *improper symbols*.

- Predicate calculus symbols may represent either *variables, constants, functions,* or *predicates*. Constant's name specific objects or properties in the world. Constant symbols must begin with a lowercase letter. Thus george, tree, tall, and blue are examples of well-formed constant symbols. The constants true and false are reserved as *truth symbols*. Variable symbols are used to designate general classes of objects or properties in the world. Variables are represented by symbols beginning with an uppercase letter.
- ThusGeorge, BILL, and KAte are legal variables, whereas geORGE and bill are not. Predicate calculus also allows functions on objects in the world of discourse. Func- tion symbols (like constants) begin with a lowercase letter. Functions denote a mapping of one or more elements in a set (called the *domain* of the function) into a unique element of a second set (the *range* of the function).
- Elements of the domain and range are objects in the world of discourse. In addition to common arithmetic functions such as addition and multiplication, functions may define mappings between nonnumeric domains.
- Note that our definition of predicate calculus symbols does not include numbers or arithmetic operators. The number system is not included in the predicate calculus primi- tives; instead, it is defined axiomatically using "pure" predicate calculus as a basis (Manna and Waldinger 1985). While the particulars of this derivation are of theoretical interest, they are less important to the use of predicate calculus as an AI representation language. For convenience, we assume this derivation and include arithmetic in the language.
- Every function symbol has an associated *arity*, indicating the number of elements in the domain mapped onto each element of the range.
- Thus, father could denote a function of arity 1 that maps people onto their (unique) male parent. plus, could be a function of arity 2 that maps two numbers onto their arithmetic sum.
- A *function expression* is a function symbol followed by its arguments. The

arguments are elements from the domain of the function; the number of arguments is equal to thearity of the function. The arguments are enclosed in parentheses and separated by com- mas.

- For example,

f(X,Y)

father(david)

price(bananas)

are all well-formed function expressions.

D E F I N I T I O N

SYMBOLS and TERMS

Predicate calculus symbols include:

1. *Truth symbols* true and false (these are reserved symbols).
2. *Constant symbols* are symbol expressions having the first character lowercase.
3. *Variable symbols* are symbol expressions beginning with an uppercase character.
4. *Function symbols* are symbol expressions having the first character lowercase. Functions have an attached arity indicating the number of elements of the domain mapped onto each element of the range.

A *function expression* consists of a function constant of arity n, followed by n terms,

t_1, t_2, t_n, enclosed in parentheses and separated by commas.

A predicate calculus *term* is either a constant, variable, or function expression.

Thus, a predicate calculus *term* may be used to denote objects and properties in a problem domain. Examples of terms are:

cat times (2,3) X

blue mother(sarah) kate

D E F I N I T I O N

- **PREDICATES and ATOMIC SENTENCES**

  - Predicate symbols are symbols beginning with a lowercase letter.
  - Predicates have an associated positive integer referred to as the *arity* or "argument number" for the predicate. Predicates with the same name but different arities are considered distinct.

An atomic sentence is a predicate constant of arity n, followed by n terms, t_1, t_2, t_n, enclosed in parentheses and separated by commas.

The truth values, true and false, are also atomic sentences.

Atomic sentences are also called ***atomic expressions***, ***atoms***, **or** ***propositions***.

- We may combine atomic sentences using logical operators to form *sentences* in the predicate calculus. These are the same logical connectives used in propositional calculus: ∧, ∨, ¬, →, and ≡.
- The *universal quantifier*, ∀, indicates that the sentence is true for all values of the variable. In the example, ∀ X likes (X, ice_cream) is true for all values in the domain of the defini- tion of X. The *existential quantifier*, ∃, indicates that the sentence is true for at least one value in the domain. ∃Y friends (Y, peter) is true if there is at least one object, indicated by Y that is a friend of peter. Quantifiers are discussed in more Sentences in the predicate calculus are defined inductively.

D E F I N I T I O N

- **PREDICATE CALCULUS SENTENCES**

Every atomic sentence is a sentence.

1. If s is a sentence, then so is its negation, ¬ s.
2. If s_1 and s_2 are sentences, then so is their conjunction, $s_1 \wedge s_2$.
3. If s_1 and s_2 are sentences, then so is their disjunction, $s_1 \vee s_2$.
4. If s_1 and s_2 are sentences, then so is their implication, $s_1 \rightarrow s_2$.
5. If s_1 and s_2 are sentences, then so is their equivalence, $s_1 \equiv s_2$.

6. If X is a variable and s a sentence, then ∀ X s is a sentence.

7. If X is a variable and s a sentence, then ∃X s is a sentence.

- The definition of predicate calculus sentences and the examples just presented suggest a method for verifying that an expression is a sentence. This is written as a recursive algorithm, verify_sentence. verify_sentence takes as argument a candidate expression and returns success if the expression is a sentence.

```
function verify_sentence(expression); begin
  case
    expression is an atomic sentence: return SUCCESS;
    expression is of the form Q X s, where Q is either r ∀ or ∃, X is a variable,
      if verify_sentence(s) returns SUCCESS
      then return SUCCESS else return
      FAIL;
    expression is of the form ¬ s:
      if verify_sentence(s) returns SUCCESS then
      return SUCCESS
      else return FAIL;
    expression is of the form s1 op s2, where op is a binary logical operator: if
      verify_sentence(s1) returns SUCCESS and
        verify_sentence(s2) returns SUCCESS
      then return SUCCESS
      else return FAIL; otherwise:
    return FAIL
  end end.
```

- **THE PROPOSITIONAL CALCULUS**:
  - ✓ Symbols and Sentences

- The propositional calculus and, in the next subsection, the predicate calculus are first of all languages. Using their words, phrases, and sentences, we can represent and reason about properties and relationships in the world. The first step in describing a language is to introduce the pieces that make it up: it's set of symbols.

D E F I N I T I O N

PROPOSITIONAL CALCULUS SYMBOLS

The *symbols* of propositional calculus are the propositional symbols:

P, Q, R, S, …

truth symbols:

true, false

and connectives:

∧, ∨, ¬, →, ≡

- Propositional symbols denote *propositions*, or statements about the world that may be either true or false, such as "the car is red" or "water is wet." Propositions are denoted by uppercase letters near the end of the English alphabet. Sentences in the propositional cal- culus are formed from these atomic symbols according to the following rules:

D E F I N I T I O N

PROPOSITIONAL CALCULUS SENTENCES

Every propositional symbol and truth symbol is a sentence.

For example: true, P, Q, and R are sentences.

The *negation* of a sentence is a sentence.

For example: ¬ P and ¬false are sentences.

The *conjunction*, or *and*, of two sentences is a sentence.

For example: $P \wedge \neg P$ is a sentence.

The *disjunction*, or *or*, of two sentences is a sentence.

For example: $P \vee \neg P$ is a sentence.

The *implication* of one sentence from another is a sentence.

For example: $P \rightarrow Q$ is a sentence.

The *equivalence* of two sentences is a sentence.

For example: $P \vee Q \equiv R$ is a sentence.

Legal sentences are also called *well-formed formulas* or *WFFs*.

- In expressions of the form $P \wedge Q$, P and Q are called the *conjuncts*. In $P \vee Q$, P and Q are referred to as *disjuncts*. In an implication, $P \rightarrow Q$, P is the *premise* or *antecedent* and Q, the *conclusion* or *consequent*.
- An expression is a sentence, or well-formed formula, of the propositional calculus if and only if it can be formed of legal symbols through some sequence of these rules. For example,
  $((P \wedge Q) \rightarrow R) \equiv \neg P \vee \neg Q \vee R$

is a well-formed sentence in the propositional calculus because:

P, Q, and R are propositions and thus sentences.

$P \wedge Q$, the conjunction of two sentences, is a sentence.

$(P \wedge Q) \rightarrow R$, the implication of a sentence for another, is a sentence.

$\neg P$ and $\neg Q$, the negations of sentences, are sentences.

$\neg P \vee \neg Q$, the disjunction of two sentences, is a sentence.

$\neg P \vee \neg Q \vee R$, the disjunction of two sentences, is a sentence.

$((P \wedge Q) \rightarrow R) \equiv \neg P \vee \neg Q \vee R$, the equivalence of two sentences, is a sentence.

- This is our original sentence, which has been constructed through a series of applications of legal rules and is therefore “well formed”.

✓ The Semantics of the Propositional Calculus

presented the syntax of the propositional calculus by defining a set of rules for producing legal sentences. In this section we formally define the *semantics* or “meaning” of these sentences. Because AI programs must reason with their representational structures, it is important to demonstrate that the truth of their conclusions depends only on the truth of their initial knowledge or premises, i.e., that log- ical errors are not introduced by the inference procedures. A precise treatment of seman- tics is essential to this goal.

A proposition symbol corresponds to a statement about the world. For example, P may denote the statement “it is raining” or Q, the statement “I live in a brown house.” A proposition must be either true or false, given some state of the world. The truth value assignment to propositional sentences is called an *interpretation*, an assertion about their truth in some *possible world*.

- Formally, an interpretation is a mapping from the propositional symbols into the set

{T, F}. As mentioned in the previous section, the symbols true and false are part of the set of well-formed sentences of the propositional calculus; i.e., they are distinct from the truth value assigned to a sentence. To enforce this distinction, the symbols T and F are used for truth value assignment.

➢ **D E F I N I T I O N**

PROPOSITIONAL CALCULUS SEMANTICS

An *interpretation* of a set of propositions is the assignment of a truth value, either T or F, to each propositional symbol.

The symbol true is always assigned T, and the symbol false is assigned F. The interpretation or truth value for sentences is determined by:

The truth assignment of *negation*, ¬ P, where P is any propositional symbol, is F if the assignment to P is T, and T if the assignment to P is F.

The truth assignment of *conjunction*, ∧, is T only when both conjuncts have truth value T; otherwise, it is F.

The truth assignment of *disjunction*, ∨, is F only when both disjuncts have truth

value F; otherwise, it is T.

The truth assignment of *implication*, →, is F only when the premise or symbol before the implication is T and the truth value of the consequent or symbol after the implication is F; otherwise, it is T.

The truth assignment of *equivalence*, ≡, is T only when both expressions have the same truth assignment for all possible interpretations; otherwise, it is F.

For propositional expressions P, Q, and R:

¬ (¬ P) ≡ P (P ∨ Q) ≡ (¬ P → Q)

the contrapositive law: (P → Q) ≡ (¬ Q → ¬ P)

de Morgan's law: ¬ (P ∨ Q) ≡ (¬ P ∧ ¬ Q) and ¬ (P ∧ Q) ≡ (¬ P ∨¬ Q)

the commutative laws: (P ∧ Q) ≡ (Q ∧ P) and (P ∨ Q) ≡ (Q ∨ P)

the associative law: ((P ∧ Q) ∧ R) ≡ (P ∧ (Q ∧ R))

associative law: ((P ∨ Q) ∨ R) ≡ (P ∨ (Q ∨ R))

the distributive law: P ∨ (Q ∧ R) ≡ (P ∨ Q) ∧ (P ∨ R)

the distributive law: P ∧ (Q ∨ R) ≡ (P ∧ Q) ∨ (P ∧ R)

P Q P Q ∧

T	T	T
T	F	F
F	T	F
F	F	F

Truth table for the operator ∧.

P Q ¬P ¬P ∨ Q P ∨ Q (¬P ∨ Q)=(P ∨ Q)

T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

Figure Truth table demonstrating the equivalence of P → Q and ¬ P ∨ Q.

Application: A Logic-Based Financial Advisor

- As a final example of the use of predicate calculus to represent and reason about problem domains, we design a financial advisor using predicate calculus. Although this is a simple example, it illustrates many of the issues involved in realistic applications.
- The function of the advisor is to help a user decide whether to invest in a savings account or the stock market. Some investors may want to split their money between the two. The investment that will be recommended for individual investors depends on their income and the current amount they have saved according to the following criteria:

1. Individuals with an inadequate savings account should always make increasing the amount saved their first priority, regardless of their income.
2. Individuals with an adequate savings account and an adequate income should consider a riskier but potentially more profitable investment in the stock market.
3. Individuals with a lower income who already have an adequate savings account may want to consider splitting their surplus income between savings and stocks, to increase the cushion in savings while attempting to increase their income through stocks.

The adequacy of both savings and income is determined by the number of dependents an individual must support. Our rule is to have at least $5,000 in savings for each dependent. An adequate income must be a steady income and supply at least $15,000 per year plus an additional $4,000 for each dependent.

To automate this advice, we translate these guidelines into sentences in the predicate calculus. The first task is to determine the major features that must be considered. Here, they are the adequacy of the savings and the income. These are represented by the predicates savings_account and income, respectively. Both of these are unary predicates, and their argument could be either adequate or inadequate. Thus,

```
savings_account(adequate).
savings_account(inadequate).
income(adequate).
```

income(inadequate). are their possible
values.

- Conclusions are represented by the unary predicate investment, with possible values of its argument being stocks, savings, or combination (implying that the investment should be split).
- Using these predicates, the different investment strategies are represented by implications. The first rule, that individuals with inadequate savings should make increased savings their main priority, is represented by

savings_account(inadequate) → investment(savings).

Similarly, the remaining two possible investment alternatives are represented by

savings_account(adequate) ∧ income(adequate) → investment(stocks).
savings_account(adequate) ∧ income(inadequate) → investment(combination).

- Next, the advisor must determine when savings and income are adequate or inade- quate. This will also be done using an implication. The need to do arithmetic calculations requires the use of functions. To determine the minimum adequate savings, the function minsavings is defined. minsavings takes one argument, the number of dependents, and returns 5000 times that argument.

Using minsavings, the adequacy of savings is determined by the rules
∀ X amount_saved(X) ∧ ∃ Y (dependents(Y) ∧ greater (X, minsavings(Y)))
→ savings_account(adequate). ∀ X amount_saved(X) ∧ ∃ Y (dependents(Y) ∧ ¬ greater (X, minsavings(Y))) → savings_account(inadequate).
where minsavings(X) ≡ 5000 ∗ X.

- In these definitions, amount_saved(X) and dependents(Y) assert the current amount in savings and the number of dependents of an investor; greater (X, Y) is the standard arithmetic test for one number being greater than another and is not formally defined in this example.
  Similarly, a function minincome is defined as

minincome(X) ≡ 15000 + (4000 ∗ X).

- minincome is used to compute the minimum adequate income when given the number of dependents. The investor's current income is represented by a predicate, earnings. Because an adequate income must be both steady and above the minimum, earnings take two arguments: the first is the amount earned, and the second must be equal to either steady or unsteady. The remaining rules needed for the advisor are

  ∀ X earnings (X, steady) ∧ ∃ Y (dependents(Y) ∧ greater (X, minincome(Y))) → income(adequate)

  ∀ X earnings (X, steady) ∧ ∃ Y (dependents(Y) ∧ ¬ greater (X, minincome(Y))) → income(inadequate)

  ∀ X earnings (X, unsteady) → income(inadequate).

- In order to perform a consultation, a description of a particular investor is added to this set of predicate calculus sentences using the predicates amount_saved, earnings, and dependents. Thus, an individual with three dependents, $22,000 in savings, and a steady income of $25,000 would be described by

amount_saved (22000).

earnings (25000, steady).

dependents (3).

✓ This yields a logical system consisting of the following sentences:

1. savings_account(inadequate) → investment(savings).
2. savings_account(adequate) ∧ income(adequate) → investment(stocks).
3. savings_account(adequate) ∧ income(inadequate) → investment(combination).
4. ∀ X amount_saved(X) ∧ ∃ Y (dependents(Y) ∧ greater (X, minsavings(Y))) → savings_account(adequate).
5. ∀ X amount_saved(X) ∧ ∃ Y (dependents(Y) ∧ ¬ greater (X, minsavings(Y))) → savings_account(inadequate).
6. ∀ X earnings (X, steady) ∧ ∃ Y (dependents (Y) ∧ greater (X, minincome(Y))) → income(adequate). 7. ∀ X earnings (X, steady) ∧ ∃ Y (dependents(Y) ∧ ¬ greater (X,

minincome(Y))) → income(inadequate). 8. ∀ X earnings (X, unsteady) → income(inadequate).

9. amount_saved (22000).

10. earnings (25000, steady).

11. dependents (3).

- **AI AS EMPIRICAL ENQUIRY:**

INTRODUCTION

- The most surprising aspects of work in artificial intelligence is the extent to which AI, and indeed much of computer science, turns out to be an empirical dis- cipline. This is surprising because most people initially think of these fields in terms of their mathematical, or alternatively, their engineering foundations. From the mathematicalviewpoint, sometimes termed the "neat" perspective, there is the rationalist desire to bring standards of proof and analysis to the design of intelligent computational devices.

- From the engineering, or "scruffy" perspective, the task is often viewed as simply making suc- cessful artifacts that society wants to call "intelligent". Unfortunately, or fortunately, depending upon your philosophy, the complexity of intelligent software and the ambigu- ities inherent in its interactions with the worlds of human activity frustrate analysis from either the purely mathematical or purely engineering perspectives.
- Furthermore, if artificial intelligence is to achieve the level of a science and become a critical component of the *science of intelligent systems*, a mixture of analytic and empiri- cal methods must be included in the design, execution, and analysis of its artifacts. On this viewpoint each AI program can be seen as an experiment: it proposes a question to the nat- ural world and the results are nature's response. Nature's response to our design and pro- grammatic commitments shapes our understanding of formalism, mechanism, and finally, of the nature of intelligence itself (Newell and Simon 1976).
- Unlike many of the more traditional studies of human cognition, we as designers of intelligent computer artifacts can inspect the internal workings of our "subjects". We can stop program execution, examine internal state, and

modify structure at will.

- As Newell and Simon note, the structure of computers and their programs indicate their potential behavior: they may be examined, and their representations and search algorithms under- stood. The power of computers as tools for understanding intelligence is a product of this duality. Appropriately programmed computers are capable of both achieving levels of semantic and behavioral complexity that beg to be characterized in psychological terms as well as offer an opportunity for an inspection of their internal states that is largely denied scientists studying most other intellectual life forms.
- Fortunately for continuing work in AI, as well as for establishing a science of intelli- gent systems, more modern psychological techniques, especially those related to neural physiology, have also shed new light on the many modes of human intelligence. We know now, for example, that human intelligent function is not monolithic and uniform.
- Rather it is modular and distributed. Its power is seen in the sense organs, such as the human retina, that can screen and preprocess visual information. Similarly, human learning is not a uni- form and homogenous faculty. Rather learning is a function of multiple environments and differing systems, each adapted to achieve specialized goals. fMRI analysis, along with PET scans, EEG, and allied neural physical imaging procedures, all support a diverse and cooperative picture of the internal workings of actual intelligent systems.
- If work in AI is going to reach the level of a science, we must also address important philosophical issues, especially those related to epistemology, or the question of how an intelligent system “knows” its world. These issues range from the question of what is the object of study of artificial intelligence to deeper issues, such as questioning the validity and utility of the physical symbol system hypothesis.
- Further questions include what a “symbol” is in the symbol system approach to AI and how symbols might relate to sets of weighted nodes in a connectionist model.
- We also question the role of rationalism expressed in the inductive bias seen in most learning programs and how this compares to the unfettered lack of

structure often seen in unsupervised, reinforcement, and emergent approaches to learning. Finally, we must question the role of embodiment, situatedness, and sociological bias in problem solving. We conclude our discussion of philosophicalissues by proposing a *constructivist epistemology* that fits comfortably with both our com- mitment to AI as a science as well as to AI as empirical enquiry.

➢ THE SCIENCE OF INTELLIGENT SYSTEM:

- It is not a coincidence that a major subgroup of the artificial intelligence community has focused its research on understanding *human* intelligence. Humans provide the prototypi- cal examples of intelligent activity, and AI engineers, even though they are usually *not* committed to "making programs that act like humans", seldom ignore human solutions.
- Some applications such as diagnostic reasoning are often deliberately modeled on the problem-solving processes of human experts working in that area. Even more importantly, understanding human intelligence is a fascinating and open scientific challenge in itself.
- Modern *cognitive science*, or *the science of intelligent systems* (Luger 1994), began with the advent of the digital computer, even though, as we saw in Chapter 1, there were many intellectual forebears of this discipline, from Aristotle through Descartes and Boole, to more modern theorists such as Turing, McCulloch and Pitts, the founders of the *neural net* model, and John von Neumann, an early proponent of a-life.
- The study became a sci- ence, however, with the ability to design and run experiments based on these theoretical notions, and to an important extent, this came about with the arrival of the computer. Finally, we must ask, "Is there an all-inclusive science of intelligence?" We can further ask, "Can a science of intelligent systems support construction of artificial intelligences?" In the following sections we discuss briefly how the psychological, epistemological, and sociological sciences support research and development in AI.

✓ PSYCHOLOGICAL CONSTRAINTS:

- Early research in cognitive science examined human solutions to logic problems, simple games, planning, and concept learning (Feigenbaum and Feldman 1963, Newell andSimon 1972, Simon 1981). Coincident with their work on the Logic Theorist, Newell and Simon began to compare their computational approaches with the search strategies used by human subjects.

- Their data consisted of *think-aloud protocols*, descriptions by human subjects of their thoughts during the process of devising a problem solution, such as a logic proof. Newell and Simon then compared these protocols with the behavior of the computer program solving the same problem. The researchers found remarkable similarities and interesting differences across both problems and subjects.
- These early projects established the methodology that the discipline of *cognitive sci- ence* would employ during the following decades:

1. Based on data from humans solving particular classes of problems, design a representational scheme and related search strategy for solving the problem.
2. Run the computer-based model to produce a trace of its solution behavior.
3. Observe human subjects working on these same problems and keep track of measurable parameters of their solution process, such as those found in think-aloud protocols, eye movements, and written partial results.
4. Analyze and compare the human and computer solutions.
5. Revise the computer model for the next round of tests and comparisons with the human subjects.

- This empirical methodology is described in Newell and Simon's Turing award lecture, quoted at the beginning of this chapter. An important aspect of cognitive science is the use of experiments to validate a problem-solving architecture, whether it be a production sys- tem, connectionist, emergent, or an architecture based on the interaction of distributed agents.
- In recent years, an entirely new dimension has been added to this paradigm. Now, not just programs can be deconstructed and observed in the act of problem solving,

but humans and other life forms can be as well. A number of new imaging techniques have been included in the tools available for observing cortical activity. These include *magne- toencephalography* (MEG), which detects the magnetic fields generated by populations of neurons. Unlike the electrical potentials generated by these populations, the magnetic field is not smeared by the skull and scalp, and thus a much greater resolution is possible.

- A second imaging technology is *positron emission tomography*, or PET. A radioactive substance, typically O^{15} is injected into the bloodstream. When a particular region of the brain is active, more of this agent passes by sensitive detectors than when the region is at rest. Comparison of resting and active images can potentially reveal functional localiza- tion at a resolution of about 1cm (see Stytz and Frieder 1990).
- Another technique for neural analysis is *functional magnetic resonance imaging*, or fMRI. This approach has emerged from more standard structured imaging based on nuclear magnetic resonance (NMR). Like PET, this approach compares resting with active neuronal states to reveal functional localization.
- A further contribution to the localization of brain function, with an important link to the imaging techniques just mentioned, is software algorithms developed by Barak Pearl- mutter and his colleagues (Pearlmutter and Parra 1997, Tang et al. 1999, 2000a, 2000b).

➢ AI CURRENT CHALLENGES & FUTURE DIRECTIONS

- The computational characterization of intelligence begins with the abstract specifica- tion of computational devices. Research through the 1930s, 40s, and 50s began this task, with Turing, Post, Markov, and Church all contributing formalisms that describe computa- tion.
- The goal of this research was not just to specify what it meant to compute, but rather to specify limits on what could be computed. The Universal Turing Machine (Turing 1950) is the most commonly studied specification, although Post's rewrite rules, the basis for production system computing (Post 1943), is also an important contribution. Church's model (1941), based on partially recursive functions, offers support for modern high-level functional languages,

such as Scheme, Ocaml, and Standard ML.

- Many consequential questions remain, however, within the epistemological founda- tions for intelligence in a physical system. We summarize, for a final time, several of these critical issues.

1. **The representation problem.** Newell and Simon hypothesized that the physical symbol system and search are necessary and sufficient characterizations of intelligence (see Section 16.1). Are the successes of the neural or sub-symbolic models and of the genetic and emergent approaches to intelligence refutations of the physical symbol hypothesis, or are they simply other instances of it?

   Even a weak interpretation of this hypothesis—that the physical symbol system is a *sufficient* model for intelligence—has produced many powerful and useful results in the modern field of cognitive science. What this argues is that we can implement physical symbol systems that will demonstrate intelligent behav- ior. Sufficiency allows creation and testing of symbol-based models for many aspects of human performance (Pylyshyn 1984, Posner 1989). But the strong interpretation—that the physical symbol system and search are *necessary* for intelligent activity—remains open to question (Searle 1980, Weizenbaum 1976, Winograd and Flores 1986, Dreyfus and Dreyfus 1985, Penrose 1989).

2. **The role of embodiment in cognition.** One of the main assumptions of the physical symbol system hypothesis is that the particular instantiation of a physical symbol system is irrelevant to its performance; all that matters is its formal structure. This has been challenged by a number of thinkers (Searle 1980, Johnson 1987, Agre and Chapman 1987, Brooks 1989, Varela et al. 1993) who essentially argue that the requirements of intelligent action in the world require a physical embodiment that allows the agent to be fully integrated into that world. The architecture of modern computers does not support this degree of situatedness, requiring that an artificial intelligence interact with its world through the extremely limited window of contemporary input/output devices. If this chal- lenge is correct, then, although some form of machine intelligence may be possi- ble, it will require a very different interface than that afforded by contemporary computers. (For further comments on this topic see Section 15.0, issues in natural language understanding, and Section 16.2.2, on epistemological constraints.)

3. **Culture and intelligence.** Traditionally, artificial intelligence has focused on the individual mind as the sole source of intelligence; we have acted as if an explanation of the way the brain encodes and manipulates knowledge would be a complete explanation of the origins of intelligence. However, we could also argue that knowledge is best regarded as a social, rather than as an individual construct. In a *meme-based* theory of intelligence (Edelman 1992), society itself carries essential components of intelligence. It is possible that an understanding of the social context of knowledge and human behavior is just as important to a theory of intelligence as an understanding of the dynamics of the individual mind/brain.

4. **Characterizing the nature of interpretation.** Most computational models in the representational tradition work with an already interpreted domain: that is, there is an implicit and *a priori* commitment of the system's designers to an interpre- tive context. Under this commitment there is little ability to shift contexts, goals, or representations as the problem solving evolves. Currently, there is little effort at illuminating the process by which humans construct interpretations.

   The Tarskian view of semantics as a mapping between symbols and objects in a domain of discourse is certainly too weak and doesn't explain, for example, the fact that one domain may have different interpretations in the light of differ- ent practical goals. Linguists have tried to remedy the limitations of Tarskian semantics by adding a theory of pragmatics (Austin 1962). Discourse analysis, with its fundamental dependence on symbol use in context, has dealt with these issues in recent years. The problem, however, is broader in that it deals with the failure of referential tools in general (Lave 1988, Grosz and Sidner 1990).

   The semiotic tradition started by C. S. Peirce (1958) and continued by Eco, Seboek, and others (Eco 1976, Grice 1975, Sebeok 1985) takes a more radical approach to language. It places symbolic expressions within the wider context of signs and sign interpretation. This suggests that the meaning of a symbol can

   of an interpretation and interaction with the environment (see Section 16.2.2).

5. **Representational indeterminacy.** Anderson's representational indeterminacy conjecture (Anderson 1978) suggests that it may in principle be impossible to determine what representational scheme best approximates the human problem solver in the context of a particular act of skilled performance. This conjecture is founded on the fact that every representational scheme is inextricably linked to a

larger computational architecture, as well as search strategies. In the detailed analysis of human skill, it may be impossible to control the process sufficiently so that we can determine the representation; or establish a representation to the point where a process might be uniquely determined. As with the uncertainty principle of physics, where phenomena can be altered by the very process of measuring them, this is an important concern for constructing models of intelli- gence but need not limit their utility.

But more importantly, the same criticisms can be leveled at the computa- tional model itself where the inductive biases of symbol and search in the context of the Church-Turing hypothesis still under constrain a system. The perceived need of some optimal representational scheme may well be the remnant of a rationalist's dream, while the scientist simply requires models sufficiently robust to constrain empirical questions. The proof of the quality of a model is in its abil- ity to offer an interpretation, to predict, and to be revised.

6. **The necessity of designing computational models that are falsifiable.** Popper (1959) and others have argued that scientific theories must be falsifiable. This means that there must exist circumstances under which the model is *not* a successful approximation of the phenomenon. The obvious reason for this is that *any* number of confirming experimental instances are not sufficient for confirmation of a model. Furthermore, much new research is done in direct response to the failure of existing theories.

   The general nature of the physical symbol system hypothesis as well as situated and emergent models of intelligence may make them impossible to falsify and therefore of limited use as models. The same criticism can be made of the conjectures of the phenomenological tradition Some AI data struc- tures, such as the semantic network, are so general that they can model almost anything describable, or as with the universal Turing machine, any computable function. Thus, when an AI researcher or cognitive scientist is asked under what conditions his or her model for intelligence will *not* work, the answer can be dif- ficult.

7. **The limitations of the scientific method.** A number of researchers (Winograd and Flores 1986, Weizenbaum 1976) claim that the most important aspects of intelligence are not and, in principle, cannot be modeled, and in particular not with any symbolic representation. These areas include learning, understanding natural

language, and the production of speech acts. These issues have deep roots in our philosophical tradition. Winograd and Flores's criticisms, for example, are based on issues raised in phenomenology (Husserl 1970, Heidegger 1962).

Most of the assumptions of modern AI can trace their roots back from Car- nap, Frege, and Leibniz through Hobbes, Locke, and Hume to Aristotle. This tra- dition argues that intelligent processes conform to universal laws and are, in principle, understandable.

- The most exciting aspect of work in artificial intelligence is that to be coherent and contribute to the endeavor we must address these issues. To understand problem solving, learning, and language we must comprehend the philosophical level of representations and knowledge. In a humbling way we are asked to resolve Aristotle's tension between *theoria* and *praxis*, to fashion a union of understanding and practice, of the theoretical and practi- cal, to live between science and art.
- AI practicioners are tool makers. Our representations, algorithms, and languages are tools for designing and building mechanisms that exhibit intelligent behavior. Through experiment we test both their computational adequacy for solving problems as well as our own understanding of intelligent phenomena.
