CBCS SCHEME

USN □□□□□□□□□□

18CS753

Seventh Semester B.E. Degree Examination, Feb./Mar.2022
Introduction to Artificial Intelligence

Time: 3 hrs. Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a. What do you mean by an Artificial Intelligence techniques? **(05 Marks)**

b. Demonstrate the turing test with example. **(07 Marks)**

c. Write short notes on problem area in Artificial Intelligence. **(08 Marks)**

OR

2 a. Solve the cryptoarithemetic problem given below with proper steps.

```
 POINT
+ZERO
-----------
ENERGY
```

(05 Marks)

b. Explain the four categories of production systems with an example for each. **(07 Marks)**

c. Explain the decomposable and non-decomposable problem with an example. **(08 Marks)**

Module-2

3 a. Differentiate between forward reasoning and backward reasoning. **(05 Marks)**

b. Explain the convert to clause form algorithm with an example. **(07 Marks)**

c. Consider the following sentences:

- John likes all kinds of food.
- Apples are food.
- Chicken is food.
- Anything any one eats and is not killed by is food.
- Bill eats peanuts and still alive.
- Sue eats everything bill eats

(i) Convert these into predicate logic.

(ii) Prove that "John likes peanuts" using back chaining. **(08 Marks)**

OR

4 a. Write the four properties of knowledge representation system. **(05 Marks)**

b. Discuss various issues in knowledge representation. **(07 Marks)**

c. Consider the following facts:

- Marcus was a man.
- Marcus was a Pompeian.
- All pompeians were Romans.
- Caeser was a ruler.
- All romans were either loyal to caeser or hated him.
- Everyone loyal to someone.
- People only try to assassinate rulers they are not loyal to.
- Marcus try to assassinate ceaser.

(i) Prove that marcus is not loyal to ceaser by backward substitution.

(ii) Represent the above statements using instance relationship and ISA relationship. **(08 Marks)**

Module-3

5 a. What is Non-Monotonic reasoning? (05 Marks)

b. Explain default reasoning and minimalist reasoning. (10 Marks)

c. Explain closed world assumptions. (05 Marks)

OR

6 a. Define the Bayes theorem. (05 Marks)

b. Explain JTMS and dependency directed backtracking. (10 Marks)

c. Explain Bayesian networks with a diagram. (05 Marks)

Module-4

7 a. Explain the steps involved in natural language processing. (08 Marks)

b. Explain the minmax search with an example. (08 Marks)

c. Write an interactive deepening A* algorithm. (04 Marks)

OR

8 a. What is conceptual parsing? (05 Marks)

b. Write a short notes on Alpha-beta cutoffs. (08 Marks)

c. Derive the parse tree for the following sentence making use of appropriate grammar: "Bill printed the file". (07 Marks)

Module-5

9 a. What is the role of expert system and knowledge acquisition? (05 Marks)

b. Write a note on explanation-based learning and explain rote learning with an example. (10 Marks)

c. Discuss the concept of learning from taking advice. (05 Marks)

OR

10 a. What do you mean by expert shell? (05 Marks)

b. Explain Winston's learning program in detail. (10 Marks)

c. Discuss the concept of learning by chunking. (05 Marks)

* * * * *

VTU QUESTION PAPER SOLUTION-MAR-22

1 a. WHAT DO YOU MEAN BY ARTIFICIAL INTELLIGENCE TECHNIQUE

- Artificial intelligence is **the simulation of human intelligence processes by machines**, especially computer systems.
- Its the **theory and development of computer systems** able to perform tasks normally requiring human intelligence.

What **kinds of techniques** will be useful for solving AI problems?

- AI problems span a very broad spectrum & intelligence that requires knowledge.
- Knowledge possesses properties like :

✓ It is voluminous
✓ It is hard to characterize accurately
✓ It is constantly changing
✓ It differs from data by being **organized in a way** that corresponds to the ways **it will be used**

- An AI technique is a ***method.***
- *It exploits knowledge* that should be represented in such a way that:

✓ The knowledge captures **generalizations.**
✓ It can be **understood** by people who must provide it
✓ It can easily be **modified** to correct errors
✓ It can be used to help **overcome** its own sheer **bulk**

- It is possible to solve AI problems without using AI techniques & its possible to apply AI techniques to solution of non-AI problems.

1 b. DEMONSTRATE THE TURING TEST WITH EXAMPLE

- The Turing test, originally called the imitation game by Alan Turing in 1950, is a test of a machine's ability to exhibit intelligent behaviour equivalent to, or indistinguishable from, that of a human.
- Turing proposed that a human evaluator would judge natural language conversations between a human and a machine designed to generate human-like responses.
- The evaluator would be aware that one of the two partners in conversation is a machine, and all participants would be separated from one another.
- The conversation would be limited to a text-only channel such as a computer keyboard and screen so the result would not depend on the machine's ability to render words as speech.
- If the evaluator cannot reliably tell the machine from the human, the machine is said to have passed the test. The test results do not depend on the machine's ability to give correct answers to questions, only how closely its answers resemble those a human would give.

1 c. WRITE SHORT NOTES ON PROBLEM AREA IN AI.

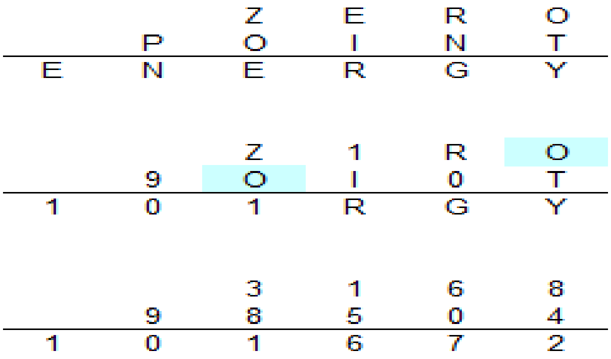
One needs to do 4 things **to build a system** to solve a particular problem:

1. *Define* the problem precisely.
2. Analyze the problem.
3. *Represent task knowledge* that is necessary to solve the problem.
4. *Choose* the best problem-solving techniques & *apply* it to the particular problem.

2 a. SOLVE CRYPTOARITHEMETIC PROBLEM GIVEN BELOW WITH PROPER STEPS

```
    POINT
+   ZERO
 --------------
   ENERGY
```

SOLUTION:



		Z	E	R	O
	P	O	I	N	T
E	N	E	R	G	Y

		Z	1	R	O
	9	O	I	0	T
1	0	1	R	G	Y

		3	1	6	8
	9	8	5	0	4
1	0	1	6	7	2

2 b. EXPLAIN FOUR CATEGORIES OF PRODUCTION SYSTEMS WITH AN EXAMPLE FOR EACH

- The process of search is *fundamental* to problem solving process.
- Search is a general mechanism that can be used when *no more direct method* is known
- It provides the framework into which more direct methods for solving subparts of a problem can be embedded.
- Search forms core of many AI problems *so AI problems* need to be *structured for performing search*

A production system consists of:

- A set of rules.
- Databases : contains appropriate information for a particular task.
- A control strategy : it specifies order in which rules will be compared to database.
- A rule applier.

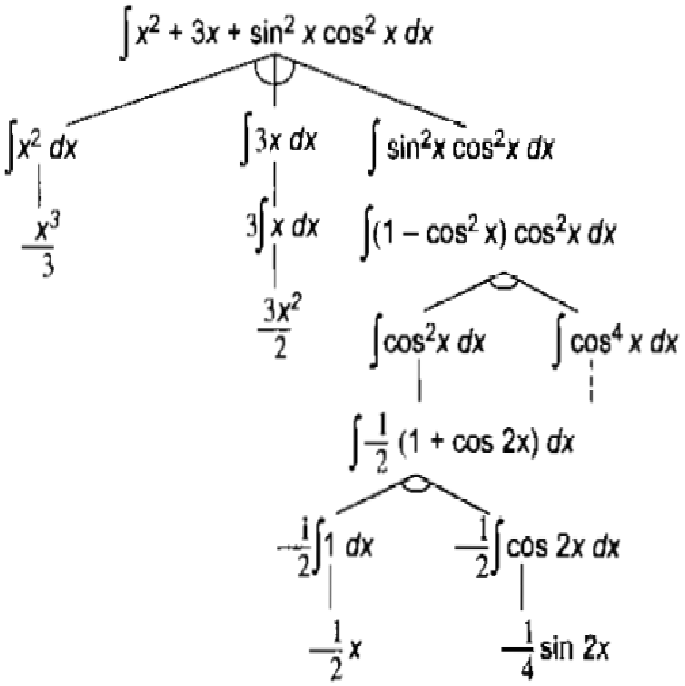
2 c. EXPLAIN DECOMPOSABLE & NON-DECOMPOSABLE PROBLEM WITH EXAMPLE.

To choose appropriate method for a particular problem its necessary to *analyze* the problem in *various dimensions* :

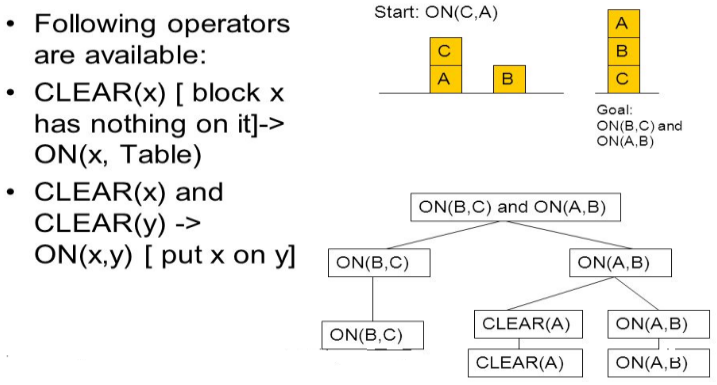
- Is the problem decomposable into a set of independent smaller sub problems?
- Can solution steps be ignored or be undo if they are unwise?
- Is the problems universe predictable?
- Is a good solution to the problem obvious with comparison?
- Is the desired solution a state of the world or a path to a state?
- Is a large amount of knowledge required to solve the problem?
- Can a computer return solution for a given problem or is human intervention required?

Suppose we want to solve the problem of computing an expression:

- Solve by breaking it into 3 sub parts- problem decomposition


$\int x^2 + 3x + \sin^2 x \cos^2 x \, dx$
$\int x^2 \, dx$
$\frac{x^3}{3}$
$\int 3x \, dx$
$3\int x \, dx$
$\frac{3x^2}{2}$
$\int \sin^2 x \cos^2 x \, dx$
$\int (1 - \cos^2 x) \cos^2 x \, dx$
$\int \cos^2 x \, dx$
$\int \cos^4 x \, dx$
$\int \frac{1}{2} (1 + \cos 2x) \, dx$
$\frac{1}{2}\int 1 \, dx$
$\frac{1}{2}\int \cos 2x \, dx$
$\frac{1}{2} x$
$\frac{1}{4} \sin 2x$

- Now consider a problem of blocks world



- Following operators are available:
- CLEAR(x) [block x has nothing on it]-> ON(x, Table)
- CLEAR(x) and CLEAR(y) -> ON(x,y) [put x on y]

3 a. DIFFERENTIATE BETWEEN FORWARD AND BACKWARD REASONONG

- The objective of search procedure is to discover a path from an initial configuration to a goal state.
- PROLOG only searches from goal state.
- There are 2 directions in which such a search could proceed:

1. Forward: from the start states
2. Backward: from the goal states.

- Consider the problem of solving a particular instance of 8-puzzle. The rules to be used for solving the puzzle can be written as shown in figure.

Reason forward from initial states:

1. Build a tree of move sequences by starting with the initial configurations at root of the tree.
2. Generate next level using rules
- whose left sides match the root node and
- use their right sides to create the new configurations.
3. Generate next level by taking each node generated at the previous level and
- apply to it all rules whose left sides match it.

4. Continue until a configuration that matches the goal state is generated.

Reason backward from goal states: (goal directed reasoning)

1. Build a tree of move sequences by starting with the goal configurations at root of the tree.
2. Generate next level using rules
- whose right sides match the root node and
- use the left sides to generate the nodes at this 2nd level of tree.
3. Generate next level by taking each node generated at the previous level and
- finding all rules whose right sides match it.
- Then use the corresponding left side to generate new nodes.
4. Continue until a node that matches the initial state is generated.

3 b. EXPLAIN CONVERT TO CLAUSE FORM ALGORITHM WITH EXAMPLE

1. Eliminate □(material implication) using the fact that a□ b is equivalent to ¬aVb.

   Performing this transformation on wff(well formed formula's) given above yields:
   Vx: ¬[Roman (x)Λ know(x, Marcus)]V [hate(x,Caesar)V (Vy:¬ (∃z: hate(y,z)) Vthinkcrazy(x,y))]

2. Reduce the scope of each ¬ to a single term, using the fact that ¬(¬p)=p

   Vx: ¬[Roman (x)V ¬know(x, Marcus)]V [hate(x,Caesar)V (Vy: Vz:¬ hate(y,z)) V thinkcrazy(x,y))]

3. Standardize variables so that each quantifier binds a unique variable.

   This process cannot affect the truth value of wff.
   Vx: P(x) V Vx: Q(x) would be converted to ***Vx: P(x) V Vy: Q(y)***

4. Move all quantifiers to left of the formula without changing their relative order.

   Performing this on step 2:
   Vx: Vy: Vz: [*¬Roman (x)V ¬know(x, Marcus)]V [hate(x,Caesar)V (¬ hate(y,z)V thinkcrazy (x,y)]*

5. Eliminate existential quantifiers.

   ∃y: President(y) can be transformed into formula President(S1)
   S1- function with no arguments, produces values to satisfy President.
   Vx:∃y: father-of(y,x)
   - y(father) here depends on x(son).

   Vx: father-of(S2(x),x))□ *Skolem functions*

6. Drop the prefix. Step 4 appears as

   ¬[Roman (x) V ¬know(x, Marcus)] V [hate(x,Caesar)V (¬ hate(y,z) V thinkcrazy(x,y))]

7. Convert Matrix into conjunction of Disjuncts.(in our example no ANDs only ORs)

   ¬Roman (x) V ¬know(x, Marcus) V hate(x,Caesar)V ¬ hate(y,z) V thinkcrazy(x,y)

8. Create separate clause corresponding to each conjunct.

9. Standardize apart the variables in the set of clauses generated in step 8.

- No 2 variables should have same name.
- for this, depend on fact that

(Vx: P(x) Λ Q(x)) = Vx: P(x) Λ Vx: Q(x)

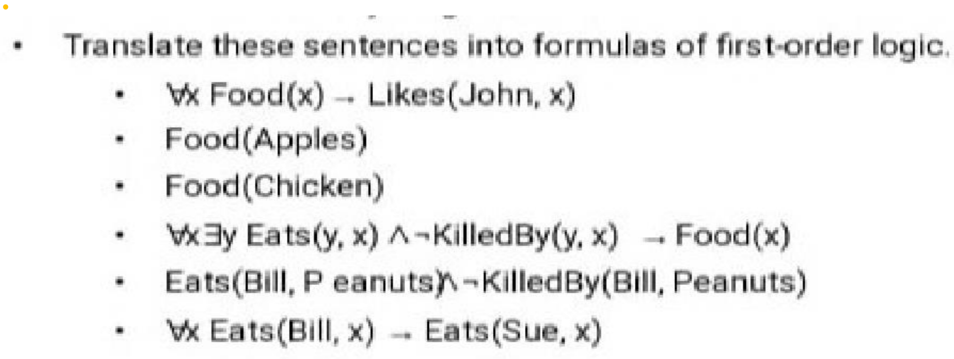
- After applying this entire procedure to set of wff's , we will have clauses.
- These clauses are used by resolution procedure to generate proofs.

3 c. CONSIDER THE FOLLOWING SENTENCES:

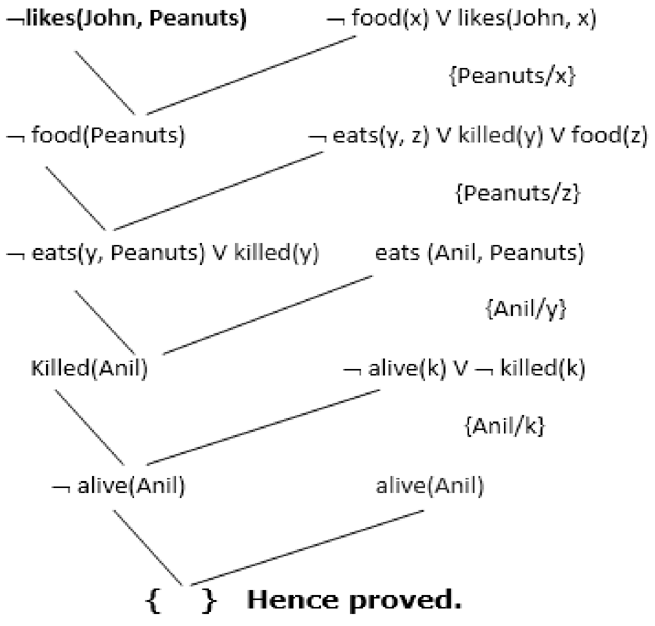
- *JOHN LIKES ALL KINDS OF FOOD*
- *APPLES ARE FOOD*
- *CHICKEN IS FOOD*
- *ANYTHING ANY ONE EATS AND IS NOT KILLED BY IS FOOD*
- *BILL EATS PEANUTS AND STILL ALIVE*
- *SUE EATS EVERYTHING BILL CATS*

i. *CONVERT TO PREDICATE LOGIC*
ii. *PROVE THAT "JOHN LIKES PEANUTS" USING BACK CHAINING*

SOLUTION:



- Translate these sentences into formulas of first-order logic.
  - $\forall x$ Food(x) → Likes(John, x)
  - Food(Apples)
  - Food(Chicken)
  - $\forall x \exists y$ Eats(y, x) ∧ ¬KilledBy(y, x) → Food(x)
  - Eats(Bill, P eanuts)∧¬KilledBy(Bill, Peanuts)
  - $\forall x$ Eats(Bill, x) → Eats(Sue, x)



4 a. WRITE THE FOUR PROPERTIES OF KNOWLEDGE REPRESENTATION SYSTEMS

✓ A good system for the representation of knowledge should have four properties:

1. **Representational Adequacy:** ability to represent all kinds of knowledge needed in that domain.

2. **Inferential Adequacy:** ability to manipulate the representational structure to drive new structures.
3. **Inferential Efficiency:** ability to incorporate additional information into knowledge structure.
4. **Acquisitional Efficiency:** ability to acquire new information easily.

- Unfortunately, no single system that optimizes all these capabilities is yet found.

4 b. DISCUSS VARIOUS ISSUES IN KNOWLEDGE REPRESENTATION

✓ Several issues wrt mechanisms that have been used to represent various kinds of real-world knowledge, as follows:

1. Are attributes of objects so basic that they occur in almost every problem domain ?
2. Are there any important relationships that exist among attributes of objects?
3. At what level should knowledge be represented?
4. How should sets of objects be represented ?
5. Given a large amount of knowledge stored in a database, how can relevant parts be accessed when they are needed?

4 c. CONSIDER THE FOLLOWING FACTS:

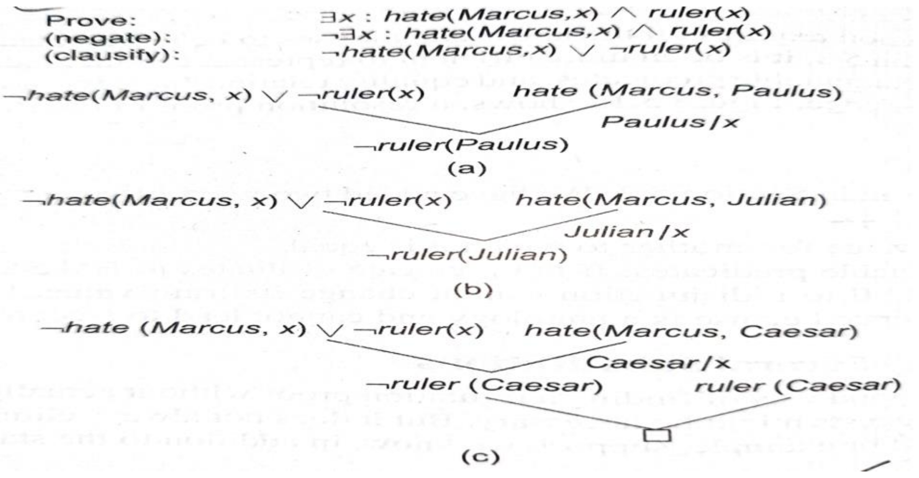
- ***Marcus was a man.***
- ***Marcus was a Pompeian.***
- ***All Pompeian were Romans.***
- ***Caesar was a ruler.***
- ***All Romans were either loyal to Caesar or hated him.***
- ***Everyone is loyal to someone.***
- ***People only try to assassinate rulers they are not loyal to.***
- ***Marcus tried to assassinate Caesar.***

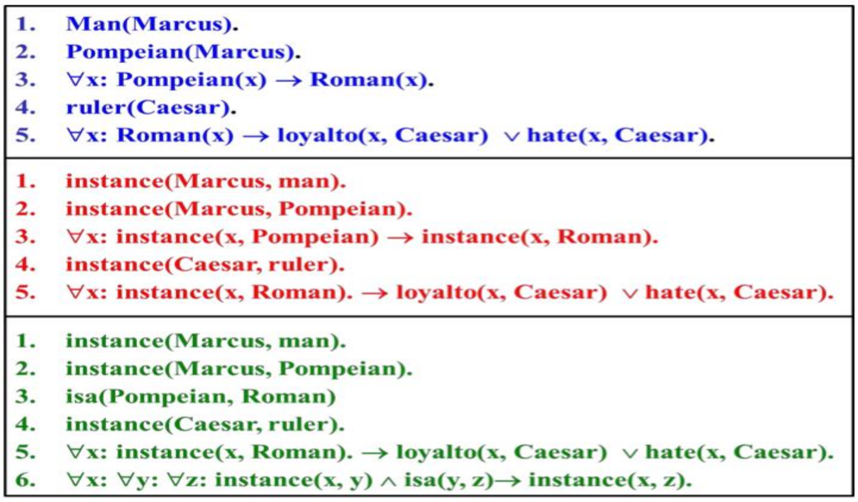
i. PROVE THAT MARCUS IS NOT LOYAL TO CAESAR BY BACKWARD SUBSTITUTION

ii. REPRESENT THE ABOVE STATEMENTS USING INSTANCE RELATIONSHIP AND ISA RELATIONSHIP

SOLUTION:

- Resolution provides a very good way of finding a refutation proof without actually trying all the substitution.
- If we want to prove Marcus hates some ruler, we could try each substitution shown in figures next





1. Man(Marcus).
2. Pompeian(Marcus).
3. $\forall x: \text{Pompeian}(x) \rightarrow \text{Roman}(x).$
4. ruler(Caesar).
5. $\forall x: \text{Roman}(x) \rightarrow \text{loyalto}(x, \text{Caesar}) \vee \text{hate}(x, \text{Caesar}).$

1. instance(Marcus, man).
2. instance(Marcus, Pompeian).
3. $\forall x: \text{instance}(x, \text{Pompeian}) \rightarrow \text{instance}(x, \text{Roman}).$
4. instance(Caesar, ruler).
5. $\forall x: \text{instance}(x, \text{Roman}). \rightarrow \text{loyalto}(x, \text{Caesar}) \vee \text{hate}(x, \text{Caesar}).$

1. instance(Marcus, man).
2. instance(Marcus, Pompeian).
3. isa(Pompeian, Roman)
4. instance(Caesar, ruler).
5. $\forall x: \text{instance}(x, \text{Roman}). \rightarrow \text{loyalto}(x, \text{Caesar}) \vee \text{hate}(x, \text{Caesar}).$
6. $\forall x: \forall y: \forall z: \text{instance}(x, y) \wedge \text{isa}(y, z) \rightarrow \text{instance}(x, z).$

5 a. WHAT IS NON-MONOTONIC REASONING?

Nonmonotonic Reasoning:

- here axioms or rules of inference are extended to make it possible to reason with incomplete information.
- These systems assume that at any given time , a statement is either TRUE or FALSE or neither TRUE or FALSE.

5 b. EXPLAIN DEFAULT REASONING AND MINIMALIST REASONING

Default Reasoning:

- Nonmonotonic reasoning is used to perform default reasoning.
- Two approaches for default reasoning are:

1. Nonmonotonic Logic
2. Default Logic.

Two common kinds of nonmonotonic reasoning that can be defined in logics are:

1. Abduction
2. Inheritance

Minimalist Reasoning

- A model is minimal if there are no other models in which fewer things are true.

5 c. EXPLAIN CLOSED WORLD ASSUMPTIONS.

- CWA says that the only object that satisfy any predicate P are those that must.
- For Ex: A personnel database can safely be assumed to list all of the company’s employees.
  If someone asks whether Smith works for the company,
  we should reply “no” unless he is explicitly listed as an employee.
- Although CWA is powerful, it can fail to produce an appropriate answer for 2 reasons:

1. Its assumptions are not always true in the world.
2. It is purely syntactic reasoning process.

- Consider a knowledge base that consists of just a single statement:

A(Joe) ∨ B(Joe)

- CWA concludes both? A(Joe) and ?B(Joe), since neither A nor B must necessarily be true of Joe.
- The resulting extended knowledge base

A(Joe) *V* B(Joe)
¬A(Joe)
¬B(Joe)

- Is inconsistent.
- Suppose we define a predicate Single and create following base:

Single(John)
Single(Mary)

- If asked about Jane, CWA will yield the answer ¬ Single(Jane)
- Suppose we had chosen instead to use predicate Married rather than single.
- Then corresponding knowledge base would be

¬ Married(John)
¬ Married(Mary)

- If we now ask about Jane , the CWA will yield the result **¬ Married(Jane)**

6 a. DEFINE BAYES THEOREM.

- Bayes' theorem states that

$$P(H_i\backslash E) = \frac{P(E \mid H_i)\cdot P(H_i)}{\sum_{n=1}^{k} P(E \mid H_n)\cdot P(H_n)}$$

P(Hi\E) = probability that hypothesis Hi is true given evidence E.
P(E\Hi) = probability that we observe evidence E given hypothesis ***i*** is true.
P(Hi) =a priori probability that hypothesis ***i*** is true in absence of specific evidence.
k= number of possible hypotheses.

- Example :

Examine geological evidence at a particular location to determine whether to dig to find desired minerals.

- If we know the prior probabilities of finding various minerals the Bayes formula can be used to compute how likely various minerals are present.

6 b. EXPLAIN JTMS AND DEPENDENCY DIRECTED BACKTRACKING

Dependency Directed Backtracking

- If we use Depth first approach for nonmonotonic reasoning, we should know fact F, derived from assumption A.
- To overcome Chronological Backtracking issues Dependency-Directed Backtracking is used.
- Ex: build a program that generates a solution to a problem of finding time at which three busy people can all attend meeting.
- This kind of situation can be handled by a straightforward tree search with chronological backtracking.
- All assumptions and inferences drawn from them, are recorded at the search node.
- When a node is determined to represent a contradiction, simply backtrack to next node from which there are unexplored paths.
- The assumptions and their inferences will automatically disappear.
- The drawback to this approach is illustrated in figure,

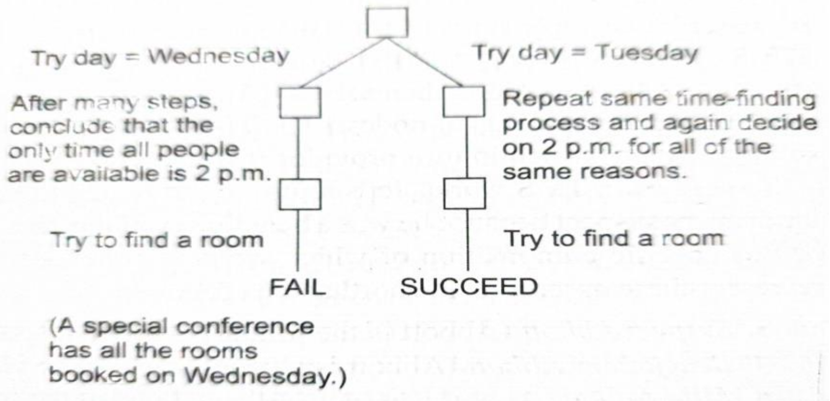

Try day = Wednesday
Try day = Tuesday
After many steps, conclude that the only time all people are available is 2 p.m.
Repeat same time-finding process and again decide on 2 p.m. for all of the same reasons.
Try to find a room
Try to find a room
FAIL
SUCCEED
(A special conference has all the rooms booked on Wednesday.)

Fig. 7.4 *Nondependency-Directed Backtracking*

6 c. EXPLAIN BAYESIAN NETWORKS WITH A DIAGRAM

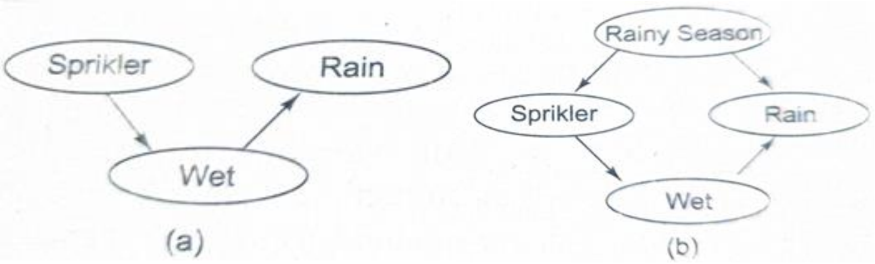
- Bayesian network preserves the formalism and rely on modularity of the world we are trying to model.
- Figure shows flow of constraints in MYCIN- style rules.

1. Constraints flowed incorrectly here. &
2. Distinction couldn't be made.

- 2 different ways propositions can influence likelihood of their symptoms.

1. First is that, causes influence the likelihood of their symptoms
2. Second is that, observing a symptom affects the likelihood of all of its possible causes.

- Bayesian network structure makes clear distinction between these 2 kind of influence.



(a) (b)

- Construct Directed Acyclic Graph(DAG) that represents causality relationships among variables.
- The variables in such graph may be propositional or they may be variables that take on values of some other type.
- DAG illustrates causality relationships that occur among the nodes it contains.
- In order to use it as a basis for probabilistic reasoning we need to know for each value of parent node what evidence is provided about the values that the child node can take on.

7 a. EXPLAIN STEPS INVOLVED IN NLP

- ➢ Natural Language process is broken down into the following pieces:

1. **Morphological Analysis-** individual words are analyzed into their components & nonword tokens, such as punctuation, are separated from the words.
2. **Syntactic Analysis-** Linear sequences of words are transformed into structures that show how the words relate to each other.

- Word sequences may be rejected if they violate the language's rules.
- For example: the sentence "the boy go the to store"

3. **Semantic Analysis- the structures created by syntactic analyzer are assigned meanings.**

- A mapping is made between syntactic structures & objects in task domain.

- Structures for which no such mapping is possible may be rejected.
- The sentence “colorless green ideas sleep furiously”, would be rejected as semantically anomalous.

4. **Discourse Integration-** meaning of an individual sentence may depend on the sentences that precede it & may influence the meanings of the sentences that follow it.
5. **Pragmatic Analysis-** the structure representing what was said is reinterpreted to determine what was actually meant.

- For example- the sentence “Do you know what time it is?” should be interpreted as a request to be told the time.
- The phrases are sometimes performed in sequence & they are sometimes performed all at once.
- It is often useful to separate these 5 processing phases to some extent, they can all interact in a variety of ways making a complete separation impossible.

7 b. EXPLAIN MINMAX SEARCH WITH AN EXAMPLE

- The minimax search procedure is a depth-first, depth-limited search procedure.
- It starts at current position and generate the set of possible successor positions.
- Then apply static evaluation function to those positions & simply choose best one.
- After doing so, we can back that value up to the starting position to represent our evaluation of it.
- The starting position is exactly as good for us as the position generated by best move we can make next.
- So the goal is to maximize the value of the static evaluation function of the next board position.
- Assume a static evaluation function that returns values ranging from -10 to 10. 10 indicating a win for us, -10 a win for the opponent, & 0 an even match.
- Choose to move to B. Backing B’s value up to A, we can conclude that A’s value is 8, since we know we can move to a position with a value of 8.

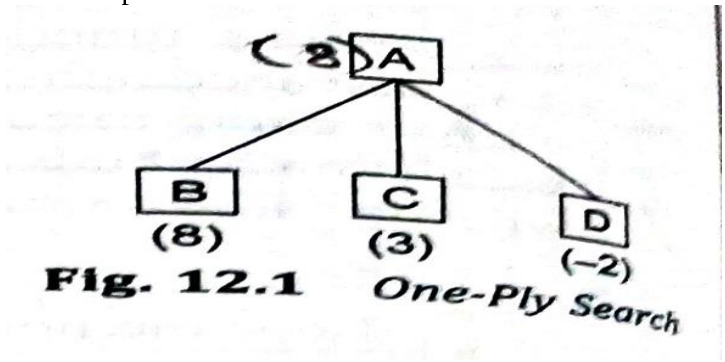
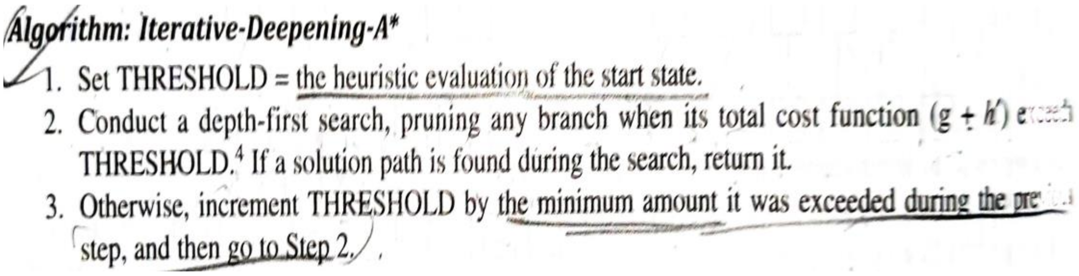


Fig. 12.1 One-Ply Search

7 c. WRITE AN INTERACTIVE DEEPENING A ALGORITHM.*

- The major practical difficulty with A* is the large amount of memory it requires to maintain the search node lists.



Algorithm: Iterative-Deepening-A*

1. Set THRESHOLD = the heuristic evaluation of the start state.
2. Conduct a depth-first search, pruning any branch when its total cost function ($g + h'$) exceed THRESHOLD.[4] If a solution path is found during the search, return it.
3. Otherwise, increment THRESHOLD by the minimum amount it was exceeded during the previous step, and then go to Step 2.

- Iterative- Deepening A* is guaranteed to find an optimal solution, provided that h’ is an admissible heuristic.
- IDA* is efficient wrt space, since Depth-first search is used.
- Its first heuristic search algorithm to find optimal solution paths for 15-puzzle within reasonable time & space.

8 a. WHAT IS CONCEPTUAL PARSING?

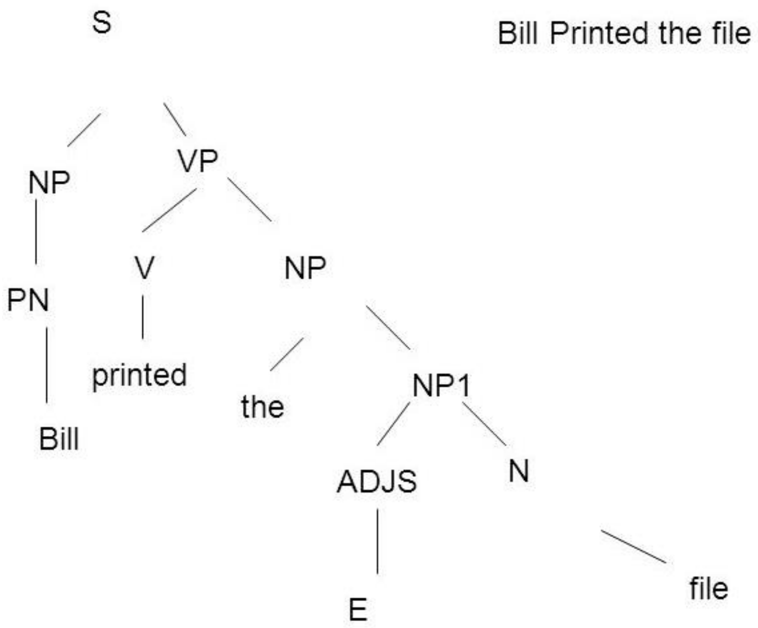
- A natural language parser is a program that figures out which group of words go together (as “phrases”) and which words are the subject or object of a verb.
- The NLP parser separates a series of text into smaller pieces based on the grammar rules.

8 b. WRITE SHORT NOTES ON ALPHA-BETA CUTOFFS.

- ALPHA-BETA cutoff is a method for reducing the number of nodes explored in the Minimax strategy.
- For the nodes it explores it computes, in addition to the score, an alpha value and a beta value.
- ALPHA value of a node Initially it is the score of that node, if the node is a leaf, otherwise it is -infinity.

8 c. DERIVE THE PARSE TREE FOR THE FOLLOWING SENTENCE MAKING USE OF APPROPRIATE GRAMMAR: “BILL PRINTED THE FILE”

SOLUTION:



9 a. WHAT IS THE ROLE OF EXPERT SYSTEM AND KNOWLEDGE ACQUISITION.

- Expert systems solve problems that are normally solved by human experts.
- One way to look at expert systems is that they represent applied AI in a very broad sense.
- Expert Systems are complex AI programs.
- The most widely used way of representing domain knowledge in expert systems is as a set of production rules.

Knowledge Acquisition

How are expert systems built?

- Typically, a knowledge engineer interviews a domain expert to Elucidate expert knowledge, which is then translated into rules.

- After the initial system is built, it must be iteratively refined until it approximates expert-level performance.
- There are many programs that interact with domain experts to extract expert knowledge efficiently.
- These programs provide support for the following activities:

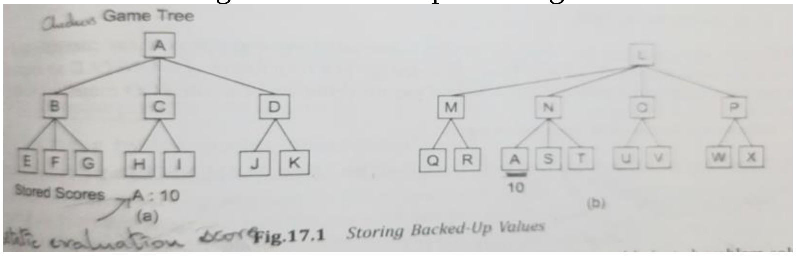
1. Entering knowledge.
2. Maintaining knowledge base consistency.
3. Ensuring knowledge base completeness.

- The most useful knowledge acquisition programs are those that are restricted to a particular problem-solving paradigm. Ex: diagnosis or design.
- It is important to be able to enumerate the roles that knowledge can play in problem-solving process.
- For example- if the paradigm is diagnosis, then the program can structure its knowledge base around symptoms, hypotheses & causes.
- 2 knowledge acquisition systems are as follows:
    1. MOLE and
    2. SALT

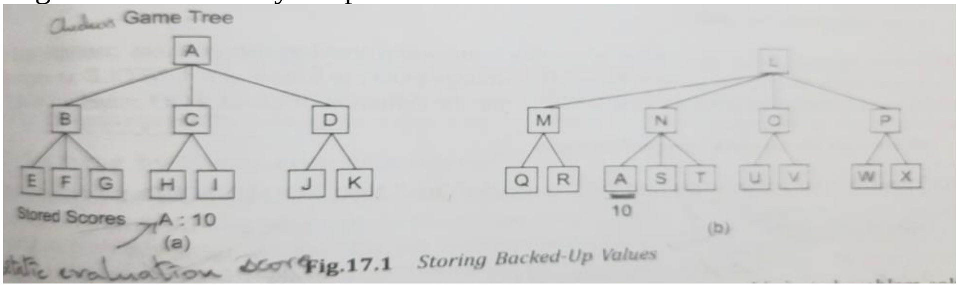
9 b. WRITE A NOTE ON EXPLAINATION-BASED LEARNING AND EXPLAIN ROTE LEARNING WITH AN EXAMPLE

Rote Learning

- It is mechanism of Caching
- When computer stores piece of data, data is cached so that recomputing is not required.
- Caching is used by AI to improve performance & such caching is known as *Rote Learning*.
- *Samuel's Checkers* program used 2 types of learning: rote learning & parameter adjustment.
- Samuels program used checkers game trees for representing states.



- Time constraints permitted it to search only a few levels in tree.
- Later static evaluation method was applied to board position.
- Then used the score obtained from this function to search the game tree.
- When search was complete, it propagated the score backwards and root position had a score.
- It now chose the best move and also recorded the board position and score at tree root.
- Instead of using the static evaluation function to compute a score for position A, the stored value for A can be used.
- Rote learning of this sort is very simple.

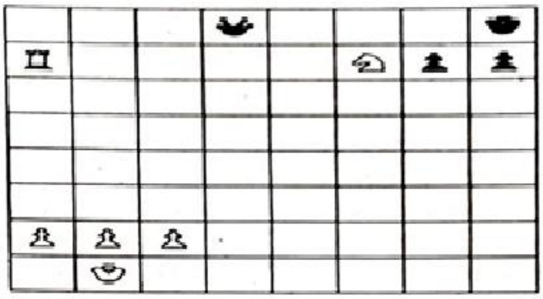


Explanation-Based Learning

- Consider a chess player who has reached the board position shown.
- Black must move the king, leaving the queen to be captured.
- From this experience black learns about FORK trap.
- FORK trap: if x attacks opponent’s king & y, y will be lost.
- What makes such single example learning possible?

Answer: **Knowledge**

- Knowledge can be used to identify critical aspects of the training example
- This is more analytical knowledge intensive approach & is referred as Explanation Based Learning (EBL)



A Fork Position in Chess

- EBL programs accept the following as input:

1. **A Training Example:** What the Learning Program sees in the world. Ex: the Car
2. **A Goal Concept:** A high level description of what the program is supposed to learn
3. **An Operationally Criterion:** A description of which concepts are usable.
4. **A Domain Theory:** A set of rules that describe relationships between objects & actions in a domain.

- From this EBL computes generalization of the training example that is sufficient to describe the goal concept & also satisfies the operationality criterion.
- An EBL program seeks to operationalize the goal concept by expressing it in terms that a problem-solving program can understand.
- Explanation Based Generalization(EBG) is an algorithm for EBL.

9 c. DISCUSS THE CONCEPT OF LEARNING FROM TAKING ADVICE

Learning by Taking Advice

- When a programmer writes a series of instructions into a computer.
- The programmer is a sort of teacher & the computer is a sort of student.
- After being programmed, the computer is able to do something it previously could not.
- Suppose the program is written in a high-level language like LISP.
- Some interpreter or compiler must intervene to change the teacher’s instructions into code that the machine can execute directly.
- A program called FOO, which accepts advice for playing hearts, a card game.
- A human user first translates the advice from English into a representation that FOO can understand.
- For example, “ Avoid taking points” becomes:
  (avoid(take-points me) (trick))
- FOO must *operationalize* this advice by turning it into an expression that contains concepts & actions FOO can use when playing the game of hearts.
- One strategy is: UNFOLD an expression by replacing some term by its definition.
- FOO comes up with:

```
(achieve(not(during(trick)(take-points me) )))
```

- FOO considers advice to apply to the player called “me”.
- Next, FOO UNFOLDs definition of trick

```
(achieve(not(during(
        scenario
                (each pl(players)(play-card pl)
                        (take-trick (trick-winner)))
                                (take-points me) )))
```

- In other words , player should avoid taking points during the scenario consisting of

1. Players playing cards &
2. One player taking the trick

- FOO then uses *case analysis* to determine which steps could cause one to take points.
- Step 2 could affect taking points , so FOO-UNFOLDs the definition of take-points:

```
(achieve (not (there-exists cl(cards-played)
        (there-exists c2 (point-cards)
                        (during(take(trick-winner)cl)
                                (take me c2))))))
```

- This device says that the player should avoid taking point-cards during the process of trick-winner taking the trick.
- The question for FOO now is: Under what conditions does (take me c2) occur during (take(trick-winner)cl)?
- By using technique called *partial match*, FOO hypothesizes that points will be taken if me= trick-winner and c2=c1.
- It transforms the advice into:

```
        (achieve (not ( and( have- points (cards-played))
                (=(trick-winner) me ))))
```

- This means “ Do not win a trick that has points”.
- Through a number of other transformations, FOO eventually settles on:

```
        (achieve (>= ( and(in-suit-led(card-of me))
                        (possible (trick-has-points)))
                                (low(card-of me)))
```

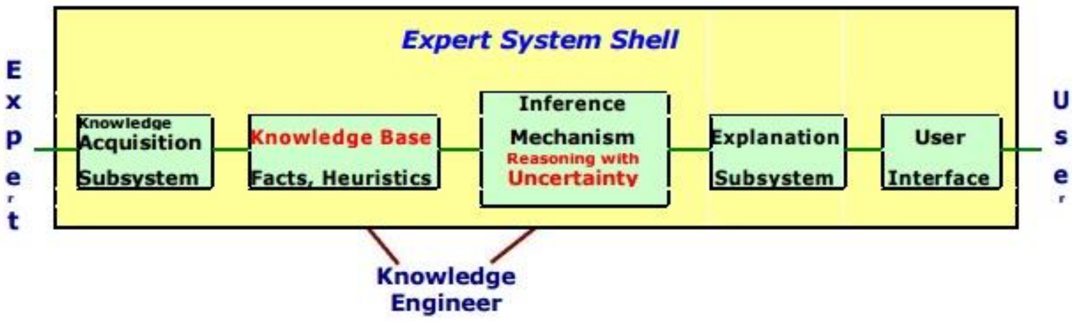
- At last, FOO has translated the rather vague advice “avoid taking points” into a specific, usable heuristic.
- FOO is able to play a better game of hearts after receiving this advice.

10 a. WHAT DO YOU MEAN BY EXPERT SHELL

- An Expert system shell is a software development environment. It contains the basic components of expert systems
- A shell is associated with a prescribed method for building applications by configuring and instantiating these components.

Shell components and description

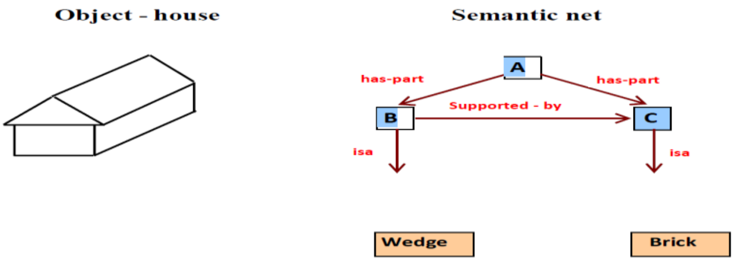
- The generic components of a shell: the knowledge acquisition, the knowledge Base, the reasoning, the explanation and the user interface are shown below.
- The knowledge base and reasoning engine are the core components.


Expert System Shell
Expert
Knowledge Acquisition Subsystem
Knowledge Base Facts, Heuristics
Inference Mechanism Reasoning with Uncertainty
Explanation Subsystem
User Interface
User
Knowledge Engineer

10 b. EXPLAIN WINSTON'S LEARNING PROGRAM

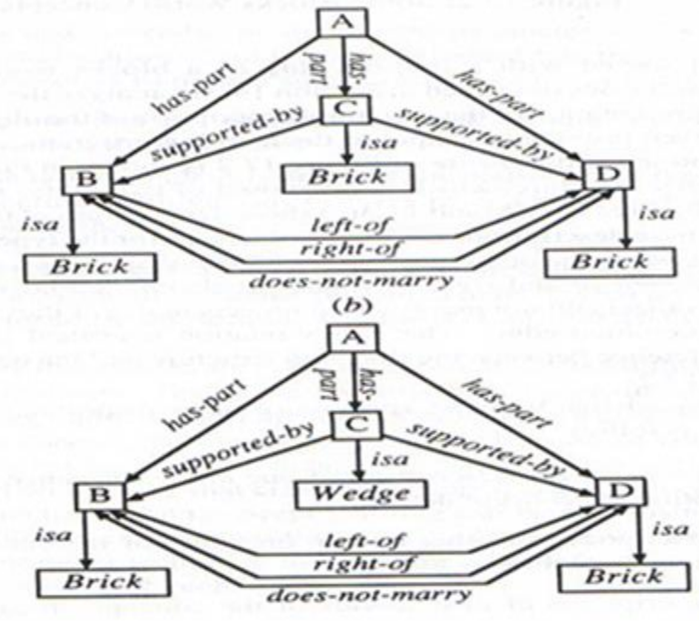
Winston's Learning Program

- The goal was "to construct representations of the definitions of concepts in blocks domain.
- **Near Miss:** it's an *Object* similar to *instances* of *concept* in question.



- The program started with a line drawing of a blocks world structure.
- Then structural description was provided as input to learning program.
- An example of such a structural description for the house is shown here.
- Node A represents entire structure. Its composed of node B(Wedge) and C(Brick)

- Then structural description of arcs.
- **2 objects Marry: if** they have faces that touch & have common edge.
- Marry Relation=
- Arch – Near miss Arch



- In comparison of 2 arches: the objects represented by node C are not identical.
- **C-note** link describes **difference** found by **comparison routine.** The difference occurred in **isa link**.

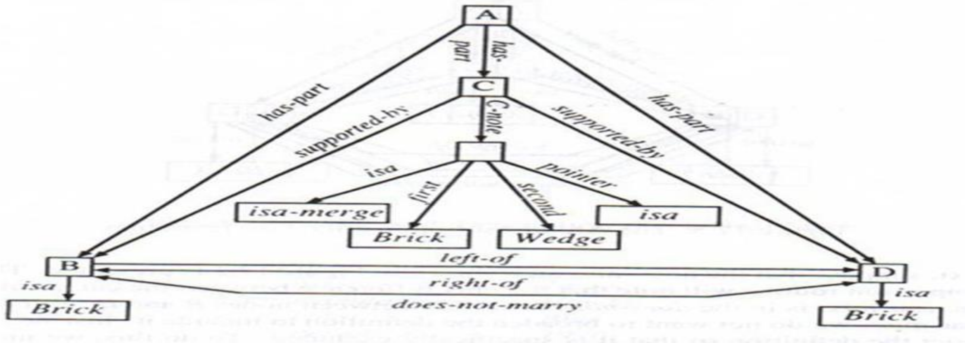


Figure 17.4: The Comparison of Two Arches

10 c. DISCUSS THE CONCEPT OF LEARNING BY CHUNKING.

Learning by Chunking

- Chunking is a process similar to macro-operators. Its universal learning method .
- Its computational basis is in production systems.
- Chunks are generalized before they are stored.
- Chunking can be used to learn general search control knowledge in addition to operator sequences.
- Chunks are generally applicable towards any goal state.
- Macro tables are structured toward reaching a particular goal state from any initial state.
- Chunking emphasizes how learning can occur during problem-solving, while macro tables are usually built during a preprocessing stage.