SOLUTION FOR VTU EXAMINATION THOERY PAPER JAN 2019

Course Code: 15CS562

Course Name: Artificial Intelligence (AI)

Course Instructor: Manju S, Assistant Professor, CMRIT, Bangalore.

.

MODULE 1:

1.A. Define artificial intelligence and list the task domains of AI

- The study of how to make computers do things at which at the moment, people are better
- A field of study that seeks to explain and emulate intelligent behaviour in terms of computational processes
- Task domains of AI :

Mundane Tasks

- Perception
  - Vision
  - Speech
- Natural language
  - Understanding
  - Generation
  - Translation
- Commonsense reasoning
- Robot control

Formal Tasks

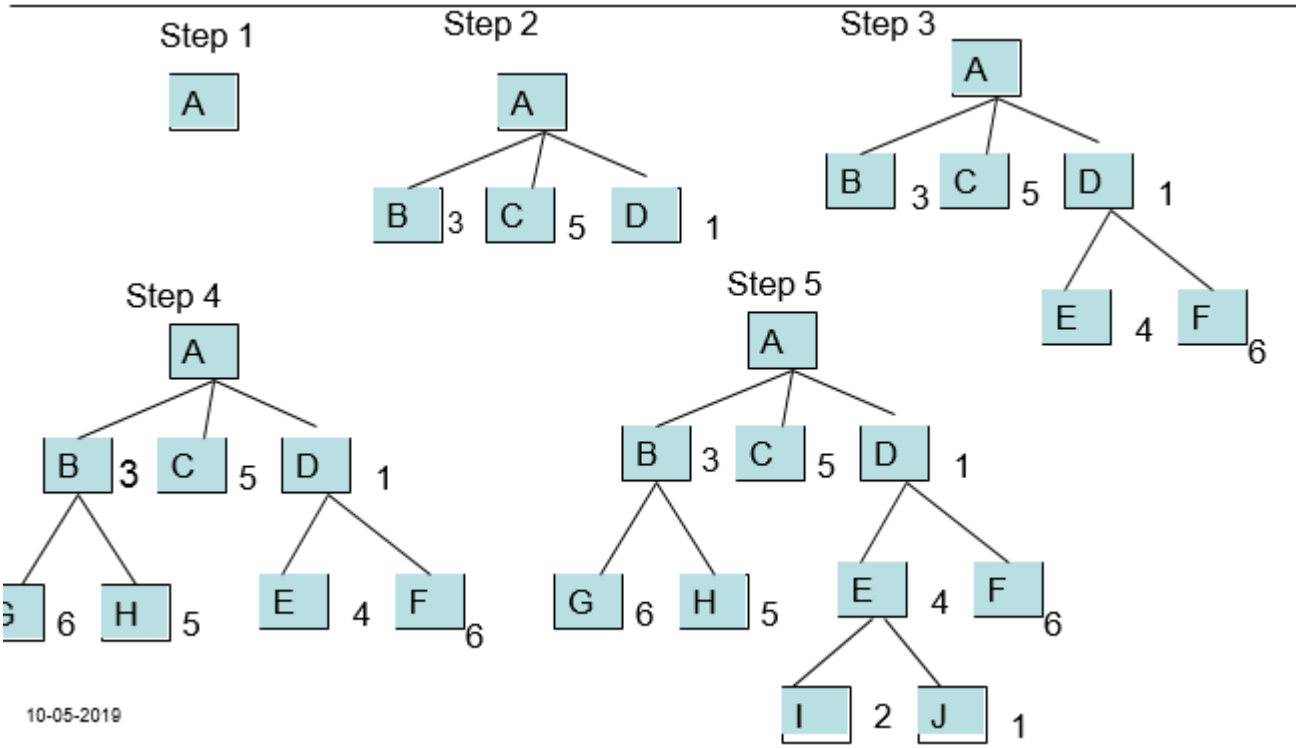
- Games
  - Chess
  - Backgammon
  - Checkers -Go
- Mathematics
  - Geometry
  - Logic
  - Integral calculus
  - Proving properties of programs

Expert Tasks

- Engineering
  - Design
  - Fault finding
  - Manufacturing planning
- Scientific analysis
- Medical diagnosis
- Financial analysis

1. B. state and explain best first search with an example
    - Combines the advantages of both DFS and BFS into a single method.
    - DFS is good because it allows a solution to be found without all competing branches having to be expanded.
    - BFS is good because it does not get branches on dead end paths.
    - One way of combining the two is to follow a single path at a time, but switch paths whenever some competing path looks more promising than the current one does.

1. Start with *OPEN* containing just the initial state.
2. Until a goal is found or there are no nodes left on *OPEN* do:
   (a) Pick them best node on *OPEN*.
   (b) Generate its successors.
   (c) For each successor do:
      (i) If it has not been generated before, evaluate it, add it to *OPEN*, and record its parent.
      (ii) If it has been generated before, change the parent if this new path is better than the previous one. In that case, update the cost of getting to this node and to any successors that this node may already, have.



1. C. Explain about Production systems.

- Structure of AI programs that facilitates search process. A production system consists of :
- A set of rules , each consisting of a left side that determine the applicability of the rule and a right side that describes the operation to be performed if the rule is applied.
- One or more knowledge/databases that contain the appropriate information.
- A control strategy that specifies the order in which the rules will be compared to the database and a way of resolving the conflicts when several rules match at once.
- A rule applier.
- Types of production system:
- **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.
- **Non-monotonic production system:** A system in which the application of a rule **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** is a production system with the property that if the application of a particular sequence of rules transforms state x into state y then any permutation (any order) of those rules that is allowable (i.e. each rules preconditions are satisfied when it is applied) also transforms state x into state y.
- **Commutative production system:** Production system which is both **monotonic and partially commutative.**

2. A. Give the production rules and solution for the following water jug problem
    - **State Representation** – we will represent a state of the problem as a tuple (x, y) where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-gallon jug.
    - Space: $0 \leq x \leq 4$, and $0 \leq y \leq 3$.
    - Initial state: (0,0)
    - Goal state: (2,n)

1. (x,y) If x<4	(4,y)	Fill the 4 gallon jug
2. (x,y) If y<3	(x,3)	Fill the 3 gallon jug

3. (x,y) If x>0	(0,y)	Empty the 4 gallon jug on the ground
4. (x,y) If y>0	(x,0)	Empty the 3 gallon jug on the ground
5. (x,y) If x+y>=4 and y>0	(4,y-(4-x))	Pour water from 3-gallon jug in to the 4 gallon jug until the 4 gallon jug is full
6. (x,y) If x+y>=3 and x>0	(x-(3-y),3)	Pour water from 4-gallon jug in to the 3 gallon jug until the 3 gallon jug is full
7. (x,y) If x+y<=3 and x>0	(0,x+y)	Pour all water from 4-gallon jug in to the 3 gallon jug
8. (x,y) Ifx+y<=4 and y>0	(x+y,0)	Pour all water from 3-gallon jug in to the 4 gallon jug
9. (x,y) If x>0	(x-d,y)	Pour some water out of 4 gallon jug
10. (x,y) If y>0	(x,y-d)	Pour some water out of 3 gallon jug

Solution:

4-Gallon jug	**3-Gallon jug**	**Rule Applied**
0	0	1
4	0	6
1	3	4
1	0	7
0	1	1
4	1	7
2	3	Final state

2.B. Explain about simulated annealing

- Variation of simple hill climbing

- Does enough exploration of whole search space earlier so that final solution is relatively insensitive to starting state.
- Thereby lowering the chances of getting caught at local maxima, plateau and ridges
- The term simulated annealing derives from the roughly analogous physical process of heating and then slowly cooling a substance to obtain a strong crystalline structure.
- Annealing : heat a metal or glass and allow it to cool slowly, in order to remove internal stresses and toughen it.
- P= 1/ e to the power of change in energy divided by k.T where T is temperature and k is some Boltzmann's constant.
- Rate at which system is cooled is called as annealing schedule
- Applying this and developing an analogy,
  - change in energy is change in value of objective function
  - P= 1/ e to the power of change in energy divided by T where T is temperature
- Algorithm :
- Evaluate the initial state. If it is also a goal state, then return it and quit. Otherwise, continue with initial state as current state.
- Initialize BEST-SO-FAR to current state.
- Initialize T according to the annealing schedule
- Loop until a solution is found or until there are no new operators left to be applied in the current state:
  - Select an operator that has not yet been applied to the current state and apply it to produce a new state
  - Evaluate the new state. Compute
  - change in E = value of current – value of new state
- If it is the goal state, then return it and quit.
- (+ve) If it is not a goal state but it is better than the current state, then make it the current state. Also set BEST -SO- FAR to this new state
- ((-ve) worst move) If it is not better than the current state, then make it the current state with probability p' by invoking a random number from 0 to 1.If that number is less than p' then move is allowed, otherwise do nothing.
- c. Revise T
- Return BEST-SO-FAR as answer.

2.C. Explain AND OR Graphs with respect to problem reduction

- AND-OR graph (or tree) is useful for representing the solution of problems that can be solved by decomposing them into a set of smaller problems, all of which must then be solved.

- One AND arc may point to any number of successor nodes, all of which must be solved in order for the arc to point to a solution.
- Decompose problem into “and” and “or”
- FUTILITY is chosen to correspond to a threshold such that any solution with a cost above it is too expensive to be practical, even if it could ever be found.
- If expected cost exceeds the FUTILITY value then process should be aborted.

Algorithm: Problem Reduction

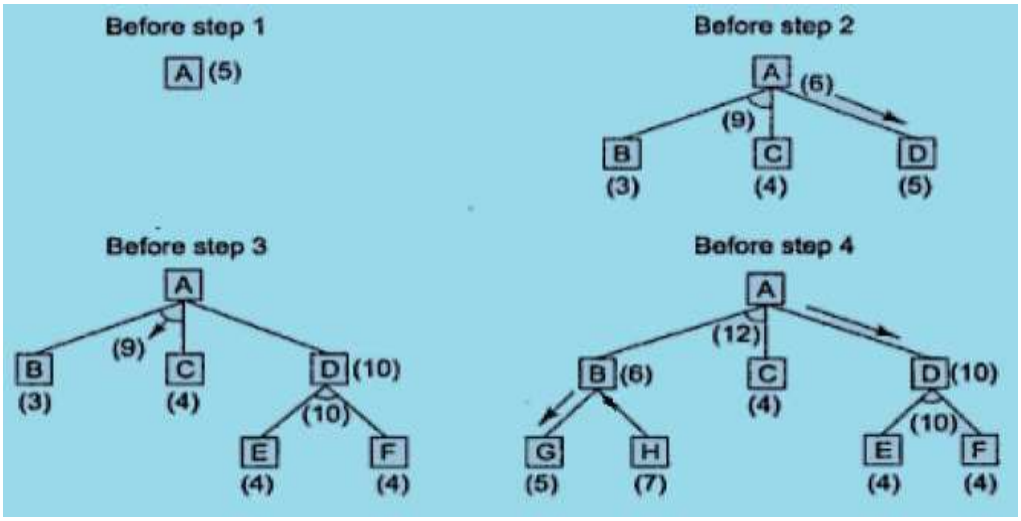
1. Initialize the graph to the starting node.

2. Loop until the starting node is labeled SOLVED or until its cost goes above FUTILITY:

a. Traverse the graph, starting at the initial node and following the current best path, and accumulate the set of nodes that are on that path and have not yet been expanded or labeled as solved.

b. Pick one of these nodes and expand it. If there are no successors, assign FUTILITY as the value of this node. Otherwise, add its successors to the graph and for each of them compute f’. If f’ of any node is 0, mark that node as SOLVED.

c. Change the f’ estimate of the newly expanded node to reflect the new information provided by its successors. Propagate this change backward through the graph. This propagation of revised cost estimates back up the tree was not necessary in the BFS algorithm because only unexpanded nodes were examined. But now expanded nodes must be reexamined so that the best current path can be selected



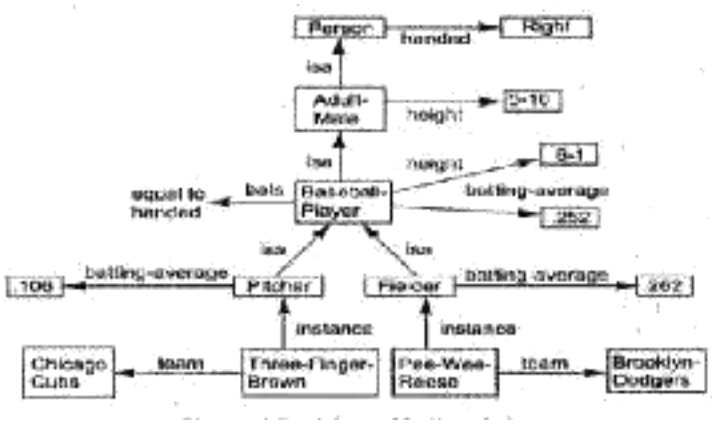
MODULE 2:

3.a. Explain approaches of knowledge representation

- Simple relational knowledge
  - Simplest way to represent declarative facts is as set of relations of the same sort used in database systems.
  - Each fact about a set of objects is set out systematically in columns.
  - Very weak inferential capabilities
  - Who is the heaviest player?
  - Procedure to compute the answer from facts is not given
  - If it was given then the given Simple relational knowledge representation would compute answer

Player	Height	Weight	Bats-Throws
Hank Aaron	6-0	180	Right-Right
Willie Mays	5-10	170	Right-Right
Babe Ruth	6-2	215	Left-Left
Ted Williams	6-3	205	Left-Right
player_info('hank aaron', '6-0', 180,right-right).			

- Inheritable knowledge
  - Knowledge is made up of objects consisting of attributes and corresponding associated values.
  - Boxed nodes: objects and values of attributes of objects. Values can be objects with attributes and so on.
  - Arrows: point from object to its value.
  - This structure is known as a slot and filler structure, semantic network or a collection of frames.
  - Property inheritance : Elements of specific class inherits attributes and values of general class



- Inferential knowledge
  - Represent knowledge as formal logic
  - Advantages:
  - A set of strict rules.
  - Can be used to derive more facts.
  - Truths of new statements can be verified.
  - Guaranteed correctness
  - E.g.: 1. "Marcus is a man" 2. "All men are mortal"
  - **Implies: 3. "Marcus is mortal"**
- Procedural knowledge
  - Representation of "how to make it" rather than "what it is".
  - May have inferential efficiency, but no inferential adequacy and acquisitional efficiency.
  - Ex. Writing LISP programs i.e. artificial intelligence list processing
  - Procedural knowledge can also represented using if then rules

3.b. Write a note on control knowledge

- Knowledge about which paths are most likely to lead quickly to a goal state is often called search control knowledge.
- Knowledge about which states are more preferable to others.
- Knowledge about the order in which to pursue sub goals.
- Knowledge about which rule to apply in given situation.
- Knowledge about useful sequences of rules to apply.
- Control knowledge is represented using control rule.
- Case study : Eg : SOAR and PRODIGY

4.a. State the algorithm to unify(L1,L2)

- In propositional it is easy to determine that two literals cannot both be true at same time. E.g. L and ~L
- In predicate matching process is complicated since arguments of predicates should be considered.

- Eg : man(John) and ~man(john) is a contradiction
- Eg: man(John) and ~man(Spot) is not a contradiction
- This determines contradictions by using a matching procedure that compares two literals and discovers whether there exist a set of substitutions that makes them identical.
- This matching is done by unification algorithm.

Algorithm:

1. If L1 or L2 are both variables and constants, then:

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

(b) Else if L1 is a variable then if L1 occurs in L2 then return F else return (L2/L1)

(c) Else if L2 is a variable then if L2 occurs in L1 then return F else return (L1/L2)

(d) Else return F.

2. If the initial predicate symbols in L1 and L2 are not identical then return F.

3. If L1 and L2 have a different number of arguments then return F.

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

5. For i = 1 to number of arguments 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)

6. Return SUBST.

4.b. Write an algorithm for conversion into clause form.

- Eliminate $\rightarrow$ using that a $\rightarrow$ b is equivalent to $\sim a \vee b$
- Reduce the scope of each ~ to a single term using a fact that $\sim(\sim p)=p$ or $\sim(a \vee b)=\sim a \wedge b$
- Standardize variables so that each quantifier binds a unique variable. $\forall x:P(x) \vee \forall x:Q(x)$ should be converted to $\forall x:P(x) \vee \forall y :Q(y)$
- Move all the quantifiers to left of the problem without changing their relative order

  $\forall x: \forall y : P(x) \vee Q(y)$
- Eliminate existential quantifiers

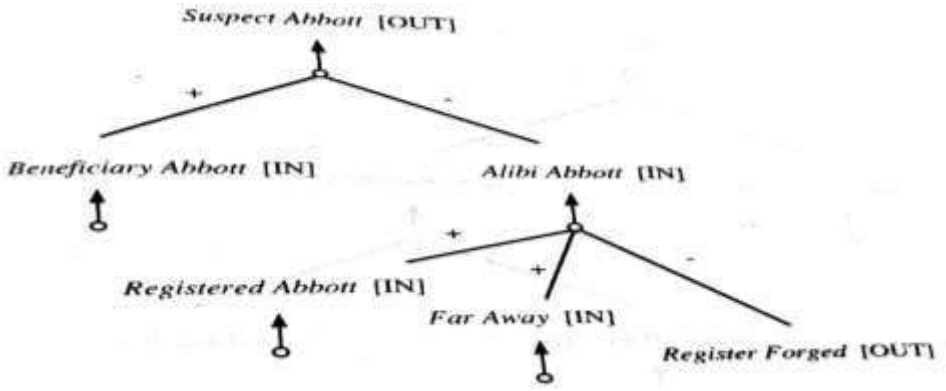
- ∃ z:President(y) is converted to President(p) where p is a value for the predicate
- Drop the prefix i.e all the quantifiers
- Convert the matrix into conjunction of disjuncts

  $(a \vee (b \wedge c)$ is converted into $(a \vee b) \wedge (a \vee c)$
- Create a separate clause corresponding to each conjunct.
- Standardize apart the variables in the set of obtained clauses. So that no two clauses make reference to same variables.

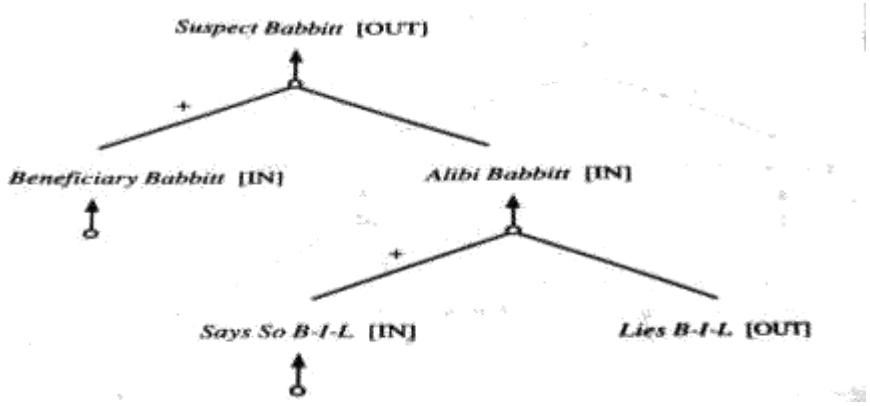
MODULE 3:

5.a. Write a note on JTMS with an example.

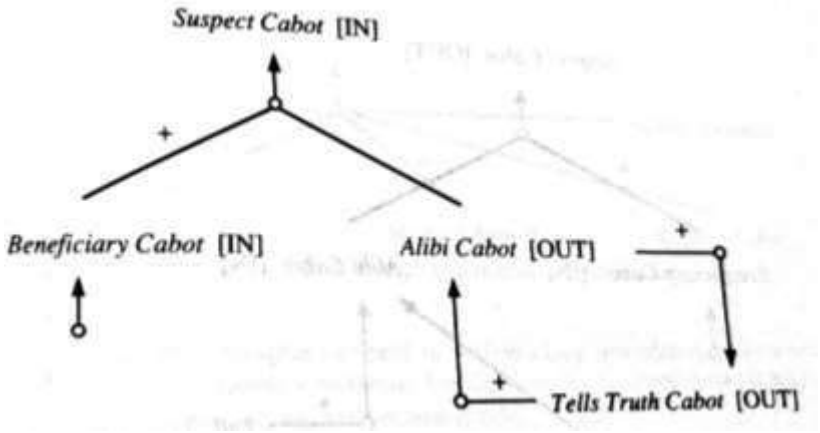
- Representation has two things associated with the network, one is justification and other is supported belief.
- Two parts to be considered in justification part is IN-list and OUT-list
- Proof: Abbott is suspect?!
- Assumption :
  - Abbott is the primary murder suspect
    - Suspect Abbott
  - Abbott is the beneficiary of the victim
    - Beneficiary Abbott
  - Abbott was at an Albany hotel at the time
    - Alibi Abbott
- DL representation : Beneficiary (x) :~alibi(x)/ Suspect(x)
- Assertions in IN list are connected to justification by + symbol
- Assertions in OUT list are connected to justification by - symbol.
- Justification is connected to belief through arrows
- Justification is valid when assertions in IN list are all believed and none of the assertions in OUT list is believed
- Abbott no longer a suspect



- Babbitt no longer a suspect



- Cabot no longer a suspect



5.b. Write note on non-monotonic and default logic

Non Monotonic Logic

- Formal logic in which the language of first order predicate logic is augmented with a modal operator M, which is read as consistent.
- E.g : $\forall$ x,y related(x, y) $\wedge$ M getalong(x,y) $\rightarrow$ willdefend(x,y)
- For all x and y if x and y are related and if the fact that x gets along with y is consistent with everything else that is believed then conclude that x will defend y.
- Issues faced during augmenting,
- Theories in non-monotonic logic is undecidable which we want to make at least semi decidable by including some assumptions. So it is necessary to define consistency on some heuristic basis.
- Inconsistency arises when multiple non monotonic statements are combined.
- Example :

- Republican(x) ∧ M ~Pacifist(x) → ~Pacifist(x)
- Quaker(x) ∧ M Pacifist(x) → Pacifist(x)
- Republican(Dick)
- Quaker(Dick)
- Dick is pacifist?
- A ∧M B → B and ~A ∧ M B → B then we can derive an expression MB → B

Default logic:

- DL: invented by Reiter.
- Add some sentence which is assumed to be true.
- DL: Alternative logic for performing default based reasoning in which new class of inference rules is introduced.
- Inference rules is of the form :- A:B/ C
- Read as "If A is provable and it is consistent to assume B then conclude C".
- i.e. A prerequisite and B is justification and C is conclusion
- New inference rules are used as a basis for computing a set of plausible extensions to the knowledge base (without affecting consistency)
- Non monotonic expressions are rules of inference rather than expression in case of DL.
- Example :
- Knowledge base : Tweety is a bird
- To prove : Tweety can fly
- Then add an assumption that tweety can fly because all birds can fly
- Eg : Bird(tweety) : flies(birds)/ flies(tweety)
- Other normal rule of DL :- A:B/ B
- If B is consistent then conclude B
- Eg : Bird(tweety) : flies(tweety)/ flies(tweety)
- Meaning of DL : Add some default assumptions as rules called default rules to the knowledge base

5.c. Explain abduction and inheritance

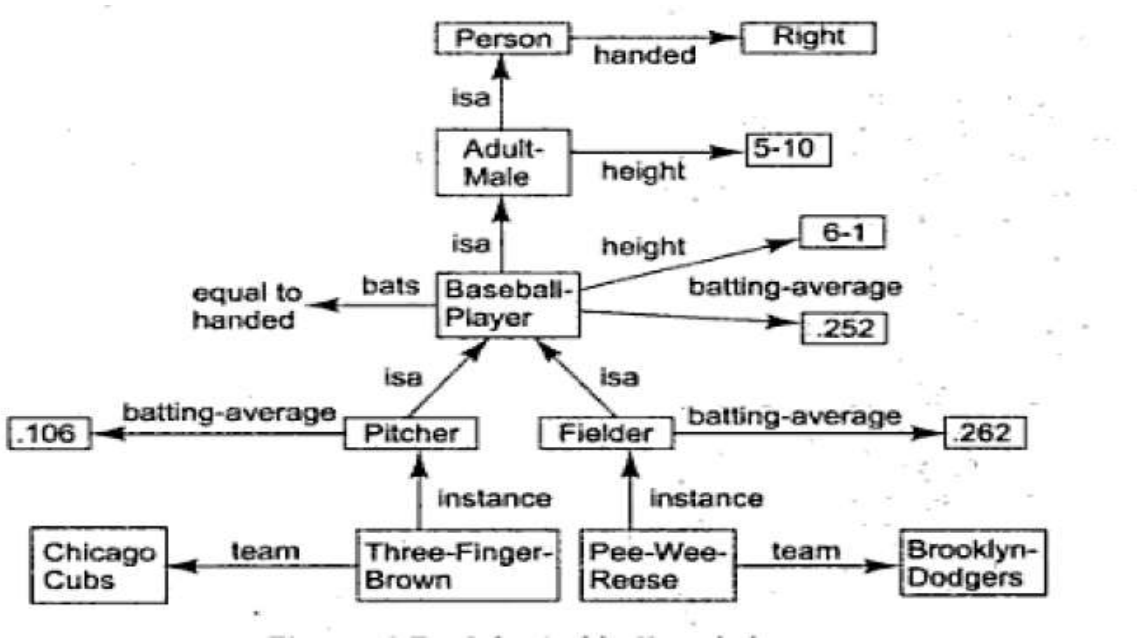
Abduction :

- Given two axioms,
  - " x: A(x)→B(x)
  - A(C)
- Conclusion : B(C) by deduction (Implication in Forward direction)

- How about applying implication in reverse?
- Eg: " x: Measles(x)→Spots(x)
- Meaning : If x has measles then it has spots
- Reverse : If x has spots then x has measles is wrong in standard logic
- Deriving conclusions in this way is thus called as abductive reasoning

Inheritance :

- Basis of inheriting attribute values from a prototype description of a class to the individual entities that belong to class
- E.g.: Baseball game
- Rule to account for the inheritance ,Height of baseball player: BaseballPlayer(x) : height(x,6-1)/height(x,6-1)



- Conclusion: what is the height of three finger brown? It is 6-1 because he is a baseball player.
- " x,y,z: height(x,y) Ù height(x,z) → y = z .
- i.e. It prohibits someone from having more heights.
  Adult-male(x):height(x,5-10)/ Height(x,5-10)
- This rule doesn't work because first rule fires height of three finger brown as 6-1 and second one as 5-10 thus arises inconsistency
- In order to get value from more specific category,
- Adult-male(x) Ù ~BaseballPlayer(x) :height(x,5-10)/height(x,5-10)
- Some set of exceptions, (Find common exceptions and include)
- Adult-male(x) Ù ~BaseballPlayer(x) Ù ~midget(x) Ù ~jockey(x) :height(x,5-10)/ height(x,5-10)

6.a. Write notes on Dempster Shafer theory [Jan'19]

- Discussed are the techniques in which proposition is assigned with a single value i.e. degree of belief
- Technique where propositions are assigned with an interval
- [Belief, Plausibility]
- Belief [bel] : Strength of evidence in favor of set of propositions
- Ranges from 0 (No evidence) to 1(Certainty)
- Plausibility [pl] : pl(s)=1-Bel(~s)
- Pl ranges from 0 to 1.
- When there is a evidence for ~s then Bel(~s) becomes 1 hence pl(s) gets a value 0.
- E.g. : A,B and C be the hypothesis in the knowledge base
- Then the interval will be from [0,1] without having any information or evidence.
- Upon addition of evidences the interval might shrink
- Θ might consist of set {All, Flu, Cold, Pneu}
- All : Allergy, Flu : Flu, Cold : Cold, Pneu : Pneumonia
- Goal : To attach some measure of belief to elements of Θ
- Addition of measures might extent support to more elements also and can extend support indirectly.
- E.g. : Fever might support Flu, Cold, Pneu
- m: Probability density function defined for elements of Θ
- No information about evidences as of now
- Hence we define m as {Θ} (1.0)
- i.e. Answer is in the set { All, Flu, Cold, Pneu}
- Suppose the evidence suggest that correct diagnosis is in the set {Flu, Cold, Pneu} with an evidence of 0.6 then m is redefined as,
  {Flu, Cold, Pneu} (0.6)
  {Θ} (0.4)
- Suppose we are given with two belief functions m1 and m2 and let X be the subsets of Θ to which m1 assigns non zero value and let Y be the corresponding set for m2.
- Then m3 is the combination of m1 and m2

$$m_3(Z) = \frac{\sum_{X \cap Y = Z} m_1(X) \cdot m_2(Y)}{1 - \sum_{X \cap Y = \emptyset} m_1(X) \cdot m_2(Y)}$$

- Suppose m1 corresponds to our belief observing fever
  {Flu,Cold,Pneu} (0.6)
  Θ (0.4)
- Suppose m2 corresponds to our belief observing runny nose
  {All,Flu,Cold} (0.8)
  Θ (0.2)
- Four sets by combining two sets are the intersection of m1 and m2
- m3 value is computed by multiplying the corresponding value of m1 and m2
  - m3:

    {Flu,Cold,Pneu} (0.48)

    {All, Flu, Cold} (0.32)

    {Flu, Cold, Pnue} (0.12)

    Θ (0.08)

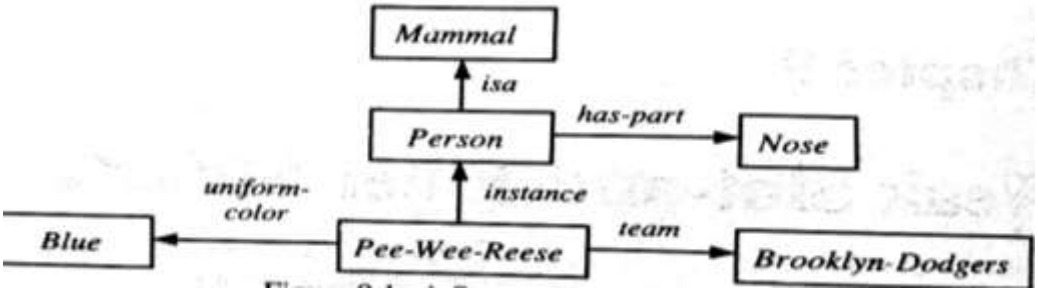
  - m4:
    {All} (0.9)
    Θ(0.1)
  - Total belief associated with null is 0.54 and hence 0.45 is associated with outcomes that are possible
  - Scaling : 1-0.54=0.46

		$\{A\}$	(0.9)	Θ	(0.1)
$\{F, C\}$	(0.48)	$\emptyset$	(0.432)	$\{F, C\}$	(0.048)
$\{A, F, C\}$	(0.32)	$\{A, F, C\}$	(0.288)	$\{A, F, C\}$	(0.032)
$\{F, C, P\}$	(0.12)	$\emptyset$	(0.108)	$\{F, C, P\}$	(0.012)
Θ	(0.08)	$\{A\}$	(0.072)	Θ	(0.008)

6.b. Define semantic network with an example

- Information/objects/values of an attribute are represented as nodes
- Relationship between information is represented as arcs

- Has ‘Isa and instance and



domain specific relations’

- Semantic nets were used to find the relationship between 2 objects by spreading the activation out from 2 nodes and seeing where the activation meets.
- E.g.: What is the connection between the Brooklyn Dodgers and blue?
- Binary predicates of above network
- isa(Person, Mammal)
- instance(Pee-Wee-Reese, Person)
- team(Pee-Wee-Reese, Brooklyn-Dodgers)
- uniform-color(Pee-Wee-Reese, Blue)
- Unary Predicate
- man(Marcus) is converted as binary predicates as
- instance(Man, Marcus)
- or more place predicates: Create a new object representing the entire predicate statement and introduce binary predicates

6.c State Bayes’ theorem

- Bayes' theorem is based on **conditional probability**.
- E.g. : India’s chance of winning game after winning toss
- P(H/E) i.e. Probability of hypothesis given the observed evidence E
- To compute P(H/E) include prior probability i.e. probability assigned without evidence and the extent until which E can act as evidence of H.
- $P(Hi/E) = P(E/Hi) . P(Hi) / \sum_{i=1} P(E/Hn).P(Hn)$

Where

- P(Hi/E) : Probability that Hi is true given the evidence E
- P(E/Hi) : Probability that we will observe evidence E given that Hi is true
- P(Hi) : Prior probability that Hi is true in the absence of any specific evidence
- K : Number of hypotheses.

MODULE 4:

7.a Explain conceptual dependency with its goals and representation.

- This represents the knowledge obtained from the events of natural language sentences.

Goals of CD:

- Facilitates drawing inferences from the sentences
- Is independent of the language in which the sentences were originally stated
- CD is based upon events and actions. Every event (if applicable) has:
- an ACTOR
- an ACTION performed by the Actor
- an OBJECT that the action performs on
- a DIRECTION in which that action is oriented i.e. dependency direction
- These are represented as slots and fillers.

Representations:

- ATRANS - Transfer of an abstract relationship(Eg: give)
- PTRANS - Transfer of the physical location of an object(Eg: go)
- PROPEL - Application of physical force to an object (Eg: push)
- MOVE - Movement of a body part by its owner (Eg : kick)
- GRASP - Grasping of an object by an actor(Eg: throw)
- INGEST - Ingesting of an object by an animal (Eg: eat)
- EXPEL - Expulsion of something from the body of an animal (cry)
- MTRANS - Transfer of mental information(Eg: tell)
- MBUILD - Building new information out of old(Eg: decide)
- SPEAK - Production of sounds(Eg: say)
- ATTEND - Focusing of sense organ towards a stimulus (Eg: listen)

- 4 conceptual categories from which dependency structures can be built are,
- ACTs –Real world Actions
- PPs – Real world Objects (Picture Producers)
- AAs – Modifiers of actions (Action Aiders) i.e. Attributes of actions
- PAs – Modifiers of PPs(Picture Aiders) i.e. Attributes of objects

Representation of tenses

- Past - p
- Future - f
- Transition- t
- Start Transition - ts
- Finished transition - tf
- Continuing - k
- Interrogative - ?
- Negative - /
- Present - nil
- Timeless - delta
- Conditional – c

Rules of dependencies:

- Rule1: Describes the relationship between an actor and event he/she causes. This is a two way dependency since neither actor nor event can be primary.
- Rule 2: Describes the relationship between a PP and a PA that is being asserted to describe it. Many state descriptions like height are represented in CD as numeric scales.
- Rule 3: describes the relationship between two PPs, one of which belongs to the set defined by other.
- Rule 4: describes the relationship between a PP and an attribute that has already been predicated of it. Direction of arrow is towards the PP being described.
- Rule 5: describes the relationship between two PPs, one of which provides a particular kind of information about other.
- Rule 6 : Describes the relationship between an ACT and the PP. Arrow directed towards the ACT

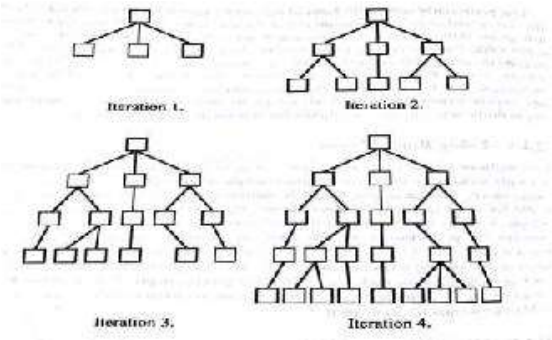
- Rule 7: Relationship between the ACT and the source and the recipient of the ACT.
- Rule 8: Describes the relationship between an ACT and the instrument with which action is performed.
- Rule 9: Describes the relationship between an ACT and its physical source and destination.
- Rule 10 : Describes the relationship between a PP and a state in which it started and another in which it ended
- Rule 11: Describes the relationship between one conceptualization and another that causes it. Arrows indicate dependency of one conceptualization on another and so point in the opposite direction of the implication arrows.
- Rule 12: Describes the relationship between a conceptualization and the time at which the event it describes occurred.
- Rule 13: Describes the relationship between one conceptualization and another that happened during the time of the first.
- Rule 14: Describes the relationship between a conceptualization and the place at which it occurred.

  *Refer ppt provided for symbolic representation of each rule

7.b Give the reasons to build large databases

- Brittleness: Specialized knowledge base are brittle. Performance degradation is not acceptable. So build large knowledge base such that we build firmer foundations
- Form and Content : Analyze where the complexity arises by coding all the common sense knowledge
- Shared knowledge : All systems that share same primitives should communicate easily so that all the small knowledge based systems and large should find some similar way of representing things

7.c. Write a note on iterative deepening



- Rather than searching to a fixed depth in a game tree , the same is searched as follows
- Single ply first, apply static evaluation function , find possible moves
- Initiate minimax again to the depth of two-ply, followed by three-ply, 4-ply and etc.,
- i.e. On each iteration the tree is searched one level deeper.
- Combination of depth first and breadth first search
- Fixes depth and restricts DFS from going beyond the depth, so it is DFS in BFS fashion

Algorithm: Depth-First Iterative Deepening

1. Set SEARCH-DEPTH = 1.
2. Conduct a depth-first search to a depth of SEARCH-DEPTH. If a solution path is found, then return it.
3. Otherwise, increment SEARCH-DEPTH by 1 and go to step 2.

- A* : Guaranteed to find optimal solution provided that h' is an admissible heuristic
- Efficient with respect to space

8.a. Write a note on global ontology

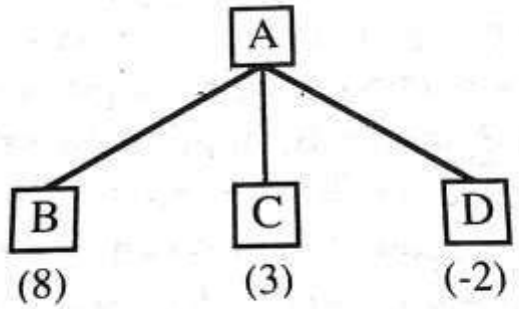
- Ontology : Study of what exists in the universe, nature of being
- Global Ontology : to understand what things exist and what their general properties are and how things are related ?
- Highest level of concept in CYC is 'thing'. Hence everything will be a instance of thing
- IndividualObject versus Collection : Collections is class. E.g.: Person is a collection while Fred is an object
- Intangible, Tangible and Composite :
  - Intangible are those which has no mass. E.g.; Sets, numbers, laws and events.
  - Tangible are those which has mass.E.g. Person's body or an apple.
  - Composite has both physical extent and intangible extent. E.g.: Person where his body is a physical extent and his mind is a intangible extent
- Intrinsic vs. extrinsic properties : A property is intrinsic if when an object has that property, then all parts of the object have that property E.g. : Color of a human and no. of fingers of human

- Event and Process : An event is anything with temporal extent e.g. Walking. If every temporal slice of an event is essentially the same as the entire event then that event is a process.
- E.g.: Walking is a process and walking two miles is not.
- Slots : It's a subclass of Intangible.
    - Book-keepingSlots record information when frame was created and by whom.
    - DefiningSlots refer to the properties of object
    - QuantitativeSlots refer to the scalar range of values. E.g.: height
- Time : Events can have temporal properties such as duration and startsBefore.
- Agent : Subset of CompositeObject is Agent, the collection of intelligent beings.

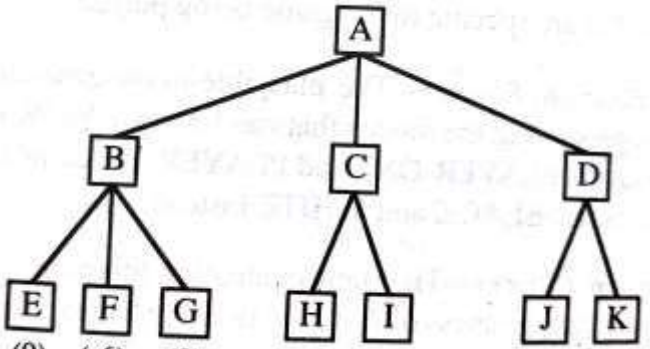
8.b. Explain min max search procedure with algorithm [Jan'18, Jul'18, Jan'19]

- Depth first and depth limited search.
- Idea is to start at current position and use plausible move generator to generate the set of possible successor positions.
- Now apply static evaluation function to choose the best position to move such that we maximize our possibility of win thereby minimizing the opponents' chance of win

One ply search

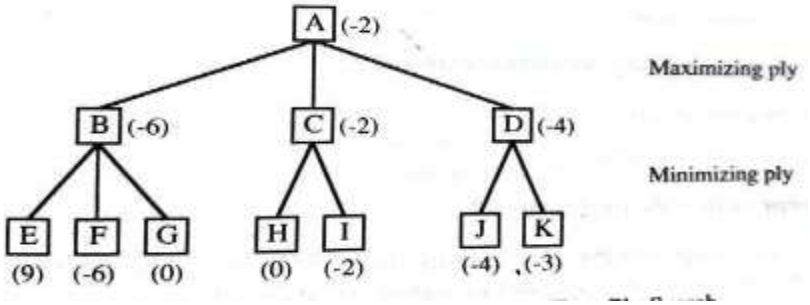


Two ply search:



Backing up the values

- Player 1: Maximize the chance
- Player 2: Minimize the chance



Algorithm :

1. If DEEP-ENOUGH(*Position, Depth*), then return the structure
   VALUE = STATIC(*Position, Player*);
   PATH = nil
   This indicates that there is no path from this node and that its value is that determined by the static evaluation function.
2. Otherwise, generate one more ply of the tree by calling the function MOVE-GEN(Position Player) and setting SUCCESSORS to the list it returns.
3. If SUCCESSORS is empty, then there are no moves to be made, so return the same structure that would have been returned if DEEP-ENOUGH had returned true.
4. If SUCCESSORS is not empty, then examine each element in turn and keep track of the best one. This is done as follows.
   Initialize BEST-SCORE to the minimum value that STATIC can return. It will be updated to reflect the best score that can be achieved by an element of SUCCESSORS.
   For each element SUCC of SUCCESSORS, do the following:
   (a) Set RESULT-SUCC to
   MINIMAX(SUCC, *Depth* + 1, OPPOSITE(*Player*))
   This recursive call to MINIMAX will actually carry out the exploration of SUCC.
   (b) Set NEW-VALUE to — VALUE(RESULT-SUCC). This will cause it to reflect the merits of the position from the opposite perspective from that of the next lower level.
   (c) If NEW-VALUE > BEST-SCORE, then we have found a successor that is better than any that have been examined so far. Record this by doing the following:
   i. Set BEST-SCORE to NEW-VALUE.
   ii. The best known path is now from CURRENT to SUCC and then on to the appropriate path down from SUCC as determined by the recursive call to MINIMAX. So set BEST-PATH to the result of attaching SUCC

MODULE 5:

9.a. Define learning and give the differences between neural net learning and genetic learning.

- A process by which one entity acquires knowledge
- Do machines learn?

- 4 general category
  - Learning by practicing
    - E.g.: Playing tennis, Riding etc.,
  - Learning by experience
    - E.g.: Knowledge acquisition based on the previous tasks
  - Learning by taking advice
  - Learning from examples
- A neural network is a system of hardware and/or software patterned after the operation of neurons in the human brain
- Efforts to identify the behavior of neurons in human as well as animal.
- E.g.: Signal Processing, Pattern recognition
- The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection

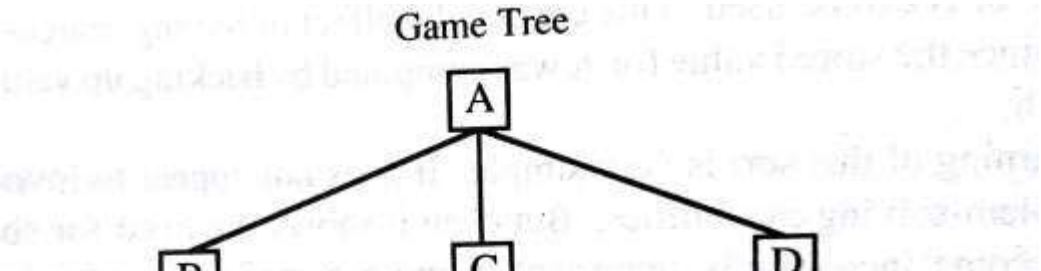
9.b. Write note on knowledge acquisition

- Initially knowledge engineer would collect knowledge which will be translated into rules.
- This way was found time consuming and expensive
- Later an idea to create an automatic knowledge acquisition system evolved after which a few semi-automatic things were found which interacts with domain experts to gather knowledge. E.g.: MOLE
- Activities taken care by this program are,
- Entering knowledge
- Maintaining knowledge base consistency
- Ensuring knowledge base completeness

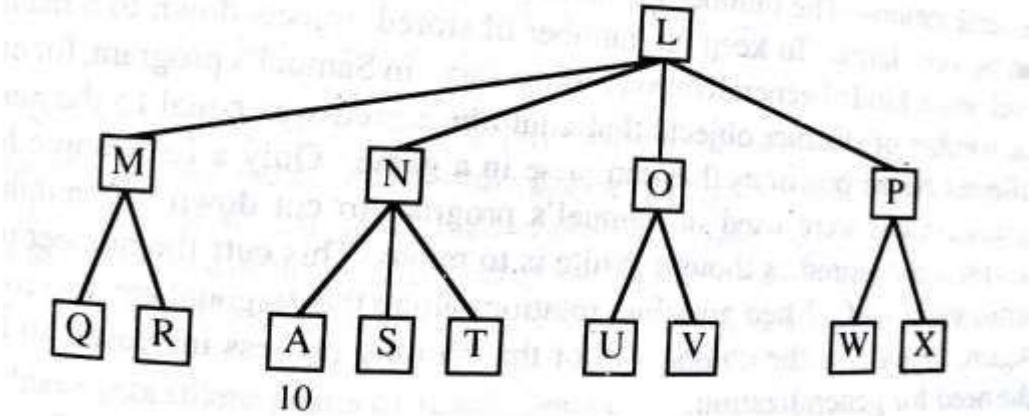
9.c. Explain rote learning.

Rote learning:

- Learn from the past things
- E.g.: In case of some successive dependent computations store we can store the computed values so that we do not have to re-compute them later.
- Avoids expensive re-computation
- E.g.: Caching
- Value of A is computed as 10



- Now, A need not be recomputed again.



10

- Things we should be cautious about caching
- Organized storage of information : The data computed should be stored in manner like it can be retrieved easily for future references
- Generalization: Number of distinct objects that might potentially be stored shall be very large. To keep the number of stored objects down to a manageable level, generalization should be done.

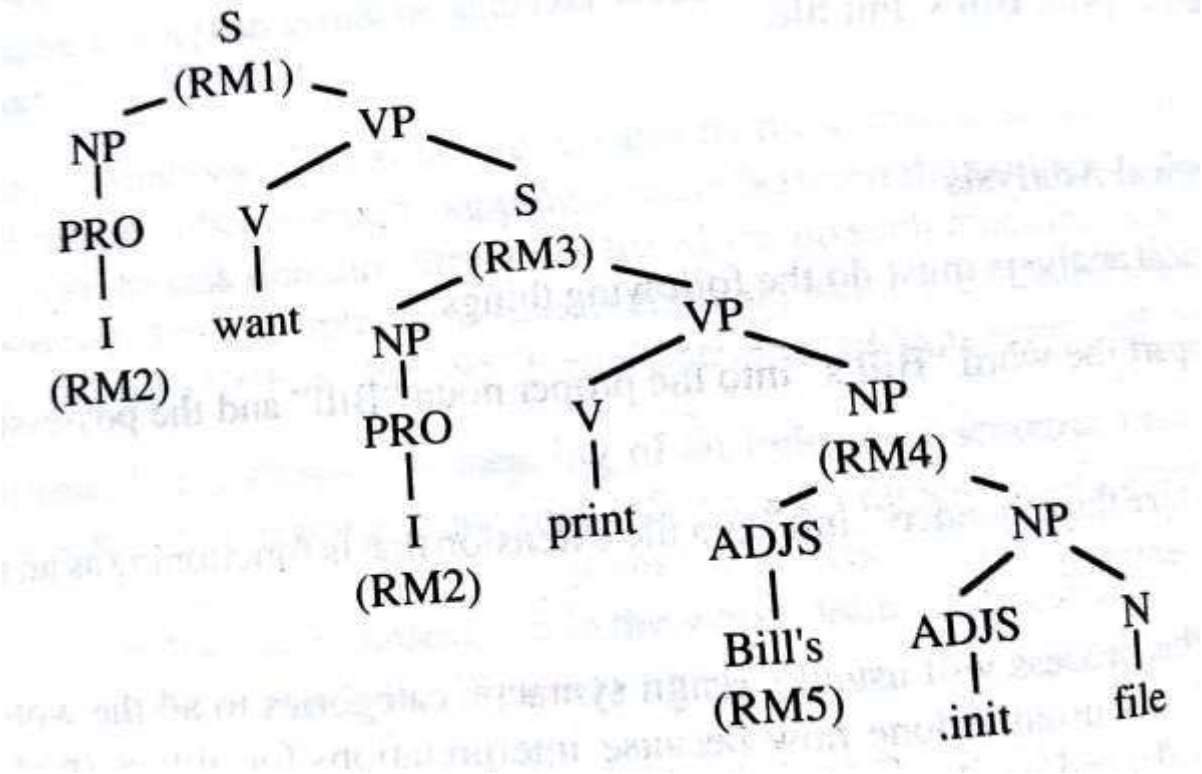
10.a. Explain steps involved in Natural Language Processing

- Morphological Analysis – Individual words are analyzed into their components, and non-word tokens, such as punctuation are separated from the words
    i. An example: I want to print Bill's .init file.
    ii. Bill's --> Bill (the proper noun) + 's (the possessive suffix)
    iii. Recognize the sequence ".init" as a file extension.
    iv. In addition, this process will usually assign syntactic categories to all the words in the sentence. E.g. verbs, Nouns, Singular, plural etc.,

- Syntactic Analysis – Linear sequences of words are transformed into structures that somehow the words relate to each other
    i. Syntactic analysis must exploit the results of morphological analysis to build a structural description of the sentence.
    ii. I.e. arrange words in a proper manner such that proper relationship is established between words
    iii. Parse tree is generated representing the relationship between words
    iv. Reference markers: Set of entities represented in parenthesis.
- Semantic analysis – The structures created by the syntactic analyzer are assigned meanings. In other words mapping is made between the syntactic structures and objects in the task domain. Things having no mapping is rejected.

E.g.: "I want to print Bill's .init file."

Category	Examples
Determiner	the, this, a
Adjective	big, high
Adverb	slowly
Noun	file, Sarah, book
Auxiliary verb	will, have
Verb	print, give
Preposition	to, in
Quantifier	all, every
Complementizer	that, which
Pronoun	she, him, their



- **Semantic Analysis** – does two things
- It must map individual words into appropriate objects in the knowledge base or database.
- It must create the correct structures to correspond to the way the meaning of the individual words combine each other

- Discourse integration – The meaning of an individual sentence may depend on the sentence that precede it and may influence the meanings of the sentence that follow it.
    i. E.g.: John went to park with his mom where he saw a toy. As John wanted it, John's mom got it for him.
- Pragmatic analysis – the structure representing what was said is reinterpreted to determine what was actually meant.

10.b. Explain spell checking techniques. [Jan'18, Jan'19]

- Spell checking is broadly classified into three categories,

Non word error detection

- This process's involves the detection of misspelled words or non-words.
- E.g: the word soper is a non-word; it's correct form being super or sober.
- N gram analysis and dictionary lookup are used to detect the errors
- They use the probabilities of occurrence of N grams in a large corpus of text to decide on the error in the word
- Those strings that contain highly infrequent sequences are treated as cases of spelling errors.
- These techniques are used in text recognition techniques tha are processed by Optical Character Recognition system
- The OCR uses features of each character such as curves and the loops made by the, to identify the character.
- N gram method uses tables to predict whether a sequence of characters does exist within corpora and then flags an error
- Dictionary lookup involves te use of an efficient dictionary lookup coupled with pattern matching algorithms, dictionary partitioning schemes and morphological processing methods

Isolated word error correction

- This focuses on the correction of an isolated non word by finding its nearest and meaningful word and makes an attempt to rectify the error.
- It thus transforms the word soper into super by some means but without looking at the context.
- This correction is usually performed context independent suggestion generation exercise which includes minimum edit distance techniques, similarity key techniques, and rule based methods, N gram, probabilistic and neural network based techniques.

- This works on error detection, candidate generation and ranking of correct candidates.

Context dependent error detection and correction

- This processes try in addition to detect errors try to find whether the corrected word fits into the context of the sentence
- Its complex to implement and requires more resources
- E.g.:
  Peace comes from within and piece comes from within are both correct.
- This involves correction of real word errors or those that result in another valid.