Q.1.(a). Explain with suitable examples, significance of each of the following in solving problems:

***i*) Contextual information.**

Ans: Contextual information plays a very important role not only in the visual understanding but also in the language and speech understanding. In case of speech understanding, consider the following example, in which the word 'with' has a number of meanings (or connotations) each being determined by the context.

- Mohan saw the boy in the park with a telescope.
- Mohan saw the boy in the park with a dog.
- Mohan saw the boy in the park with a statue.

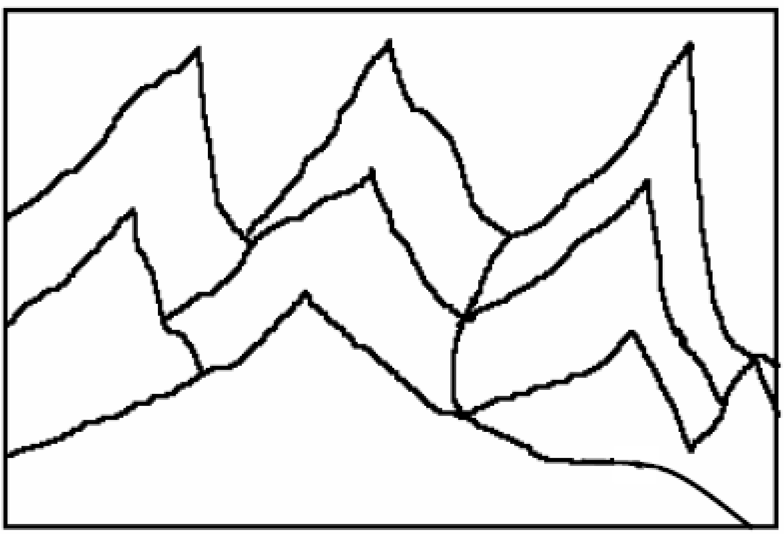
Further, the phrase 'for a long time' may stand for a few hours to millions of years, but again determined by the context, as explained below.

For a long time…..

- He waited in the doctor's room for a long time.
- It has not rained for a long time.
- Dinosaurs ruled the earth for a long time.

We consider another example that shows the significance of contextual information or knowledge and its simultaneous availability for visual understanding.

From the following picture, we can conclude that one of the curved lines represents a river and other curved lines represent sides of the hills only on the basis of the simultaneous availability of information of the pixels.



In addition to the advantage that human beings have in the matter of parallel processing as explained above, **Boden [12]** says: *humans have two psychological strengths which are yet to be approached by computer systems: a teeming richness of conceptual sources and the ability to evaluate new ideas in many different ways. The first of these is difficult enough for AI to emulate, the second is even more problematic*.

The definition is rather weak in the sense that it fails to include some areas of potentially large importance viz, problems that can be solved at present neither by human beings nor by computers. Also, it may be noted that, by and by, if computer systems become so powerful that there is no problem left, which human beings can solve better than computers, then nothing is left of AI according to this definition.

***ii*) Simultaneous availability of information**

Ans. : in visual understanding and speech understanding, which require simultaneous availability (availability in parallel) of large amount of information. In essence, it is found that computers are better than human beings in tasks requiring sequential but fast computations, where human beings are better than computers in tasks, requiring essentially parallel processing. In order to clarify what it is for a problem to essentially require parallel processing for its solution, we **consider the following problem:**

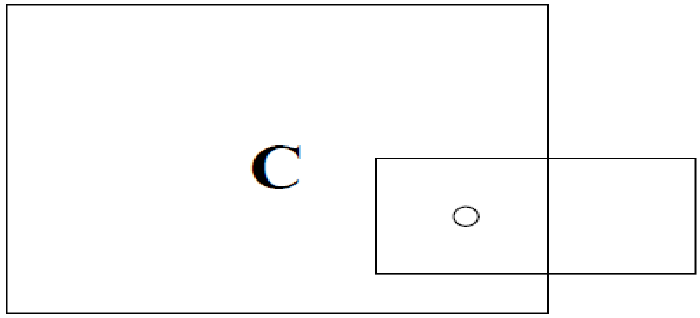


Figure 1.1

We are given a paper with some letter, say, C written on it and a card-board with a pin-hole in it. The card board is placed on the paper in such a manner that the letter is fully covered by the card board as shown in Figure 1.1. We are allowed to look at the paper only through the pin-hole in the card-board. The problem is to tell correctly the letter written on the paper by just looking through the pin-hole. As the information about the black and white pixels is not available simultaneously, it is not possible to figure out the letter written on the paper. The figuring out the letter on the paper requires, simultaneous availability of the whole of the grey-level information of all the points constituting the letter and its surrounding on the paper. The gray-level information of the surrounding of the letter provides the context in which to interpret the letter.

***b)*Discuss relations between 'organisation', 'information' and 'intelligence'.**

Ans. : Organization : *An organised system* of simpler objects, e.g., fundamental particles organise into nucleolus, nucleolus organise to f orm atomic nuclei, which alongwith electrons and protons organise into atoms and so on. Molecules, polymers, membranes, organs, living beings, societies, planets, planetary systems, galaxies … and finally the whole universe, each is known as an organised system of some simpler objects. An organisation builds upon pre-existing organisations. **Thus an organised system is recursively obtained (or defined) as an interdependent assembly of elements and/or organised systems.**

Intelligence: "defining intelligence usually takes a semester-long struggle, and even after that I am not sure we ever get a definition really nailed down". However, there are some characteristics of intelligence which are readily acceptable, some others acceptable after some thinking and still others that may be controversial. We enumerate the characteristics as considered by some A.I. writers and contributors and others. Enumeration of these characteristics here is essential because as A.I. technologists, we would study various techniques that help us in incorporating these characteristics, through computer programs, into machines, which we attempt to make intelligent according to Definition 4 of Artificial Intelligence. We give below the attributes verbatim from the respective sources.

Intelligence is a property of advanced information processing systems, which nt only engage in information processing, but are able to analyse their dynamically changing environment and to respond to it in such a way that:
i) survivability of the system is enhanced
ii) its reproducibility is enhanced (reproducibility is sort of self propagation through another system)
iii) if the system is goal-oriented, then achievability of goal is enhanced.

Information : *to maintain the integrity of the system as long as the system survives against the fourth* *fundamental property of the universe, i.e., evolution or change*. Gravitational pull, now an established entity, is just an information processing activity. **Thus 'information' is no more or no less an abstract concept than 'energy' or 'matter'.**

The relation between information and organisation and the characteristic difference between the two is exactly what **is the relation and characteristic difference between a number and a numeral.** A number is an *abstract* concept, wheras a numeral is its *physical manifestation or representation*. A number may have many representations and even may use many mediums for representations or manifestations. In the form **of, writing on the paper**, as patterns of ink dots on a piece of paper, the same number may be represented as 7 in decimal, 111 in binary, and even 4 + 2 + 1 again in decimal. In computer's memory, the same number is represented with the help of electronic components, a different medium, and not as shapes composed of ink-dots. In human brain the same number is represented, possibly, as some neural net.

Similarly, information is a concept and an organisation is its representation, i.e., physical manifestation. For the purpose of applying operations (like refining information, adding information etc) or for conveying information we use organisation (as patterns of ink dots on paper or as neural net in brain etc). Then, we manipulate the organisation or representation for applying operations on information (operations again are abstract, whereas manipulations are their physical realizations). Also, we communicate the organization for conveying information (communication is physical realisation of conveying). As in the case of number, information's representation may be through various organisations on various type of media such as patterns of ink dots on paper, neural nets in brain, or on flip-flops in electronic memory. For example, the information content of the organisation in the form of pattern of inkdots in the sentence 'Heat is a form of energy' is stored in the brain as an organization in the form of a Neural Network etc.

Q.2. *a)* For each of the following formulae, construct a truth-table, and then determine whether it is valid, consistent or inconsistent:

i) $(\sim C \rightarrow \sim D) \rightarrow (D \rightarrow C)$

ii) $((\sim C \vee D) \rightarrow B) \rightarrow (\sim C \rightarrow B)$

Ans:- *i)* $(\sim C \rightarrow \sim D) \rightarrow (D \rightarrow C)$

C	D	~ C	~ D	~ C → ~ D	D → C	(~ C → ~ D) → (D → C)
T	T	F	F	T	T	T
T	F	F	T	T	T	T
F	T	T	F	F	F	T
F	F	T	T	T	T	T

Definition: A formula is said to be valid if and only if it is true under all its interpretations. A formula is said to be invalid if and only if it is not true under at least one interpretation. A valid formula is also called a Tautology. A formula is invalid if there is at least one interpretation for which the formula has a truth value False.

Definition: A formula is said to be inconsistent (or unsatisfiable) if and only if it is False under all its interpretations. A formula is said to be consistent or satisfiable if and only if it is not inconsistent. In other words, a formula is consistent if there is at least one interpretation for which the formula has a truth value true.

By assuming both definitions the given formula (~ C → ~ D) → (D → C) is valid, consistence and tautology.

***Ans:-ii)* ((~ C ∨ D) → B) → (~ C → B)**

B	C	D	~ B	~ C	~ D	(~ C ∨ D)	((~ C ∨ D) → B)	(~ C → B)	((~ C ∨ D) → B) → (~ C → B)
T	T	T	F	F	F	T	T	T	T
T	T	F	F	F	T	F	T	T	T
T	F	T	F	T	F	T	T	T	T
T	F	F	F	T	T	T	T	T	T
F	T	T	T	F	F	T	F	T	T
F	T	F	T	F	T	F	T	T	T
F	F	T	T	T	F	T	F	F	T
F	F	F	T	T	T	T	F	F	T

Definition: A formula is said to be valid if and only if it is true under all its interpretations. A formula is said to be invalid if and only if it is not true under at least one interpretation. A valid formula is also called a Tautology. A formula is invalid if there is at least one interpretation for which the formula has a truth value False.

Definition: A formula is said to be inconsistent (or unsatisfiable) if and only if it is False under all its interpretations. A formula is said to be consistent or satisfiable if and only if it is not inconsistent. In other words, a formula is consistent if there is at least one interpretation for which the formula has a truth value true.

By assuming both definitions the given formula ((~ C ∨ D) → B) → (~ C → B) is valid, consistence and tautology.

***b)* Determine whether the following equivalence between the formulae on two sides of '=' holds or not (A → B) → C = (A → B) → (A → C) ?, by reducing each of the formulae on the two sides of '=' to one of the normal forms (DNF or CNF).**

Ans:-

L.H.S. (A → B) → C

(i) Removing inner „→‟ we get

= (~ A ∨ B) → C

(ii) Removing outer „→‟ we get

= ~ (~ A ∨ B) ∨ C

(iii) By using De Morgan‟s Law we get

= (~ (~ A) ∧ (~B)) ∨ C

= (A ∧ ~B) ∨ C which is in DNF. -------------- (1)

R.H.S. (A → B) → (A → C)

(i) Removing inner „→‟ we get

= (~ A ∨ B) → (~ A ∨ C)

(ii) Removing outer „→‟ we get

= ~ (~ A ∨ B) ∨ (~ A ∨ C)

(iii) By using De Morgan‟s Law we get

= (~ (~ A) ∧ (~B)) ∨ (~A ∨ C)

= (A ∧ ~B) ∨ (~A ∨ C)

(iv) By using equivalences formula (ii) we get

= (A ∧ ~B) ∨ C which is also in DNF. -------------- (2)

Now we have:

L.H.S. = (A ∧ ~B) ∨ C -------------- (1)

R.H.S. = (A ∧ ~B) ∨ C -------------- (2)

By using equation (1) and (2) we get:

L.H.S. = R.H.S.

(A ∧ ~B) ∨ C = (A ∧ ~B) ∨ C

Hence the given formula is equivalence between two sides of „=‟ holds in DNF.

Q.3. Translate first statements, given in the following argument into Propositional Logic, and then show that the conclusion logically follows from the premisses (given statements):

Premisses: Either taxes are increased, or if expenditures rise, then the debt ceiling is raised. If taxes are increased, then the cost of collecting taxes increases. If a rise in expenditures implies that the government borrows more money, then if the debt ceiling is raised, then interest rates increase. If taxes are not increased and the cost of collecting taxes does not increase, then if the debt ceiling is raised, then the government borrows more money. The cost of collecting taxes does not increase. Either interest rates do not increase or the government does not borrow more money.

Conclusion: Either the debt ceiling is not raised or expenditures do not rise. *You may use the symbols:* (T: Taxes are increased. E: Expenditures rise. D: The debt ceiling is raised. C: The cost of collecting taxes increases. G: The government borrows more money. I: Interest rates increase).

Ans:-

We have:

T: Taxes are increased.

E: Expenditures rise.

D: The debt ceiling is raised.

C: The cost of collecting taxes increases.

G: The government borrows more money.

I: Interest rates increase

(i) $(T \vee E) \rightarrow D$

(ii) $(T \rightarrow C)$

(iii) $((E \rightarrow G) \rightarrow D) \rightarrow I$

(iv) $((\sim T \wedge \sim C) \rightarrow D) \rightarrow G$

(v) $\sim C$

(vi) $(\sim I \vee \sim G)$

$$((((T \vee E) \rightarrow D) \vee (T \rightarrow C) \vee (((E \rightarrow G) \rightarrow D) \rightarrow I) \vee (((\sim T \wedge \sim C) \rightarrow D) \rightarrow G)) \vee (\sim C) \vee (\sim I \vee \sim G))$$

Q.4. Transform the following formula first in Prenex Normal Form and then into Skolem Standard Form:

$$(\forall x) ([(\forall y) (\exists z) (\sim P (x, y) \wedge Q (x, z))] \rightarrow (\exists u) R (x, y, u))$$

Ans:- Prenex Normal Form:

Answer: <u>Prenex Normal Form:</u>

<u>Step1</u>

Removing inner $\rightarrow$ sign, we get:

$(\forall x) ([(\forall y) (\exists z) (\sim P(x, y) \wedge Q(x , z))] \vee (\sim (\exists u) R(x , y, u))$

<u>Step2</u>

Taking inner most ~ sign we get the formula as:

$(\forall x) ([(\forall y) (\exists z) (\sim P(x, y) \wedge Q(x, z))] \vee ((\forall u) \sim R(x, y, u))$

<u>Step3</u>

Renaming u as z in $((\forall u) \sim R(x, y, u))$, we get

$(\forall x) ([(\forall y) (\exists z) (\sim P(x, y) \wedge Q(x, z))]) \vee ((\forall z) \sim R(x , y, z))$

<u>Step4</u>

Taking outer ~ sign and simplifying, we get the formula as:

$(\forall x) ([(\forall y) (\forall z) (\sim (\sim P(x, y) \wedge Q(x, z)))] \vee ((\forall z) \sim R(x, y, z))$ Or,

$(\forall x) ([(\forall y) (\forall z) (P(x, y) \vee \sim Q(x, z))] \vee ((\forall z) \sim R(x, y, z))$

<u>Step5</u> As P (x, y) does not involve z, we get the formula as:

$(\forall x) (\forall z) ((\forall y) P(x, y) \vee \sim Q(x, z)) \vee ((\forall z) \sim R (x, y, z))$

And hence the answer

Skolem Standard Form:

First of all we reduce the matrix to CNF:

$= (\sim P(x, y) \wedge Q(x, z) \rightarrow (\exists u) R(x, y, u))$

By removing „→" we get

$= (\sim (\sim P(x, y) \wedge Q(x, z) \vee (\exists u) R(x, y, u))$

By Using De Morgan‟s law

$= (\sim(\sim P(x, y) \vee \sim Q(x, z) \vee (\exists u) R(x, y, u))$

$= (P(x, y) \vee \sim Q(x, z) \vee (\exists u) R(x, y, u))$ Now we have

$= ((\exists u)(P(x, y) \vee R(x, y, u) \vee \sim Q(x, z) \vee R(x, y, u))$

Now the standard for is:

$(\forall x)(\forall y)(\exists z)(\exists u)(P(x, y) \vee R(x, y, u) \vee \sim Q(x, z) \vee R(x, y, u)$

Q.5. Translate the following three statements in First Order Predicate Logic, and then deduce *(iii)* from *(i)* and *(ii)*:

(You should not use resolution method)

i) Lord Krishna is loved by everyone who loves someone.

ii) No one loves nobody

iii) Lord Krishna is loved by everyone.

Ans:- : Let us use the symbols

Love (x, y): *x loves y (or y is loved by x)*

LK: *Lord Krishna*

Then the given problem is formalized as :

(i) $(\forall x)((\exists y) \text{Love}(x, y) \rightarrow \text{Love}(x, LK))$

(ii) $\sim (\exists x)((\forall y) \sim \text{Love}(x, y))$

To show: $(\forall x)(\text{Love}(x, LK))$

As resolution is a refutation method, assume negation of the last statement as an axiom.

(iii) $\sim (\forall x) \text{Love}(x, LK)$

The formula in (i) above is reduced in standard form as follows:

$(\forall x)(\sim (\exists y) \text{Love}(x, y) \vee \text{Love}(x, LK))$

$= (\forall x)((\forall y) \sim \text{Love}(x, y) \vee \text{Love}(x, LK))$

$= (\forall x)(\forall y)(\sim \text{Love}(x, y) \vee \text{Love L}(x, LK))$

(∵ (∀ y) does not occurs in Love (x, LK))

After dropping universal quantifications, we get

(iv) $\sim \text{Love}(x, y) \vee \text{Love}(x, LK)$

Formula (ii) can be reduced to standard form as follows:

(ii) = (∀x) (∃y) Love (x, y)

y is replaced through skolomization by f(x) so that we get

(∀x) Love (x, f(x))

Dropping the universal quantification

(v) Love (x, f(x))

The formula in (iii) can be brought in standard form as follows:

(iii) = (∃x) (~ Love (x, LK))

As existential quantifier *x* is not preceded by any universal quantification, therefore, *x* may be substituted by a constant *a* , i.e., we use the substitution {a/x} in (iii) to get the standard form:

(vi) ~ Love (a, LK).

Thus, to solve the problem, we have the following standard form formulas for resolution:

(iv) ~ Love (x, y) ∨ Love (x, LK)

(v) Love (x, f(x))

(vi) ~ Love (a, LK).

*Two possibilities of resolution exist for two pairs of formulas viz.*one possibility: resolving (v) and (vi).second possibility: resolving (iv) and (vi).

The possibilities exist because for each possibility pair, the predicate *Love* occurs in complemented form in the respective pair.

Q.6. **a)**

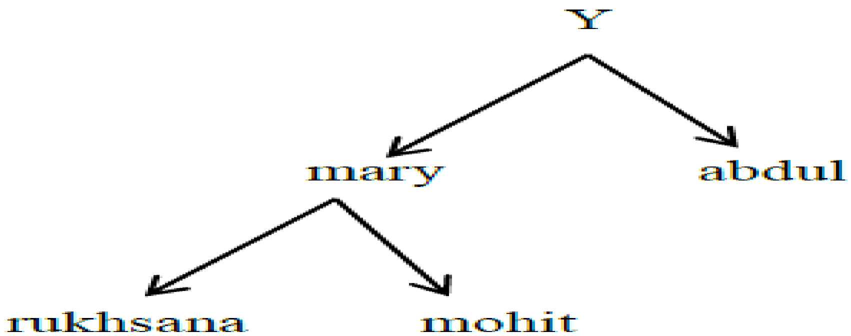
Write a recursive function in LISP named **power** that takes two numeric arguments, n and m, that computes n^{th} power of m (i.e., m^n).

Ans:-

```
(defun power(n m)
        (do*    (
                    (answer m        (* m answer))
                    (power n          (- power 1))
                    (counter          (-power 1)
                                      (-power1)
                    )
                    )
                    ((zero counter) answer)
                )
        )
```

b) Write a PROLOG programme that answers questions about family members and relationships. Include predicates and rules which define sister, brother, father, mother, grandfather, grand-child and uncle. The programme should be able to answer queries such as the following:
? – father (X, mohit)
? – grandson (X, Y)
? – uncle (abdul, ruksana)
? – mother (mary, X)

Family Tree of the given situation



<u>Knowledge Base:</u>

From the above illustration of the given family tree, we came into notice about the following facts:

1. Y is parent of X and abdul.
2. X is parent of rukhsana and mohit

Therefore, these may be stated in predicate logic as below:
Parent(X,Y), Parent(X, abdul), Parent(mohit, X), parent(rukhsana, X)

<u>Rule for the above realtion:</u>

is_mother-of(mery, mohit):- Parent(X, Y) ^ Parent(abdul, Y) ^ Parent(mohit, X) ^ Parent(rukhsana, X) .

<u>Program</u>

/* Family Tree*/ Parent(X, Y). Parent(abdul, Y). Parent(mohit, X).

Parent(rukhsana, X).

Father(mohit).

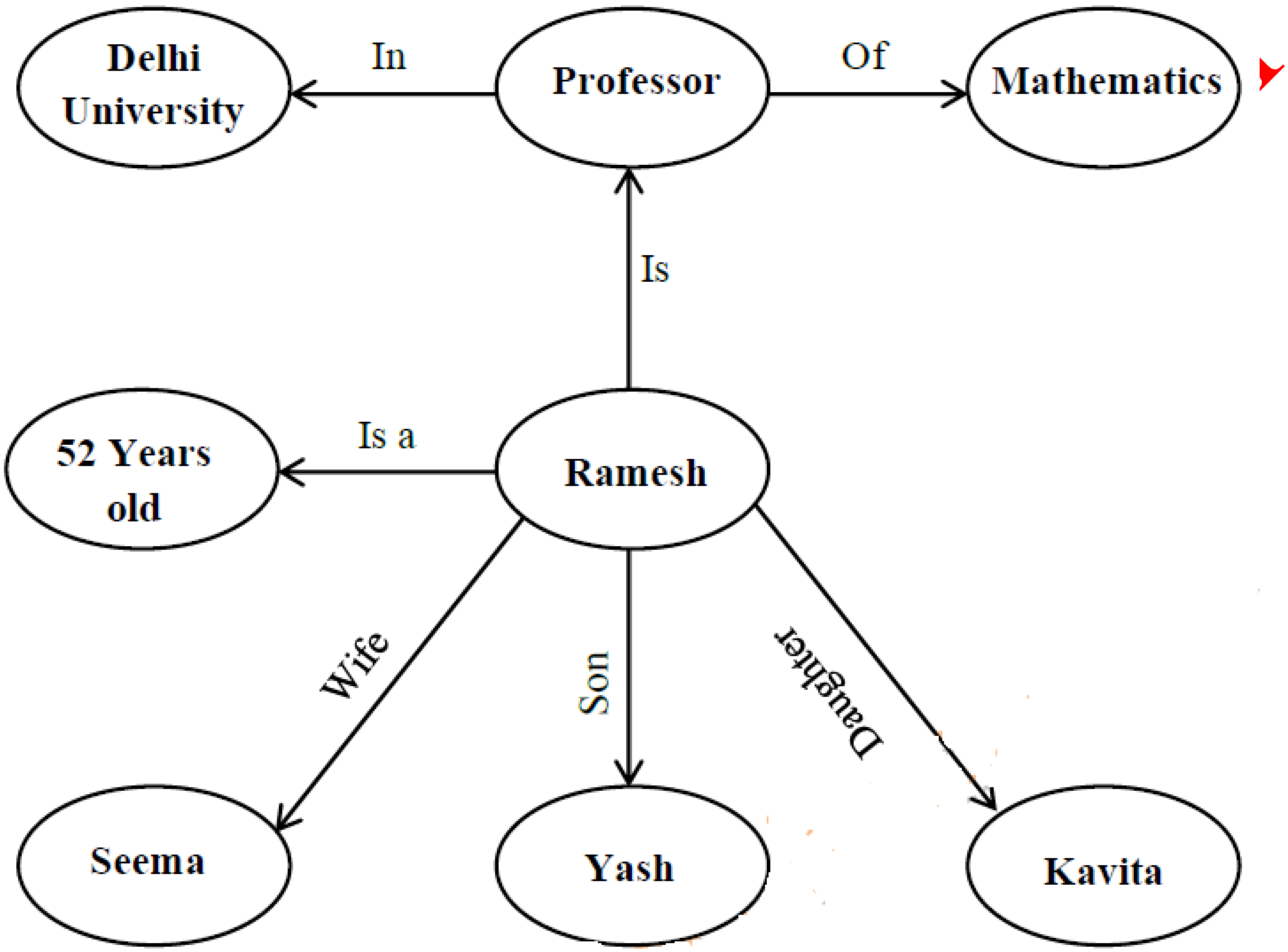
Grandson(Y).

Uncle(rukhsana).

Mother(X).

Q.7. a) Give Semantic Net representation of the facts given below : *"Ramesh is a 52 year old Professor of Mathematics in Delhi University. The name of his wife, son and daughter are respectively Seema, Yash and Kavita".*

Ans:-



b) Create a frame network for terrestrial motor vehicles (cars, trucks, motorcycles) and give one complete frame in detail for cars which includes the slots for the main component parts, their attributes, and relations between parts. Include an as-needed slot for the gas of each type mileage.

Ans:-

Knowledge about frame of vehicle:

Vehicle:

Subclass	:	Motor Vehicle
gears	:	yes
breaks	:	yes
emergency breaks	:	yes
engine	:	yes
transportation	:	yes
cargo	:	yes
Subclass	:	Locomotive Vehicle
gears	:	yes
breaks	:	yes
emergency breaks	:	yes
engine	:	yes (high capacity)
transportation	:	yes (high capacity)
cargo	:	yes (high capacity)

Knowledge about frame of motor vehicle(as per the Question):

Vehicle:

Subclass	:	Motor Vehicle
gears	:	yes
breaks	:	yes
engine	:	yes
parts	:	yes
average	:	yes

Cars:

instance	:	vehicle
consumable parts	:	yes (1…n)
durable parts	:	yes (1…n)
engine	:	yes 800CC
seating capacity	:	5 persons
fuel average	:	22 KMPL

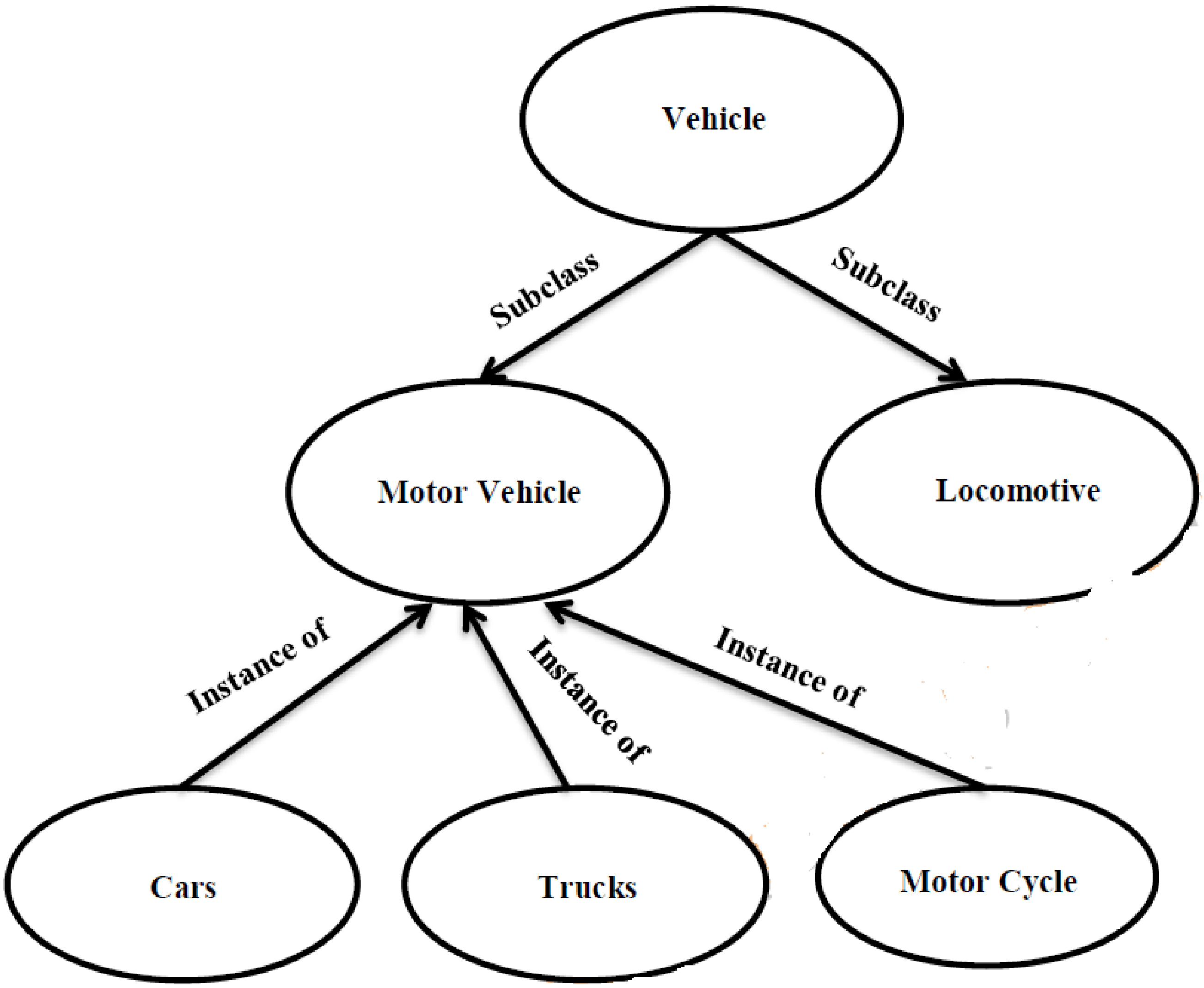
Trucks:

instance	:	vehicle
consumable parts	:	yes (1…n)
durable parts	:	yes (1…n)
engine	:	yes 4000CC
seating capacity	:	3 persons
fuel average	:	40 KMPL

Motorcycle:

instance	:	vehicle
consumable parts	:	yes (1…n)
durable parts	:	yes (1…n)
engine	:	yes 125CC
seating capacity	:	2 persons
fuel average	:	70 KMPL

Diagram for Terrestrial Motor vehicle



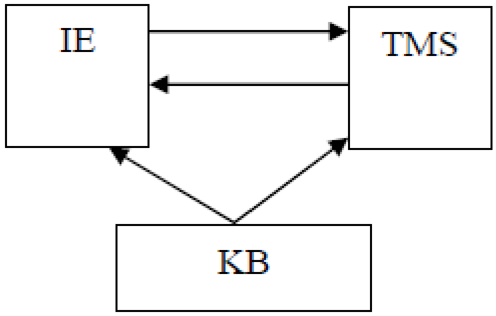
Question 8:
(a) For the following fuzzy sets:
X = {*x*/7, *y*/3, *z*/0, *u*/1, *v*/4} and
Y = {*x*/3, *y*/8, *z*/6, *u*/9, *v*/0}
Find (i) X ∪ Y (ii) X ∩ Y (iii) (X′ ∩ Y)′
Answer:

(i) X ∪ Y = { x/7, y/8, z/6, u/9, v/3 }
(ii) X ∩ Y = { x/3, y/3, z/0, u/1, v/0 }
(iii) (X′ ∩ Y)′ = { x/2, y/1, z/0, u/1, v/1}

Q.8**. b) Write a note on Non-monotonic reasoning systems.?**
Ans. : A non-monotomic reasoning system is one which allows retracting of old knowledge due to discovery of new facts which contradict or invalidate a part of the current knowledge base. Such systems also take care that retracting of a fact may necessitate a chain of retractions from the knowledge base or even reintroduction of earlier retracted ones from K.B. Thus, chain-shrink and chain emphasis of a K.B and reintroduction of earlier retracted ones are part of a non-monotomic reasoning system.

To meet the requirement for reasoning in the real-world, we need non-monotomic reasoning systems also, in addition to the monotomic ones. This is true specially, in view of the fact that it is not reasonable to expect that **all the knowledge needed** for a set of tasks could be acquired, validated, and loaded into the system just at the outset. In general, initial knowledge is an *incomplete* set of *partially true* facts. The set may also be redundant and may contain inconsistencies and other sources of uncertainty.

Major components of a Non-Monotomic reasoning system
Next, we discuss a typical non-monotomic reasoning system (NMRS) consists of the following three major components:
(1) Knowledge base (KB)- The KB contains information, facts, rules, procedures etc. relevant to the type of problems that are expected to be solved by the system.
(2) Inference Engine (IE)- The component IE of NMRS gets facts from KB to draw new inferences and sends the new facts discovered by it (i.e., IE) to KB.
(3) Truth-Maintenance System (TMS)- The component TMS, after addition of new facts to KB. either from the environment or through the user or through IE, checks for validity of the KB. It may happen that the new fact from the environment or inferred by the IE may conflict/contradict some of the facts already in the KB.



Next, We explain the ideas discussed above through an example: Let us assume KB has two facts P and ~ Q → ~ P and a rule called Modus Tollens. When IE is supplied these knowledge items, it concludes Q and sends Q to KB. However, through interaction with the environment, KB is later supplied with the information that ~ P is more appropriate than P. Then TMS, on the addition of ~ P to KB, finds that KB is no more consistent, at least, with P. The knowledge that ~ P is more appropriate, suggests that P be retracted. Further Q was concluded assuming P as True. But, in the new situation in which P is assumed to be not appropriate, Q also becomes inappropriate. P and Q are not deleted from KB, but are just marked as dormant or ineffective. This is done in view of the fact that later on, if again, it is found appropriate to include P or Q or both, then, instead of requiring some mechanism for adding P and Q, we just remove marks that made these dormant. Non-monotomic Reasoning Systems deal with Non-monotomic Reasoning Systems deal with

1) Revisable belief systems

2) incomplete K.B. —— Default Reasoning
   —— Closed World assumption

AI programs that achieve expert-level competence in solving problems in different

Q.8. c) Discuss briefly various methods/ mechanism for handling incompleteness of a knowledge-base (KB).

Ans. : One method of handling uncertainty due to incomplete KB is through default reasoning which is also a form of non-monotomic reasoning and is based on the following mechanism: Whenever, for any entity relevant to the application, information is not in the KB, then a default value for that type of entity, is assumed and is assigned to the entity. The default assignment is not arbitrary but is based on experiments, observations or some other rational grounds. However, the typical value for the entity is removed if some information contradictory to the assumed or default value becomes available.

Another mechanism of handling incompleteness of a KB is called 'Closed World Assumption' (CWA). This mechanism is useful in applications where most of the facts are known and therefore it is reasonable to assume that if a proposition cannot be proved, then it is FALSE. This is called CWA with failure as negation.

It contains facts and heuristic knowledge. Developers try to use a uniform representation of knowledge as for as possible. There are many knowledge representation schemes for expressing knowledge about the application domain and some advance expert system shells use both frames (objects) and IF-THEN rules. In PROLOG the knowledge is represented as logical statements.

A knowledge-based system is any system which performs a job or task by applying rules of thumb to a symbolic representation of knowledge, instead of employing mostly algorithmic or statistical methods. Often the term *expert systems* is reserved for programs whose knowledge base contains the knowledge used by human experts, in contrast to knowledge gathered from textbooks or non-experts.

But more often than not, the two terms, expert systems and knowledge-based systems are taken us synonyms. Together they represent the most widespread type of *AI* application. The area of human intellectual endeavour to be captured in an expert system is sometimes called the task domain. **Task** refers to some goal-oriented, problem-solving activity. **Domain** refers to the area within which the task is being performed. Some of the typical tasks are diagnosis, planning, scheduling, configuration and design.

For example, a program capable of conversing about the weather would be a knowledge-based system, even if that program did not have any expertise in meteorology, but an expert system must be able to perform weather forecasting.

Building a expert system is known as **knowledge engineering** and its practitioners are called **knowledge engineers**. It is the job of the knowledge engineer to ensure to make sure that the computer has all the knowledge needed to solve a problem. **The knowledge engineer must choose** one or more forms in which to represent the required knowledge i.e., s/he must choose **one or more knowledge representation schemes** (A number of knowledge representing schemes like semantic nets, frames, predicate logic and rule based systems have been discussed above. We would be sticking to rule based representation scheme for our discussion on expert systems). S/he must also ensure that the computer can use the knowledge efficiently by selecting from a handful of reasoning methods.

Q.9 Translate the following argument into FOPL and then using Deductive Rules of Inference (given in Unit 2 of Block 2) prove/ refute the following argument. No man who is a candidate will be defeated if he is a good campaigner. Any man who runs for office is a candidate. Any candidate who is not defeated will be elected. Every man who is elected is a good campaigner. Therefore, any man who runs for office will be elected if and only if he is a good campaigner. You may use the notation (Mx, Cx, Dx, Gx, Rx, Ex)

Ans:- Let us use the notations as:

M(x) = x is a man. C(x) = x is a candidate.

D(x) = x is a defeated candidate. G(x) = x is a good campaigner. R(x) = x runs for office.

E(x) = x is elected candidate.

Translations of the statements into FOPL

No man who is candidate will be defeated if he is a good campaigner:

1. $(\exists x)\ (C(x)\ G(x) \rightarrow D(x))$

Any man who runs for office is a candidate:

2. $(\forall x)\ (R(x) \rightarrow C(x))$

Every man who is elected is a good campaigner:

3. $(\forall x)\ (E(x) \rightarrow G(x))$

Therefore, any man who runs for office will be elected of and only if he is a good campaigner:

4. $(\forall x)\ (R(x)\ \exists G(x) \rightarrow E(x)\ )$

From expressions 2 and 3 **Hypothetical Syllogism (H.P.)** we get

$R(x) \rightarrow G(x)$ -------------- 5

Therefore, from the above simplification we can assume that **any many runs for the office is also a good campaigner**, and **hence the statement is accepted**.

Q.10. ***a)*** **Describe briefly each of the components of an expert system shell.**

Ans. : An expert system tool, or shell, is a software development environment containing the basic components of expert systems. Associated with a shell is a prescribed method for building applications by configuring and instantiating these components. Expert system shells are basically used for the purpose of allowing non-programmers to take advantage of the already developed templates or shells and which have evolved because of the efforts of some pioneers in programming who have solved similar problems before. The core components of an expert systems are the knowledge base and the reasoning engine. A generic expert system shell is shown below :

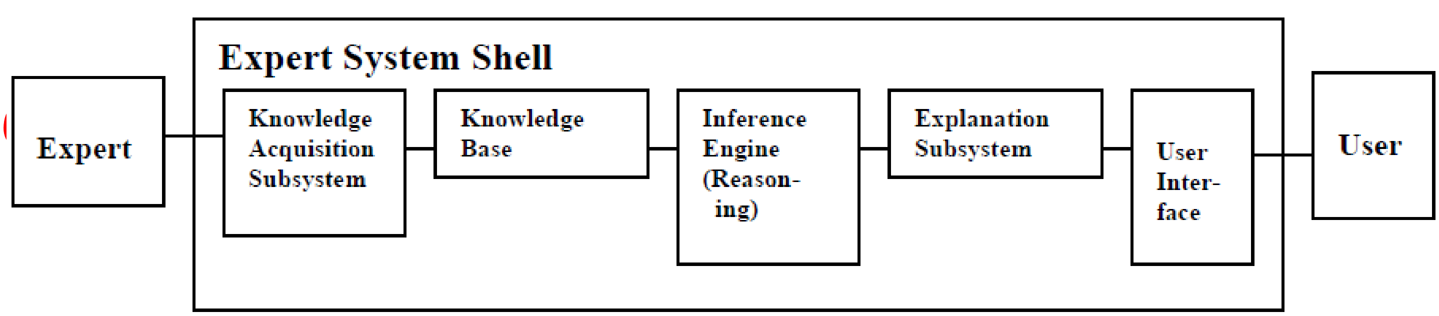


Figure 1.2: Components of Expert System Tool (Shell)

As we can see in the figure above, **the shell includes** the *inference engine, a knowledge acquisition subsystem, an explanation subsystem and a user interface*. When faced with a new problem in any given domain, we can find a shell which can provide the right support for that problem, so all we need is the knowledge of an expert. There are many commercial expert system shells available now, each one adequate for a different range of problems. Taking help of expert system shells to develop expert systems greatly reduces the cost and the time of development as compared to developing the expert system from the scratch. Let us now discuss the components of a generic expert system shell. We will discuss about:

• **Knowledge Base**
• **Knowledge Acquisition Subsystem**
• **Inference Engine**
• **Explanation Subsystem**
• **User Interface**

- ***Knowledge Base:*** *It contains facts and heuristic knowledge. Developers try to use a uniform representation of knowledge as for as possible. There are many knowledge representation schemes for expressing knowledge about the application domain and some advance expert system shells use both frames (objects) and IF-THEN rules. In PROLOG the knowledge is represented as logical statements.*
- **Knowledge Acquisition Subsystem :** As an expert may not be a computer literate, so capturing information includes interviewing, preparing questionnaires etc. which is a very slow and time consuming process. So **collecting knowledge needed to solve problems** and build the knowledge base has always been **the biggest bottleneck** in developing expert systems.
- **Inference Engine** An inference engine is used to perform reasoning with both the expert knowledge which is extracted from an expert and most commonly a human expert) and data which is specific to the problem being solved. Expert knowledge is mostly in the form of a set of IF-THEN rules. The case specific data includes the data provided by the user and also partial conclusions (along with their certainty factors) based on this data. In a normal forward chaining rule-based system, the case specific data is the elements in the working memory.
- ***Explanation Subsystem (Example MYCIN)*** *An explanation subsystem allows the program to explain its reasoning to the user. The explanation can range from how the final or intermediate solutions were arrived at to justifying the need for additional data.*
- ***Explanation Subsystem in MYCIN (An overview)*** *MYCIN is one of the first popular expert systems made for the purpose of medical diagnosis. Let us have a look at how the explanation subsystem in MYCIN works : To explain the reasoning for deciding on a particular medic al parameter's or symptom's value, it retrieves a set of rules and their conclusions. It allows the user to ask questions or queries during a consultation.*
- ***User interface*** *It is used to communicate with the user. The user interface is generally not a part of the xpert system technology, and was not given much attention in the past. However, it is now widely accepted that the user interface can make a critical difference in the utility of a system regardless of the system's performance.*
- **EMYCIN (An expert system shell)** EMYCIN provides a domain-independent framework or template for constructing and running any consultation programs. EMYCIN stands for "Empty MYCIN" or "Essential MYCIN" because it basically constitutes a MYCIN system minus its domain-specific medical knowledge. However, EMYCIN is something more than this, as it offers a number of software tools for helping expert system designers in building and debugging performance programs

b) **What is an agent? Discuss briefly different (*at least four*) types of agents.**

Ans.:Agent- An **agent** may be thought of as an entity that acts, generally on behalf of someone else. More precisely, an **agent** is an entity that ***perceives*** its environment through ***sensors*** and ***acts*** on the environment through ***actuators***. Some experts in the field require an agent to be additionally autonomous and goal directed also.

A **percept** may be thought of as an input to the agent through its censors, over a unit of time, sufficient enough to make some sense from the input. **Percept sequence** is a sequence of percepts, generally long enough to allow the agent to initiate some action.

In order to further have an idea about what a *computer agent is*, let us consider one of the first definitions of agent, which was coined by John McCarthy and his friends at MIT.

> ***A software agent is a system which, when given a goal to be achieved, could carry out the details of the appropriate (computer) operations and further, in case it gets stuck, it can ask for advice and can receive it from humans, may even evaluate the appropriateness of the advice and then act suitably.*** :

some of the general categories of agents based on their agents programs:

- ***SR (Simple Reflex) agents***
- ***Model Based reflex agents***
- ***Goal-based agents*** **Intelligence**
- ***Utility based agents***

Simple Reflex (SR) Agents

These are the agents or machines that have no internal state (i.e., the don't remember anything) and simply react to the current percepts in their environments. An interesting set of agents can be built, the behaviour of the agents in which can be captured in the form of a simple set of functions of their sensory inputs. One of the earliest implemented agent of this category was called ***Machina Speculatrix.*** This was a device with wheels, motor, photo cells and vacuum tubes and was designed to move in the direction of light of less intensity and was designed to avoid the direction of the bright light. **A boundary following robot is also an SR agent**.

For example : ***IF*** *a human being comes in front of the automobile suddenly* **THEN** *apply breaks immediately*.

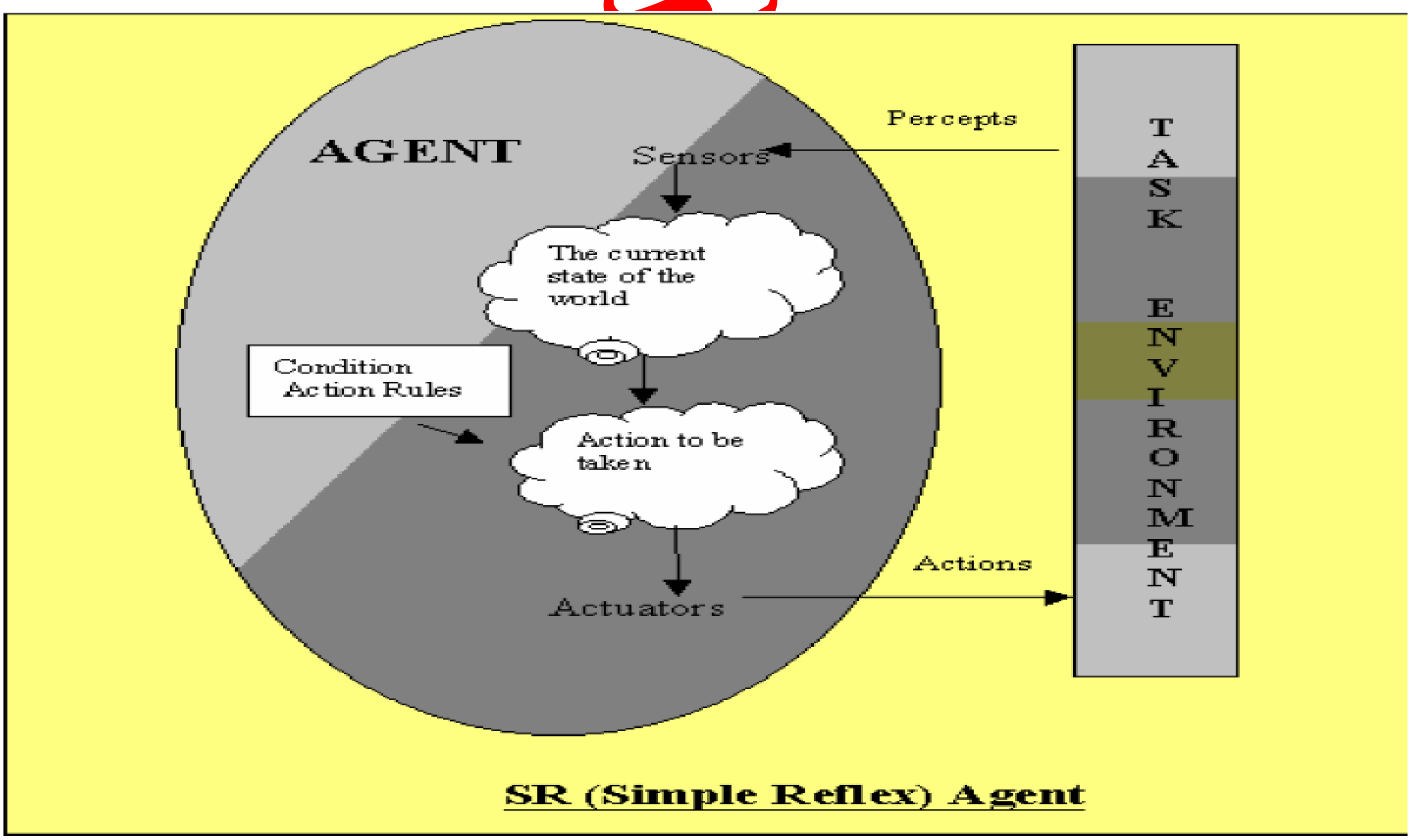


Figure : SR (Simple Reflex) Agent

Model Based Reflex agents

Simple Reflex agents are not capable of handling task environments that are not fully observable. In order to handle such environments properly, in addition to reflex capabilities, the agent should, maintain some sort of internal state in the form of a function of the sequence of percepts recovered up to the time of action by the agent. Using the percept sequence, the internal state is determined in such a manner that it reflects some of the aspects of the unobservable environment. Further, in order to reflect properly the unobserved environment, the agent is expected to have a model of the task environment encoded in the agent's program, where the model has the knowledge about–

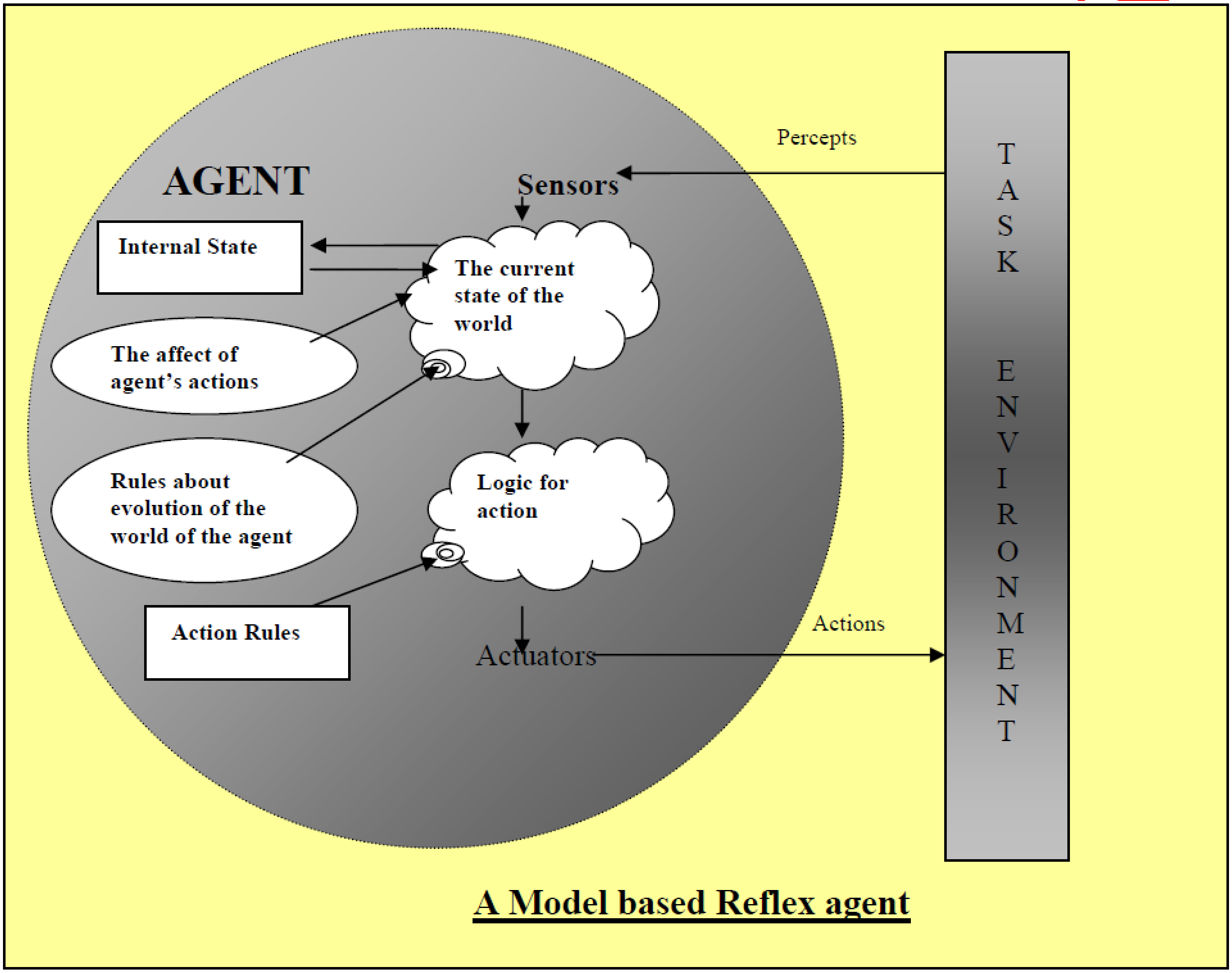


Figure 2.4: A Model based Relfex agent

Goal Based Agents

In order to design appropriate agent for a particular type of task, we know the nature of the task environment plays an important role. Also, it is desirable that the complexity of the agent should be minimum and just sufficient to handle the task in a particular environment. In this regard, first we discussed the simplest type of agents, viz., Simple Reflex Agents.

Goal based agents are **driven by the goal** they want to achieve, i.e., **their actions are based on the information regarding their goal, in addition to, of course, other information in the current state.**

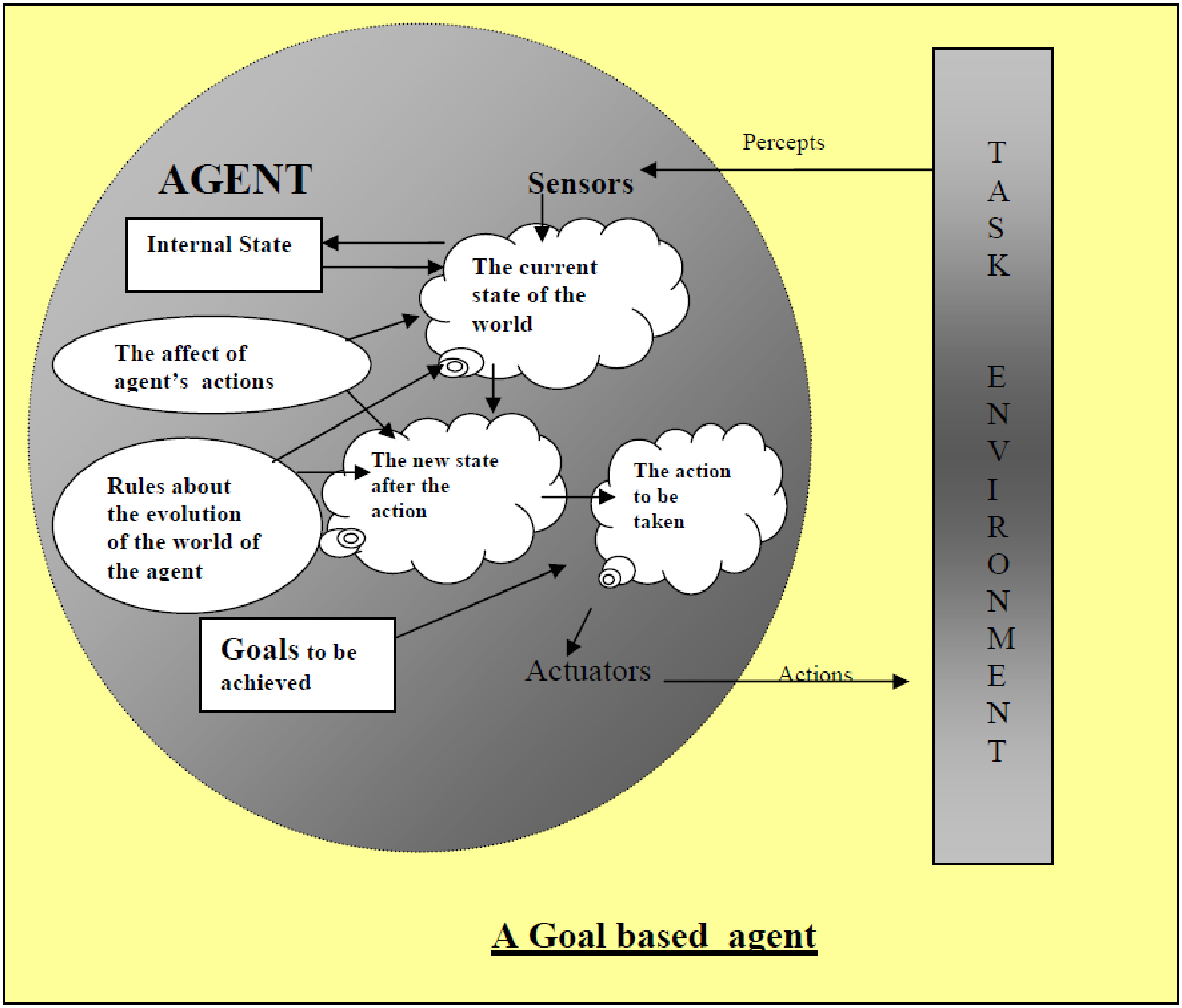


Figure 2.5: A Goal based agent

Utility Based Agents

Goal based agent's success or failure is judged in terms of its capability for achieving or not achieving its goal. A goal-based agent, for a given pair of environment state and possible input, only knows whether the pair will lead to the goal state or not. Such an agent will not be able to decide in which direction to proceed when there are more than one conflicting goals. Also, in a goal-based agent, there is no concept of partial success or somewhat satisfactory success. Further, if there are more than one methods of achieving a goal, then no mechanism is incorporated in a Goal-based agent of choosing or finding the method which is faster and more efficient one, out of the available ones, to reach its goal.

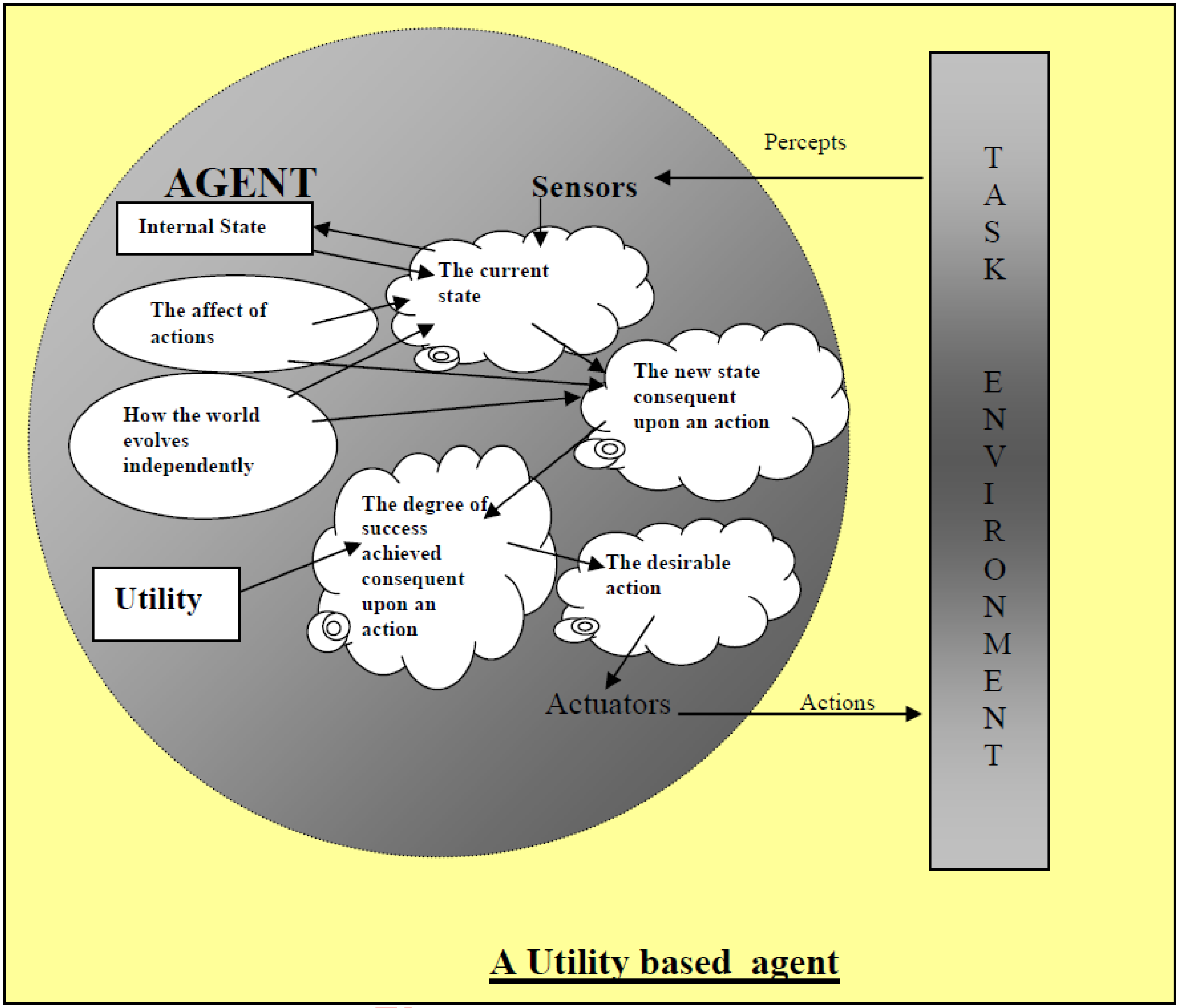


Figure 2.6: A Utility based agent