**QUESTION PAPER CODE: 50397**

**B.E/B.TECH. DEGREE EXAMINATION, NOVEMBER/DECENVER 2017**

**FIFTH/SIXTH SEMESTER**

**COMPUTER SCIENCE AND ENGINEERING**

**CS 6659- ARTIFICIAL INTELLIGENCE**

**(REGULATIONS 2013)**

PART A

1. **State the advantages of Breadth First Search.**

Page No.: 25 question No.3 Unit-I

1. **What is Commutative production system?**

A commutative production system: A commutative production system is a production system that is both monotonic and partially commutative.

1. **Convert the following into Horn clauses.**

**∀x: ∀y: cat9x) v fish(y)-> likes –to –eat (x,y)**

All cats like to eat all types of fish.

1. **Differentiate forward and backward reasoning**

Page no,:116 question no.: 20 and 21 Unit-2

1. **Define fuzzy reasoning.**

In real world, there exists much fuzzy knowledge; Knowledge that is vague, imprecise, uncertain, ambiguous, inexact, or probabilistic in nature. Human thinking and reasoning frequently involve fuzzy information, originating from inherently inexact human concepts. Humans can give satisfactory answers, which are probably true. However, our systems are unable to answer many questions. The reason is, most systems are designed based upon classical set theory and two-valued logic which is unable to cope with unreliable and incomplete information and give expert opinions. We want, our systems should also be able to cope with unreliable and incomplete information and give expert opinions. Fuzzy sets have been able provide solutions to many real world problems. Fuzzy Set theory is an extension of classical set theory where elements have degrees of membership.

1. **Compare production based system with frame based system**

**A production based system** (or production rule system) is a computer program typically used to provide some form of [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence), which consists primarily of a set of rules about behavior. These rules, termed productions, are a basic [representation](https://en.wikipedia.org/wiki/Knowledge_representation) found useful in [automated planning](https://en.wikipedia.org/wiki/Automated_planning_and_scheduling), [expert systems](https://en.wikipedia.org/wiki/Expert_systems) and [action selection](https://en.wikipedia.org/wiki/Action_selection). A production system provides the mechanism necessary to execute productions in order to achieve some goal for the system.Productions consist of two parts: a sensory precondition (or "IF" statement) and an action (or "THEN").

**Frame-based systems** are knowledge representation systems that use frames, a notion originally introduced by Marvin Minsky, as their primary means to represent domain knowledge. A frame is a structure for representing a CONCEPT or situation such as "living room" or "being in a living room."

1. **Define adaptive learning.**

Refer unit-4 q.no 9 first paragraph pg.no 170

1. **What is hierarchical planning?**

Refer unit-4 q.no 2 pg no 155

1. **List the characteristic features of expert system.**

Refer unit-5 part-A q.no-3 pg.no 180

1. **What is MOLE?**

**MOLE**is a knowledge-acquisition tool for generating expert systems that do heuristic classification. The problem-solving method proposed by MOLE makes several heuristic assumptions about the space of covering hypotheses that MOLE is able to exploit when acquiring knowledge.

**PART B**

**11 A) Explain the following types of Hill climbing search techniques**

1. **Simple Hill Climbing**
2. **Steepest-Ascent Hill Climbing**
3. **Simulated Annealing.**

Page No. 33, Question No.:4, Unit – I

**Or**

**B) Discuss Constraint Satisfaction problem with an algorithm for solving crypt arithmetic problem**

Constraint Satisfaction Problem1.

CONSTRAINT SATISFACTION:-

Many problems in AI can be considered as problems of constraint satisfaction, in which the goal state satisfies a given set of constraint. constraint satisfaction problems can be solved by using any of the search strategies. The general form of the constraint satisfaction procedure is as follows:

Until a complete solution is found or until all paths have led to lead ends, do

1. select an unexpanded node of the search graph.

2. Apply the constraint inference rules to the selected node to generate all possible new

constraints.

3. If the set of constraints contains a contradiction, then report that this path is a dead end.

4. If the set of constraints describes a complete solution then report success.

5. If neither a constraint nor a complete solution has been found then apply the rules to generate new partial solutions. Insert these partial solutions into the search graph.

Example: consider the crypt arithmetic problems.

SEND

+ MORE

----------

MONEY

----------

Assign decimal digit to each of the letters in such a way that the answer to the problem is correct to the same letter occurs more than once , it must be assign the same digit each time . no two different letters may be assigned the same digit. Consider the crypt arithmetic problem.

SEND

+ MORE

-----------

MONEY

-----------

**CONSTRAINTS:-**

1. No two digit can be assigned to same letter.

2. Only single digit number can be assign to a letter.

1. No two letters can be assigned same digit.

2. Assumption can be made at various levels such that they do not contradict each other.

3. The problem can be decomposed into secured constraints. A constraint satisfaction approach may be used.

4. Any of search techniques may be used.

5. Backtracking may be performed as applicable us applied search techniques.

6. Rule of arithmetic may be followed.

Initial state of problem.

D=?

E=?

Y=?

N=?

R=?

O=?

S=?

M=?

C1=?

C2=?

C1 ,C 2, C3 stands for the carry variables respectively.

**Goal State**: the digits to the letters must be assigned in such a manner so that the sum is satisfied.

**Solution Process:**

We are following the depth-first method to solve the problem.

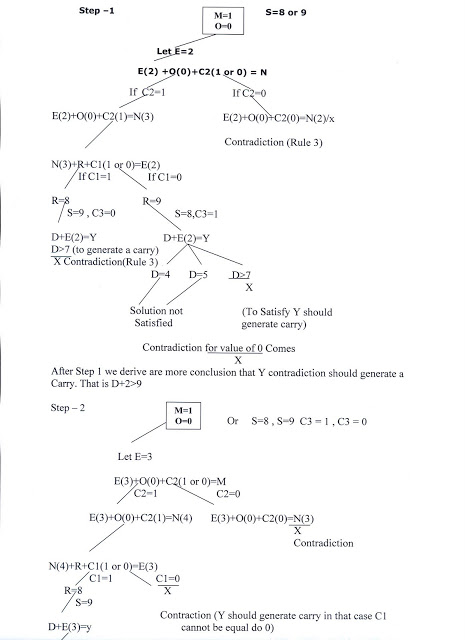
1. initial guess m=1 because the sum of two single digits can generate at most a carry '1'.

2. When n=1 o=0 or 1 because the largest single digit number added to m=1 can generate the sum of either 0 or 1 depend on the carry received from the carry sum. By this we conclude that o=0 because m is already 1 hence we cannot assign same digit another letter(rule no.)

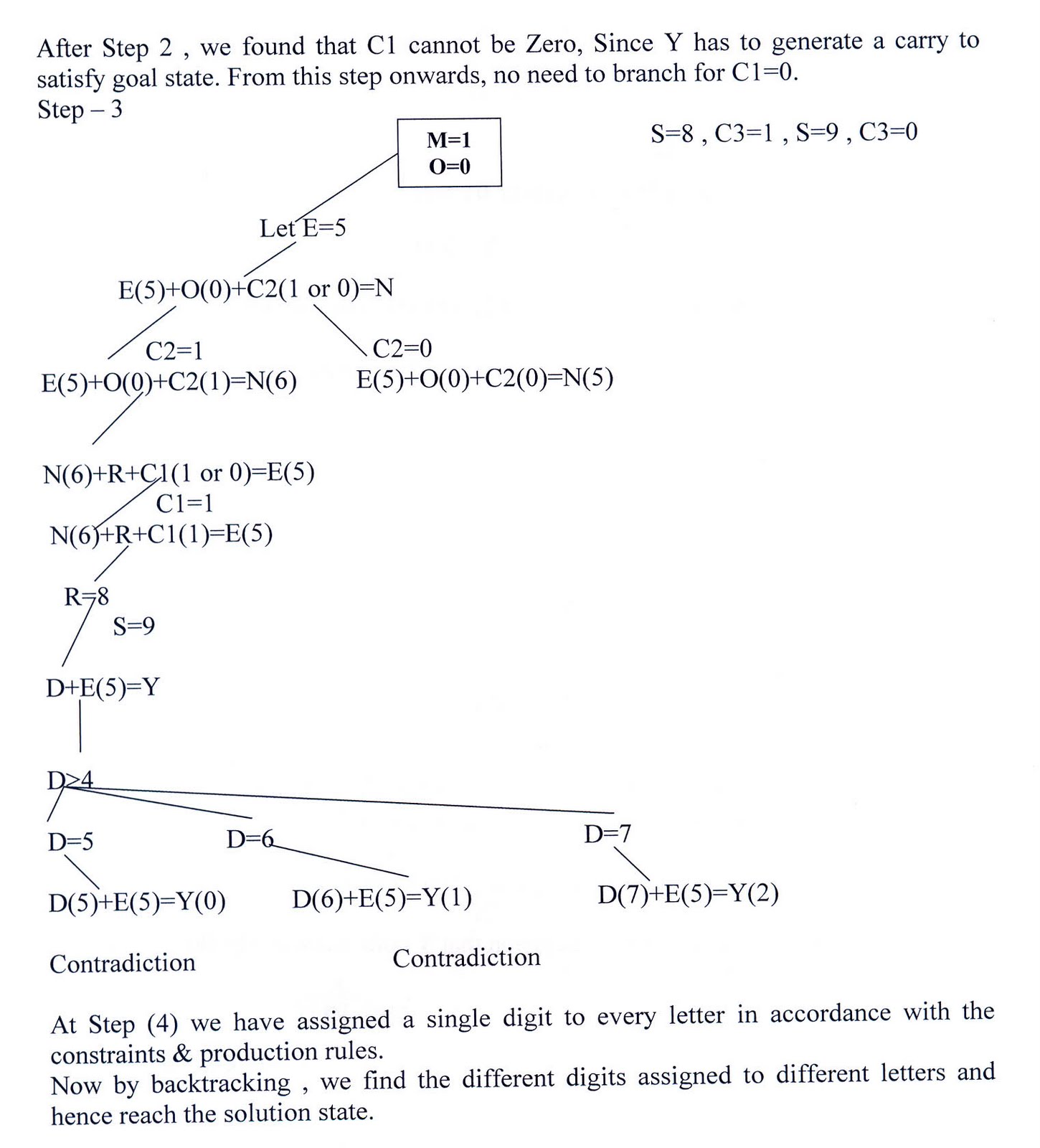
3. We have m=1 and o=0 to get o=0 we have s=8 or 9, again depending on the carry received from the earlier sum.

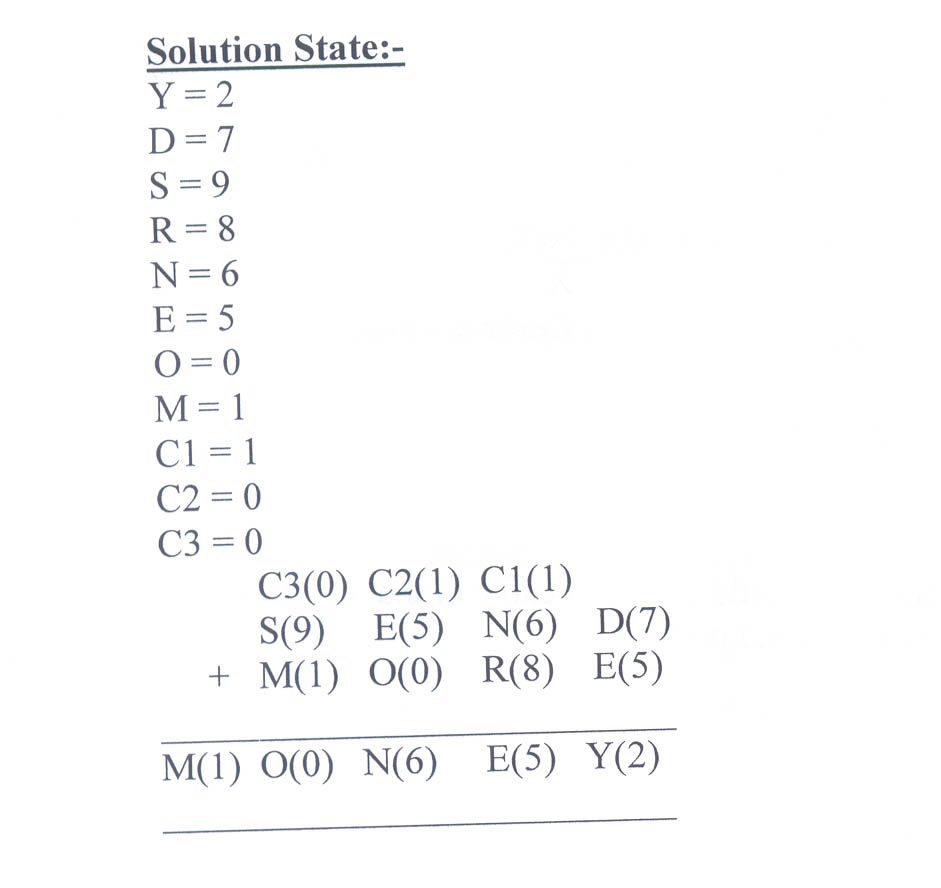
The same process can be repeated further. The problem has to be composed into various constraints. And each constraints is to be satisfied by guessing the possible digits that the letters can be assumed that the initial guess has been already made . rest of the process is being shown in the form of a tree, using depth-first search for the clear understandability of the solution process.

D>6(Controduction)



D>6(Controduction)





1. **A) consider the following sentences:**

* **John likes all kinds of food**
* **Apples are food**
* **Chicken is food**
* **Anything anyone eats and isn’t killed by is food.**
* **Bill eats peanuts and is still alive**
* **Sue eats everything Bill eats.**

1. **Translate the sentences into formulas in predicate logic.**
2. **Convert the formulas of part a into clause form**

i) John likes all kinds of food.

∀x: food(x) ∀likes(john, x)

2. Apples are food.

food(apple)

3. Chicken is food.

food(chicken)

4. Anything anyone eats and isn't killed by is food.

∀x:( y: ∃(eats(y, x) ¬ ∧ killedby(y, x)) → food(x)

5. Bill eats peanuts and is still alive.

A. eats(Bill, peanuts) B. alive(Bill)

6. Sue eats everything Bill eats.

∀x:eats(Bill, x) → eats(Sue, x)

7. ∀x: ∀y: alive(x) ¬ → killed by (x,y)

ii)**Clause form:**

1. ¬ food(x1) ∨ likes(John, x1)

2. food(apples)

3. food(chicken)

4. ¬ eats(y4,x4) ∨ killedby(y4 , x4) ∨ food(x4)

5. Eats (Bill, peanuts)

6. Alive (Bill)

7. ¬ eats(Bill,x7) v eats(Sue,x7)

8. ¬ alive(x8) ¬ ∨ killedby(x8, y8)

Or

b) **Trace the operation of the unification algorithm on each of the following pairs of literals:**

**i) f(Marcus) and f(Caesar)**

**ii) f(x) and f(g(y))**

**iii) f(marcus,g(x,y)) and f(x,g(Caesar, Marcus))**

In propositional logic it is easy to determine that two literals can not both be true at the same time. Simply look for L and ~L . In predicate logic, this matching process is more complicated, since bindings of variables must be considered.

For example man (john) and man(john) is a contradiction while man (john) and man(Himalayas) is not. Thus in order to determine contradictions we need a matching procedure that compares two literals and discovers whether there exist a set of substitutions that makes them identical . There is a recursive procedure that does this matching . It is called Unification algorithm.

In Unification algorithm each literal is represented as a list, where first element is the name of a predicate and the remaining elements are arguments. The argument may be a single element (atom) or may be another list. For example we can have literals as

( tryassassinate Marcus Caesar)

( tryassassinate Marcus (ruler of Rome))

To unify two literals , first check if their first elements re same. If so proceed. Otherwise they can not be unified. For example the literals

( try assassinate Marcus Caesar)

( hate Marcus Caesar)

Can not be Unfied. The unification algorithm recursively matches pairs of elements, one pair at a time. The matching rules are :

i) Different constants , functions or predicates can not match, whereas identical ones can.

ii) A variable can match another variable , any constant or a function or predicate expression, subject to the condition that the function or [predicate expression must not contain any instance of the variable being matched (otherwise it will lead to infinite recursion).

iii) The substitution must be consistent. Substituting y for x now and then z for x later is inconsistent. (a substitution y for x written as y/x)

The Unification algorithm is listed below as a procedure UNIFY (L1, L2). It returns a list representing the composition of the substitutions that were performed during the match. An empty list NIL indicates that a match was found without any substitutions. If the list contains a single value F, it indicates that the unification procedure failed.

UNIFY (L1, L2)

1. if L1 or L2 is an atom part of same thing do

(a) if L1 or L2 are identical then return NIL

(b) else if L1 is a variable then do

(i) if L1 occurs in L2 then return F else return (L2/L1)

© else if L2 is a variable then do

(i) if L2 occurs in L1 then return F else return (L1/L2)

else return F.

2. If length (L!) is not equal to length (L2) then return F.

3. Set SUBST to NIL

( at the end of this procedure , SUBST will contain all the substitutions used to unify L1 and L2).

4. For I = 1 to number of elements in L1 do

i) call UNIFY with the i th element of L1 and I’th element of L2, putting the result in S

ii) if S = F then return F

iii) if S is not equal to NIL then do

(A) apply S to the remainder of both L1 and L2

(B) SUBST := APPEND (S, SUBST) return SUBST.

**13)(a) Explain the production based knowledge representation technique.**

Knowledge representation formalism consists of collections of condition-action rules(Production Rules or Operators), a database which is modified in accordance with the rules, and a Production System Interpreter which controls the operation of the rules i.e The 'control mechanism' of a Production System, determining the order in which Production Rules are fired.

A system that uses this form of knowledge representation is called a production system.

A production system consists of rules and factors. Knowledge is encoded in a declarative from which comprises of a set of rules of the form

Situation ------------ Action

SITUATION that implies ACTION.

**Example:-**

IF the initial state is a goal state THEN quit.

The major components of an AI production system are

i. A global database

ii. A set of production rules and

iii. A control system

The goal database is the central data structure used by an AI production system. The production system. The production rules operate on the global database. Each rule has a precondition that is either satisfied or not by the database. If the precondition is satisfied, the rule can be applied. Application of the rule changes the database. The control system chooses which applicable rule should be applied and ceases computation when a termination condition on the database is satisfied. If several rules are to fire at the same time, the control system resolves the conflicts.

**Four classes of production systems:-**

1. A monotonic production system

2. A non monotonic production system

3. A partially commutative production system

4. A commutative production system.

**Advantages of production systems:-**

1. Production systems provide an excellent tool for structuring AI programs.

2. Production Systems are highly modular because the individual rules can be added, removed or modified independently.

3. The production rules are expressed in a natural form, so the statements contained in the knowledge base should the a recording of an expert thinking out loud.

**Disadvantages of Production Systems:-**

One important disadvantage is the fact that it may be very difficult analyse the flow of control within a production system because the individual rules don’t call each other.

Production systems describe the operations that can be performed in a search for a solution to the problem. They can be classified as follows.

Monotonic production system :- A system in which the application of a rule never prevents the later application of another rule, that could have also been applied at the time the first rule was selected.

**Partially commutative production system:-**

A production system in which the application of a particular sequence of rules transforms state X into state Y, then any permutation of those rules that is allowable also transforms state x into state Y.

Theorem proving falls under monotonic partially communicative system. Blocks world and 8 puzzle problems like chemical analysis and synthesis come under monotonic, not partially commutative systems. Playing the game of bridge comes under non monotonic , not partially commutative system.

For any problem, several production systems exist. Some will be efficient than others. Though it may seem that there is no relationship between kinds of problems and kinds of production systems, in practice there is a definite relationship.

Partially commutative , monotonic production systems are useful for solving ignorable problems. These systems are important for man implementation standpoint because they can be implemented without the ability to backtrack to previous states, when it is discovered that an incorrect path was followed. Such systems increase the efficiency since it is not necessary to keep track of the changes made in the search process.

Monotonic partially commutative systems are useful for problems in which changes occur but can be reversed and in which the order of operation is not critical (ex: 8 puzzle problem).

Production systems that are not partially commutative are useful for many problems in which irreversible changes occur, such as chemical analysis. When dealing with such systems, the order in which operations are performed is very important and hence correct decisions have to be made at the first time itself.

**(or)**

**13)b)**

**i) Discuss about Bayesian Theory and Bayesian Network**

Refer unit-3 part-b q.no-1 pg no 126-127 and q.no-9 pg.no144

**ii) Describe about Dempster –Shafer theory.**

Refer unit-3 part-b q.no-127-129

**14a) Write short notes on the**

1. **Learning by parameter Adjustment.** Here the learning system relies on evaluation procedure that combines information from several sources into a single summary static. For example, the factors such as demand and production capacity may be combined into a single score to indicate the chance for increase of production. But it is difficult to know apriori how much weight should be attached to each factor.

The correct weight can be found by taking some estimate of the correct settings and then allow the program modify its settings based on its experience. This type of learning systems is useful when little knowledge is available. In game programs, for example, the factors such as piece advantage and mobility are combined into a single score to decide whether a particular board position is desirable. This single score is nothing but a knowledge which the program gathered by means of calculation.

1. **Learning with Macro operators**

Sequence of actions that can be treated as a whole are called macro-operators. Once a problem is solved, the learning component takes the computed plan and stores it as a macro-operator. The preconditions are the initial conditions of the problem just solved, and its post conditions correspond to the goal just achieved.

The problem solver efficiently uses the knowledge base it gained from its previous experiences. By generalizing macro-operators the problem solver can even solve different problems. Generalization is done by replacing all the constants in the macro-operators with variables.The STRIPS, for example, is a planning algorithm that employed macro-operators in it’s learning phase. It builds a macro operator MACROP, that contains preconditions, post-conditions and the sequence of actions. The macro operator will be used in the future operation.

1. **Learning by chunking**

Chunking is similar to learning with macro-operators. Generally, it is used by problem solver systems that make use of production systems.

A production system consists of a set of rules that are in if-then form. That is given a particular situation, what are the actions to be performed. For example, if it is raining then take umbrella.

Production system also contains knowledge base, control strategy and a rule applier. To solve a problem, a system will compare the present situation with the left hand side of the rules. If there is a match then the system will perform the actions described in the right hand side of the corresponding rule.

Problem solvers solve problems by applying the rules. Some of these rules may be more useful than others and the results are stored as a chunk. Chunking can be used to learn general search control knowledge. Several chunks may encode a single macro-operator and one chunk may participate in a number of macro sequences. Chunks learned in the beginning of problem solving, may be used in the later stage. The system keeps the chunk to use it in solving other problems.

Soar is a general cognitive architecture for developing intelligent systems. Soar requires knowledge to solve various problems. It acquires knowledge using chunking mechanism. The system learns reflexively when impasses have been resolved. An impasse arises when the system does not have sufficient knowledge. Consequently, Soar chooses a new problem space (set of states and the operators that manipulate the states) in a bid to resolve the impasse. While resolving the impasse, the individual steps of the task plan are grouped into larger steps known as chunks. The chunks decrease the problem space search and so increase the efficiency of performing the task.

In Soar, the knowledge is stored in long-term memory. Soar uses the chunking mechanism to create productions that are stored in long-term memory. A chunk is nothing but a large production that does the work of an entire sequence of smaller ones. The productions have a set of conditions or patterns to be matched to working memory which consists of current goals, problem spaces, states and operators and a set of actions to perform when the production fires. Chunks are generalized before storing. When the same impasse occurs again, the chunks so collected can be used to resolve it.

**(Or)**

**b) Write down STRIPs style operators that corresponds to the following blocks**

**ON(A,B,S0)^**

A

**ONTABLE(B,S0)^**

B

**CLEAR9A,S0)**

Refer unit-4, part –b q.no 6 pg.no 165

**iii) Write short notes on Nonlinear planning using constraint posting.**

Let us reconsider the SUSSMAN ANOMALY

* Problems such as this one require subproblems to be worked on simultaneously.
* Thus a nonlinear plan using heuristics such as:
  1. Try to achieve ON(A,B) clearing block A putting block C on the table.
  2. Achieve ON(B,C) by stacking block B on block C.
  3. Complete ON(A,B) by stacking block A on block B.

Constraint posting has emerged as a central technique in recent planning systems (*E.g.* MOLGEN and TWEAK)

Constraint posting builds up a plan by:

* suggesting operators,
* trying to order them, and
* produce bindings between variables in the operators and actual blocks.

The initial plan consists of no steps and by studying the goal state ideas for the possible steps are generated.

There is *no order or detail* at this stage.

Gradually more detail is introduced and constraints about the order of subsets of the steps are introduced until a *completely ordered* sequence is created.

In this problem means-end analysis suggests two steps with end conditions ON(A,B) and ON(B,C) which indicates the operator STACK giving the layout shown below where the operator is preceded by its preconditions and followed by its post conditions:

CLEAR(B) CLEAR(C)

\*HOLDING(A) \*HOLDING(B) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

STACK(A,B) STACK(B,C) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ARMEMPTY ARMEMPTY

ON(A,B) ON(B,C)

tex2html_wrap_inline7182 CLEAR(B) tex2html_wrap_inline7182 CLEAR(C)

tex2html_wrap_inline7182 HOLDING(A) tex2html_wrap_inline7182 HOLDING(B)

NOTE:

* There is no order at this stage.
* Unachieved preconditions are starred (\*).
* Both of the HOLDING preconditions are unachieved since the arm holds nothing in the initial state.
* Delete post conditions are marked by (tex2html_wrap_inline7182).

Many planning methods have introduced heuristics to achieve goals or preconditions. The TWEAK planning method brought all these together under one formalism. Other methods that introduced/used the following heuristics are mentioned in brackets in the following section.

**15a) Explain the following expert systems.**

**i)MYCIN**

Refer unit-5 part-b pg.no:193 (from heuristics) to 196

**ii) DART**

Refer unit-5 part-b q.no-4 pg.no:200

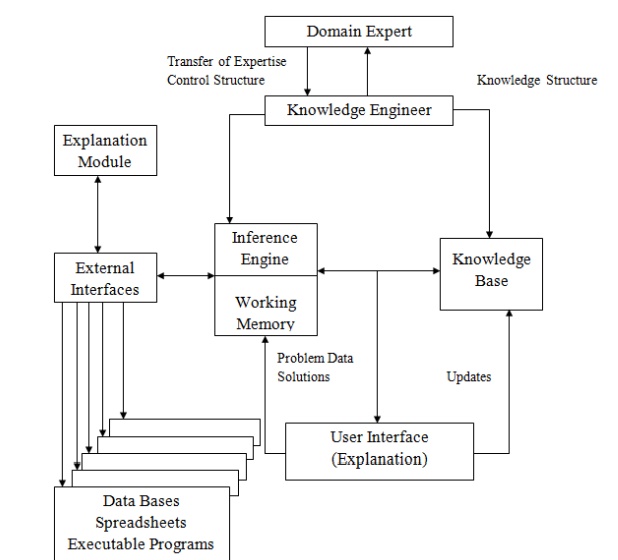
**(OR)**

**b) Explain the expert system architectures**

**i) Rule based system architecture**

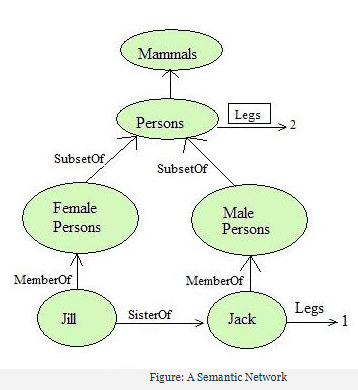
* The most common form of architecture used in expert and other types of knowledge based systems is the production system or it is called rule based systems.
* This type of system uses knowledge encoded in the form of production rules i.e. if-then rules.
* The rule has a conditional part on the left hand side and a conclusion or action part on the right hand side.
* Example

if: condition1 and condition2 and condition3 Then: Take action4

Each rule represents a small chunk of knowledge to the given domain of expertise. When the known facts support the conditions in the rule’s left side, the conclusion or action part of the rule is then accepted as known. The rule based architecture of an expert system consists of the domain expert, knowledge engineer, inference engine, working memory, knowledge base, external interfaces, user interface, explanation module, database spreadsheets executable programs as mentioned in figure.

**ii) Associative or Semantic Network Architecture.**

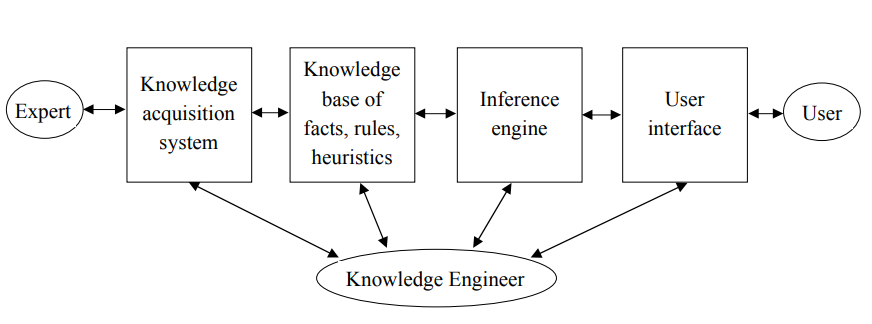
A semantic net (or semantic network) is a knowledge representation technique used for propositional information. So it is also called a propositional net. Semantic nets convey meaning. They are two dimensional representations of knowledge. Mathematically a semantic net can be defined as a labelled directed graph.Semantic nets consist of nodes, links (edges) and link labels. In the semantic network diagram, nodes appear as circles or ellipses or rectangles to represent objects such as physical objects, concepts or situations. Links appear as arrows to express the relationships between objects, and link labels specify particular relations. Relationships provide the basic structure for organizing knowledge. The objects and relations involved need not be so concrete. As nodes are associated with other nodes semantic nets are also referred to as associative nets.



* Semantic network allows us to perform inheritance reasoning
* Semantic network allows a common form of inference known as inverse links.
* Advantage of semantic nets is the ability to represent default values for categories.
* Drawback of semantic network is that the links between the objects represent only binary relations.

**iii) Network architecture**

The process of building expert systems is often called knowledge engineering. The knowledge engineer is involved with all components of an expert system: Building expert systems is generally an iterative process. The components and their interaction will be refined over the course of numerous meetings of the knowledge engineer with the experts and users.

****

**iv)Black Board system architecture**

A **blackboard system** is an artificial intelligence approach based on the blackboard architectural model  where a common knowledge base, the "blackboard", is iteratively updated by a diverse group of specialist knowledge sources, starting with a problem specification and ending with a solution. Each knowledge source updates the blackboard with a partial solution when its internal constraints match the blackboard state. In this way, the specialists work together to solve the problem. The blackboard model was originally designed as a way to handle complex, ill-defined problems, where the solution is the sum of its parts.

A blackboard-system application consists of three major components

1. The software specialist modules, which are called **knowledge sources (KSs)**. Like the human experts at a blackboard, each knowledge source provides specific expertise needed by the application.
2. The **blackboard**, a shared repository of problems, partial solutions, suggestions, and contributed information. The blackboard can be thought of as a dynamic "library" of contributions to the current problem that have been recently "published" by other knowledge sources.
3. The **control shell**, which controls the flow of problem-solving activity in the system. Just as the eager human specialists need a moderator to prevent them from trampling each other in a mad dash to grab the chalk, KSs need a mechanism to organize their use in the most effective and coherent fashion. In a blackboard system, this is provided by the control shell.

The following scenario provides a simple metaphor that gives some insight into how a blackboard functions:

A group of specialists are seated in a room with a large blackboard. They work as a team to brainstorm a solution to a problem, using the blackboard as the workplace for cooperatively developing the solution.

The session begins when the problem specifications are written onto the blackboard. The specialists all watch the blackboard, looking for an opportunity to apply their expertise to the developing solution. When someone writes something on the blackboard that allows another specialist to apply their expertise, the second specialist records their contribution on the blackboard, hopefully enabling other specialists to then apply their expertise. This process of adding contributions to the blackboard continues until the problem has been solved.

**16. a) Design an expert system for Travel recommendation and discuss its roles.**

**Expert system for Travel recommendation**

An expert system for tourism was developed to recommend a suitable travel schedule that satisfies the tourist's interest. The system is useful for tourists, and tourism agencies to select the best package based on the proper time, budget, and preferences of required tourist places. The system was designed as a rule based expert system and implemented with a prolog language. The system was evaluated and tested with a specialist and the results concluded were up to the level of human expert.

The system can provide tourists with information on the route and the distance between any two towns in the region. For example Tourism in Jordan

**Tourism categories**

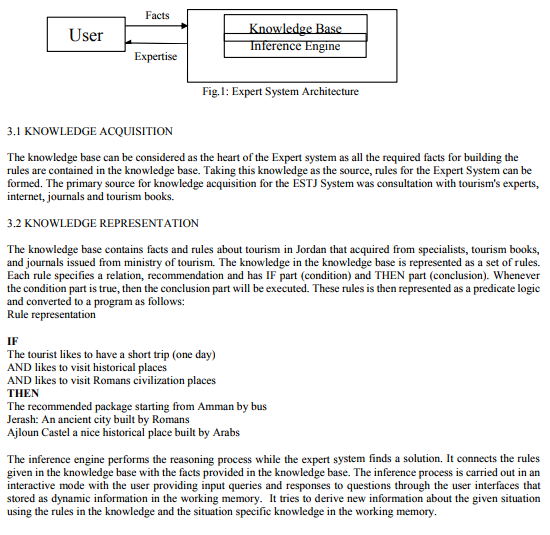
The tourism in Jordan can be classified into the following categories:

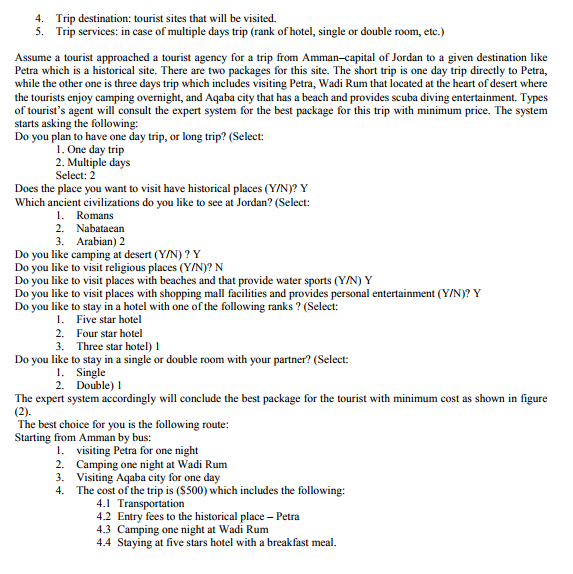
1. Historical
2. Religion
3. Beaches and water sports
4. Reserves
5. Entertainment
6. Shopping

7- Medical Treatment.

**Three parts**

The expert system was designed as a rule based expert system architecture and it is consisted from three parts as shown in figure (1). These parts are as follows: 1. Knowledge base 2. Inference engine 3. User interface



****

**(or)**

**b) Analyze any two machine learning algorithms with an example.**

**Supervised learning:**

Supervised learning can be explained as follows: use labeled training data to learn the mapping function from the input variables (X) to the output variable (Y).

Y = f (X)

Supervised learning problems can be of two types:

a. Classification: To predict the outcome of a given sample where the output variable is in the form of categories. Examples include labels such as male and female, sick and healthy.

b. Regression: To predict the outcome of a given sample where the output variable is in the form of real values. Examples include real-valued labels denoting the amount of rainfall, the height of a person.

The 1st 5 algorithms that we cover in this blog– Linear Regression, Logistic Regression, CART, Naïve Bayes, KNN are examples of supervised learning.

Ensembling is a type of supervised learning. It means combining the predictions of multiple different weak ML models to predict on a new sample. Algorithms 9-10 that we cover– Bagging with Random Forests, Boosting with XGBoost are examples of ensemble techniques.

**Unsupervised learning:**

Unsupervised learning problems possess only the input variables (X) but no corresponding output variables. It uses unlabeled training data to model the underlying structure of the data.

Unsupervised learning problems can be of two types:

a. Association: To discover the probability of the co-occurrence of items in a collection. It is extensively used in market-basket analysis. Example: If a customer purchases bread, he is 80% likely to also purchase eggs.

b. Clustering: To group samples such that objects within the same cluster are more similar to each other than to the objects from another cluster.

c. Dimensionality Reduction: True to its name, Dimensionality Reduction means reducing the number of variables of a dataset while ensuring that important information is still conveyed. Dimensionality Reduction can be done using Feature Extraction methods and Feature Selection methods. Feature Selection selects a subset of the original variables. Feature Extraction performs data transformation from a high-dimensional space to a low-dimensional space. Example: PCA algorithm is a Feature Extraction approach.

Algorithms 6-8 that we cover here - Apriori, K-means, PCA are examples of unsupervised learning.