4.8 Inference in FOL

4.8.1 Forward Chaining

University Question

Q. Explain forward-chaining and backward-chaining algorithm with the help of example. **MU - Dec. 19**

- For any type of inference there should be a path from start to goal. When based on the available data a decision is taken, then the process is called as the forward chaining. Forward chaining or data-driven inference works from an initial state, and by looking at the premises of the rules (IF-part), perform the actions (THEN-part), possibly updating the knowledge base or working memory. This continues until no more rules can be applied or some cycle limit is met.
- For example, "If it is raining then, we will take umbrella". Here, "it is raining" is the data and "we will take umbrella" is a decision. This means it was already known that it's raining that's why it was decided to take umbrella. This process is **forward chaining.**

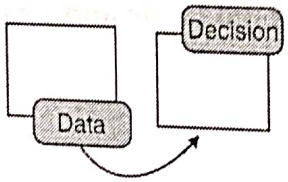


Fig. 4.8.1 : Forward Chaining

- "Forward chaining" is called as a data-driven inference technique.
- Example

  Given :
  - Rule : human(A) → mortal(A)
  - Data : human(Mandela)
- To prove : mortal(Mandela)

Forward Chaining Solution

- Human (Mandela) matches Left Hand Side of the Rule. So, we can get A = Mandela
- based on the rule statement we can get : mortal(Mandela)
- Forward chaining is used by the "design expert systems", as it performs operation in a forward direction (i.e. from start to the end).

Example

- Consider following example. Let us understand how the same example can be solved using both forward.
- Given facts are as follows:
  1. It is a crime for an American to sell weapons to the enemy of America.
  2. Country Nono is an enemy of America.
  3. Nono has some missiles.
  4. All the missiles were sold to Nono by Colonel West.
  5. Missile is a weapon.
  6. Colonel West is American.

- We have to prove that West is a criminal.
- Let's see how to represent these facts by FOL.

1. It is a crime for an American to sell weapons to the enemy nations.
   - American(x) Λ Weapon(y) Λ sell (x, y, z) Λ enemy(z, America) => Criminal (x)
2. Country Nono is an enemy of America.
   - Enemy (Nono, America)
3. Nono has some missiles.
   - Owns (Nono, x)
   - Missile(x)
4. All the missiles were sold to Nono by Colonel West.
   - Missile(x)Λ owns(Nono, x)=> Sell(West, x, Nono)
5. Missile is a weapon.
   - Missile(x)=> weapon(x)
6. Colonel West is American.
   - American (West)

Proof by forward chaining

- The proof will start from the given facts. And as we can derive other facts from those, it will lead us to the solution. Please refer to Fig. 4.8.2 As we observe from the given facts we can reach to the predicate Criminal (West).

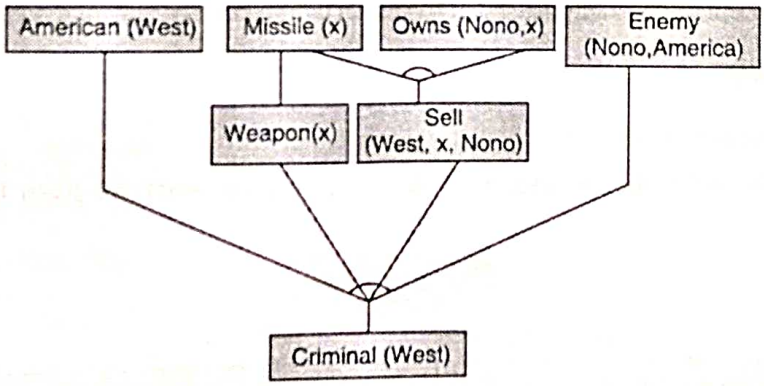


Fig. 4.8.2 : Proof by forward chaining

4.8.2 Backward Chaining

University Question

Q. Describe backward chaining algorithm with an example. **MU - Dec. 12, May 14, Dec. 19**

- If based on the decision the initial data is fetched, then it is called as **backward chaining**. Backward chaining or goal-driven inference works towards a final state, and by looking at the working memory to see if goal already there. If not look at the actions (THEN-parts) of rules that will establish goal, and set up sub-goals for achieving premises of the rules (IF-part). This continues until some rule can be applied, apply to achieve goal state.

- **For example,** If while going out one has taken umbrella. Then based on this decision it can be guessed that it raining. Here, "taking umbrella" is a decision based on which the data is generated that "it's raining". This proces is **backward chaining.** "Backward chaining" is called as a decision-driven or goal-driven inference technique.

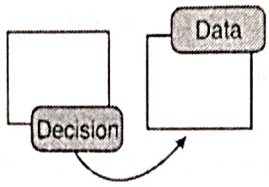


Fig. 4.8.3 : Backward Chaining

- **Given :**
  - Rule : human(A) → mortal(A)
  - Data : human(Mandela)
- To prove : mortal (Mandela)

Backward Chaining Solution

- mortal (Mandela) will be matched with mortal (A) which gives human (A) i.e. human (Mandela) which is also given fact. Hence proved.
- It makes use of right hand side matching. backward chaining is used by the "diagnostic expert systems", becaus it performs operations in a backward direction (i.e. from end to start).

Example

Let us understand how the same example used in forward chaining can be solved using backward chaining.

Proof by backward Chaining

The proof will start from the fact to be proved. And as we can map it with given facts, it will lead us to th solution. Please refer to Fig. 4.8.4. As we observe, all leaf nodes of the proof are given facts that means "West Criminal".

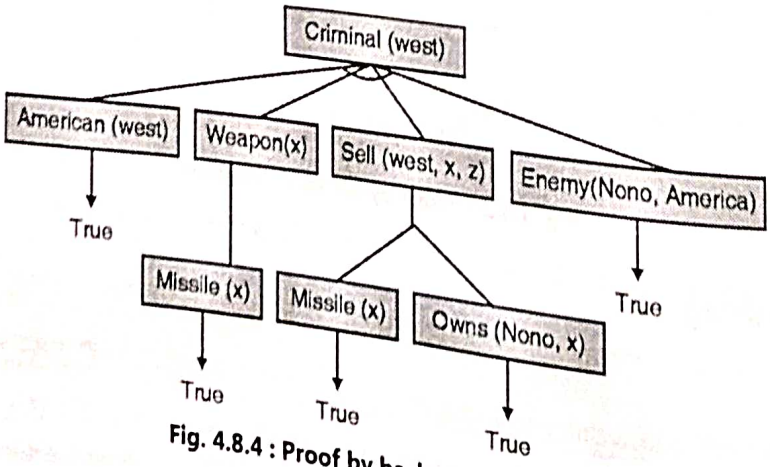


Fig. 4.8.4 : Proof by backward Chaining

4.8.3 Differentiate between Forward Chaining and backward Chaining

Attribute	Backward Chaining	Forward Chaining
Also known as	Goal-driven	Data-driven
Starts from	Possible conclusion	New data
Processing	Efficient	Somewhat wasteful
Aims for	Necessary data	Any Conclusion(s)
Approach	Conservative/Cautious	Opportunistic
Practical if	Number of possible final answers is reasonable or a set of known alternatives is available.	Combinatorial explosion creates an infinite number of possible right answers.
Appropriate for	Diagnostic, prescription and debugging application	Planning, monitoring, control and interpretation application
Reasoning	Top-down reasoning	Bottom-up reasoning
Type of Search	Depth-first search	Breadth-first search
Who determine search	Consequents determine search	Antecedents determine search
Flow	Consequent to antecedent	Antecedent to consequent

Ex. 4.8.1 : Using predicate logic find the course of Anish's liking for the following :

(i) Anish only likes easy courses.

(ii) Computer courses are hard.

(iii) All electronics courses are easy

(iv) DSP is an electronics course.

Soln. :

Step 1 : Converting given facts to FOL

(i) $\forall x$: course (x) $\wedge$ easy (x) $\rightarrow$ likes (Anish, x)

(ii) $\forall x$: course (x) $\wedge$ computes (x) $\rightarrow$ hard (x)

(iii) $\forall x$: course (x) $\wedge$ electronics (x) $\rightarrow$ easy (x)

(iv) Electronics (DSP)

(v) course (DSP)

Step 2 : Proof by backward chaining :

As we have to find out which course Anish likes. So we will start the proof from the same fact.

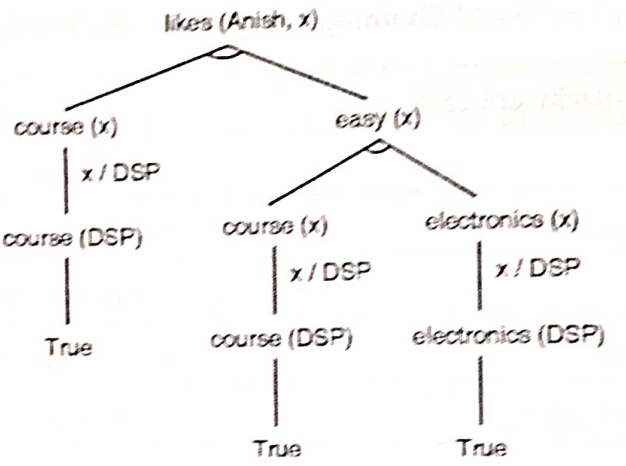


Fig. P. 4.8.1

Hence proved that Anish likes DSP course.

4.9 Knowledge Engineering in First order Logic

Knowledge engineering is a process of knowledgebase construction. It requires a knowledge engineer to investigate a particular domain, learn the important concepts in that domain, and create a formal representation and logical relations among objects in that domain.

4.9.1 Knowledge Engineering Process

Following is the general knowledge engineering process which can be applied to problem of any domain.

1. **Identify the task :** This step is analogues to PEAS process while designing an agent. While identifying task the knowledge engineer must define the scope of knowledgebase and the range of questions that can be answered through the database. He also need to specify the type of facts that will be available for each specific problem instance.

2. **Assemble the relevant knowledge :** Assembling the relevant knowledge of that particular domain is called the process of knowledge acquisition. In this the knowledge engineer needs to extract the domain knowledge either by himself provided he is the domain expert or needs to work with the real experts of the domain. In this process knowledge engineer learns how the domain actually works and can determine the scope of the knowledgebase as per the identified tasks.

3. **Defining vocabulary :** Defining a complete vocabulary including predicates, functions and constants is a very important step of knowledge engineering. This process transforms the domain level concepts to logic level symbols. It should be exhaustive and precise. This vocabulary is called as ontology of the domain. And once the ontology is defined, it means, the existence of the domain is defined. That is, what kind of things exist in the domain is been decided.

4. **Encoding of general knowledge about the domain :** In this step the knowledge engineer defines axioms for all the vocabulary terms by define meaning of each term. This enables expert to cross check the vocabulary and its contents. If he finds any misinterpretations or gaps, it can be fixed at this point by redoing step 3.

Ex. 4.10.1 : Represent following sentences in FOL using a consistence vocabulary.

(i) Every person who buys a policy is smart.

(ii) No person buys an expensive policy.

(iii) There is an agent who sells policies only to people who are not insured.

(iv) There is a barber who shaves all men in town who do not save themselves.

MU - Dec. 12

Soln. :

(i) $\forall x \forall y : \text{person}(x) \wedge \text{policy}(y) \wedge \text{buys}(x, y) \rightarrow \text{smart}(x)$

(ii) $\forall x, \forall y : \text{person}(x) \wedge \text{policy}(y) \wedge \text{expensive}(y) \rightarrow \sim \text{buys}(x, y)$

(iii) $\forall x : \text{person}(x) \wedge \sim \text{insured}(x)$

$\forall y : \exists x \wedge \text{policy}(y) \sim \text{agent}(x) \rightarrow \text{sells}(x, y, x)$

(iv) $\exists x \ \forall y : \text{barber}(x) \wedge \text{person}(y) \wedge \sim \text{shaves}(y, y) \rightarrow \text{shaves}(x, y)$

Ex. 4.10.2 : Represent following statements in FOPL.

1. Anyone who kills an animal is loved by no one.
2. A square is breezy if there is a pit in the neighbouring squares.

MU - Dec. 15

Soln. :

(i) $\forall x \forall y : \text{kills}(x, \text{animal}) \rightarrow \sim \text{loves}(y, x)$

(ii) $\forall x \forall y : \text{pit}(x, y) \rightarrow \text{breez}(x, y - 1) \wedge \text{breez}(x, y + 1) \wedge \text{breez}(x - 1, y) \wedge \text{breez}(x + 1, y)$

Ex. 4.10.3 : Write first order logic statements for following statements :

(i) If a perfect square is divisible by a prime p then it is also divisible by square of p.

(ii) Every perfect square is divisible by some prime.

(iii) Alice does not like chemistry and history.

(iv) If it is Saturday and warm, then sam is in the park.

(v) Anything anyone eats and is not killed by is food.

MU - May 16

Soln. :

(i) $\forall x : \text{square}(x) \wedge \text{prime}(y) \wedge \text{divides}(p, x) \rightarrow [\exists z : \text{square\_of}(z, p) \wedge \text{divides}(z, x)]$

(ii) $\forall x \exists y : \text{square}(x) \wedge \text{divides}(p, x)$

(iii) $\sim \text{likes}(\text{Alice}, \text{History}) \wedge \sim \text{likes}(\text{Alice}, \text{Chemistry})$

(iv) $\text{day}(\text{Saturday}) \wedge \text{weather}(\text{warm}) \rightarrow \text{in\_park}(\text{Sam})$

(v) $\forall x : \forall y : \text{person}(x) \wedge \text{eats}(x, y) \wedge \sim \text{killed}(x) \rightarrow \text{food}(y)$

4.11 Resolution

University Question

Q. Write a short note on : Resolution. **MU - May 14**

3. Eliminate existential quantifiers by replacing with Skolem constants or Skolem functions :
   e.g. $\forall X \exists Y(P_1(X,Y) \vee (P_2(X,Y)) \equiv \forall X (P_1(X, f(X)) \vee (P_2(X, f(X)))$
4. Rename variables to avoid duplicate quantifiers.
5. Drop all universal quantifiers
6. Place expression into Cconjunctive Normal Form.
7. Convert to clauses i.e. separates all conjunctions as separate clause.
8. Rename variables to avoid duplicate clauses.

Ex. 4.11.1 : Convert following propositional logic statement into CNF. A→(B→ C) **MU - Dec. 15, 4 Marks**

Soln. :

FOL : $A \rightarrow (B \leftrightarrow C)$

Normalizing the given statement.

(i) $A \rightarrow (B \rightarrow C \wedge C \rightarrow B)$

(ii) $(A \rightarrow (B \rightarrow C)) \wedge (A \rightarrow (C \rightarrow B))$

Converting to CNF.

Applying Rule, $\alpha \rightarrow \beta \equiv \sim \alpha \vee \beta$

$\sim A \vee (\sim B \vee C) \wedge \sim A \vee (\sim C \vee B)$

i.e. $\sim A \vee ((\sim B \vee C) \wedge (\sim C \vee B))$

4.11.3 Facts Representation

- To show how facts can be represented let's take a simple problem:
  - "Heads X wins, Tails Y loses."
  - Our goal is to show that X always wins with the help of resolution.
- Solution can be given as follows :
  1. $H \Rightarrow Win(X)$
  2. $T \Rightarrow Loose(Y)$
  3. $\neg H \Rightarrow T$
  4. $Loose(Y) \Rightarrow Win(X)$

  Thus we have : Win(X)

  We can write a proof for this problem as follws :
  1. $\{\neg H, Win(X)\}$
  2. $\{\neg T, Loose(Y)\}$
  3. $\{H, T\}$
  4. $\{\neg Loose(Y), Win(X)\}$

5. {¬ Win(X)}
6. {¬T, Win(X)} ..... (From 2 and 4)
7. {T, Win(X)} ..... (From 1 and 3)
8. {Win(X)} ..... (From 6 and 7)
9. {} ..... (From 5 and 8)

4.11.4 Example

Let's take the same example of forward and backward chaining to learn how to write proofs for resolution.

Step 1 :

The given facts are :

1. It is a crime for an American to sell weapons to the enemy nations.
   - American(x) Λ Weapon(y) Λ sell (x,y,z) Λ enemy(z, America) => Criminal (x)
2. Country Nono is an enemy of America.
   - Enemy (Nono, America)
3. Nono has some missiles.
   - Owns (Nono, x)
   - Missile(x)
4. All the missiles were sold to Nono by Colonel West.
   - Missile(x)Λ owns(Nono, x)=> Sell(West, x, Nono)
5. Missile is a weapon.
   - Missile(x)=> weapon(x)
6. Colonel West is American.
   - American (West)

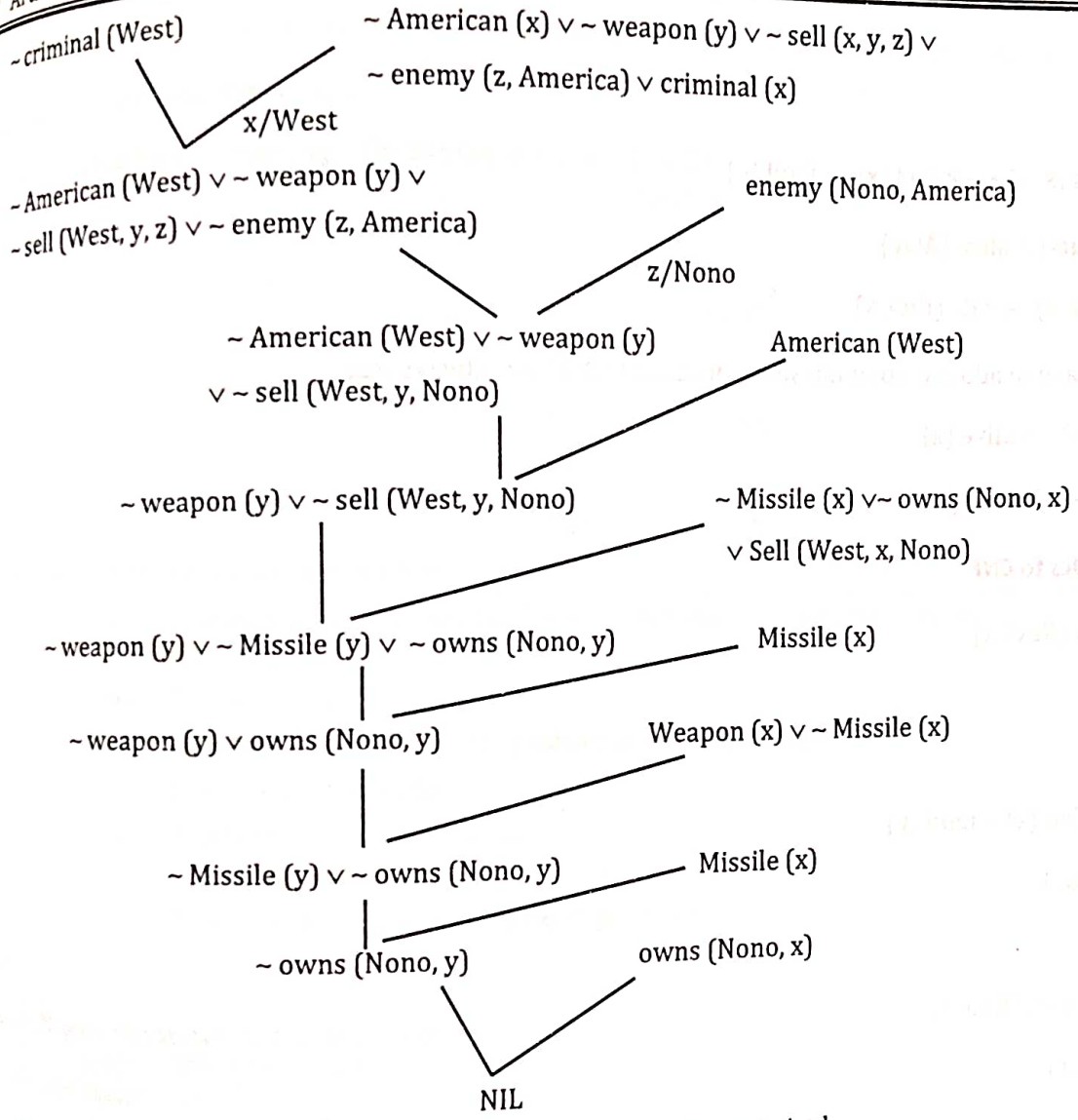
Step 2 :

Lets convert them to CNF

1. ~ American(x) V ~ Weapon(y) V~ sell (x, y, z) V ~ enemy(z, America)
   - V Criminal (x)
2. Enemy (Nono, America)
3. Owns (Nono, x)
4. Missile(x)
5. ~ Missile(x) V ~ owns(Nono, x) V Sell(West, x, Nono)
6. ~Missile(x) V weapon(x)
7. American (West)

Step 3 :

To prove that West is criminal using resolution.



Hence our assumption was wrong. Hence proved that West is criminal.

Ex. 4.11.2 : Consider following statements :

(a) Ravi Likes all kind of food.

(b) Apple and Chicken are food

(c) Anything anyone eats and is not killed is food.

(d) Ajay eats peanuts and still alive.

(e) Rita eats everything that Ajay eats.

Prove that Ravi Likes Peanuts using resolution. What food does Rita eat? **MU - Dec. 15**

Soln. :

(A) Proof by Resolution

Step 1 : Negate the statement to be proved.

~ likes (Ravi, Peanuts)

Step 2 : Convert given facts to FOL

(a) $\forall$ x, food (x) $\rightarrow$ likes (Ravi, x)

(b) food (Apple)

(c) food (Chicken)

(d) $\forall$ x $\forall$ y : eats (x, y) $\wedge$ ~ killed (x) $\rightarrow$ food (y)

(e) eats (Ajay, Peanuts) $\wedge$ alive (Ajay)

(f) $\forall$ x : eats (Ajay, x) $\rightarrow$ eats (Rita, x)

In this case we have to add few common sense predicate which are always true.

(g) $\forall$ x : ~ killed (x) $\rightarrow$ alive (x)

(h) $\forall$ x : alive (x) $\rightarrow$ ~ killed (x)

Step 3 : Converting FOLs to CNF

(a) ~ food (x) $\vee$ likes (Ravi, x)

(b) food (Apple)

(c) food (Chicken)

(d) ~ eats (x, y) $\vee$ killed (x) $\vee$ food (y)

(e) eats (Ajay, Peanuts)

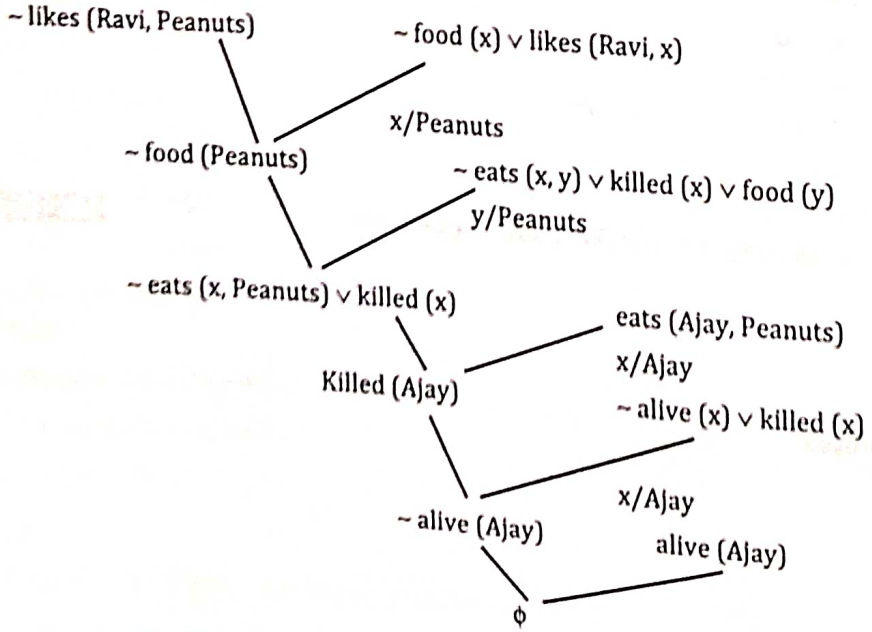
(f) alive (Ajay)

(g) ~ eats (Ajay, x) $\vee$ eats (Rita, x)

(h) killed (x) $\vee$ alive (x)

(i) ~ alive (x) $\vee$ killed (x)

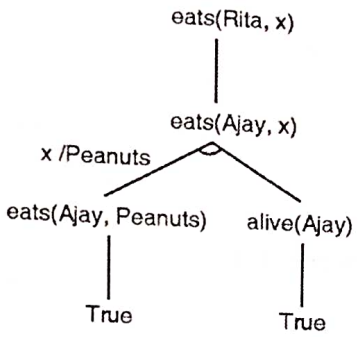
Step 4 : Proof by Resolution



As the result of this resolution is NIL, it means out assumption is wrong. Hence proved that "Ravi likes Peanuts".

To answer : What food Rita eats ?

(B) Proof by backward chaining : (Referring to FOLs of Step 2)



Hence the answer is Rita eats peanuts.

Ex. 4.11.3 : Using a predicate logic convert the following sentences to predicates and prove that the statement "Ram did not jump" is false.

(a) Ram went to temple.

(b) The way to temple is, walk till post box and take left or right road.

(c) The left road has a ditch.

(d) Way to cross the ditch is to jump

(e) A log is across the right road.

(f) One needs to jump across the log to go ahead.

Soln.:

Step 1 : Negate the statement to be proved.

~ jump (Ram)

Step 2 : Converting given statement to FOL

(a) At (Ram, temple)

(b_1) $\forall$ x : At (x, temple) $\longrightarrow$ At (x, PostBox) $\wedge$ take left (x)

(b_2) $\forall$ x : At (x, temple) $\longrightarrow$ At (x, PostBox) $\wedge$ take right (x)

(c) $\forall$ x: take left (x) $\longrightarrow$ cross (x, ditch)

(d) $\forall$ x cross (x, ditch) $\longrightarrow$ jump (x)

(e) $\forall$ x : take right (x) $\longrightarrow$ at (x, log)

(f) $\forall$ x : at (x, log) $\longrightarrow$ jump (x)

Step 3 : Converting FOLs to CNF

(a) At (Ram, temple)

(b_{11}) ~ At (x, temple) $\vee$ At (x, PostBox)

(b_{12}) ~ At (x, temple) $\vee$ take left (x)

(b_{21}) ~ At (x, temple) ∨ At (x, PostBox)

(b_{22}) ~ At (x, temple) ∨ take right (x)

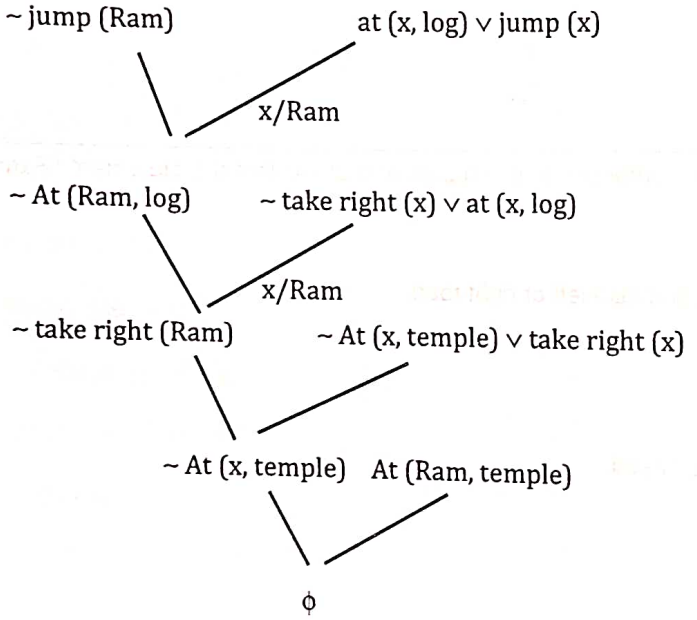
(c) ~ take left (x) ∨ cross (x, ditch)

(d) ~ cross (x, ditch) ∨ jump (x)

(e) ~ take right (x) ∨ at (x, log)

(f) ~ at (x, log) ∨ jump (x)

Step 4 : Proof by Resolution



Hence proved.

Ex. 4.11.4 : Consider following statements.

1. Rimi is hungry.
2. If Rimi is hungry she barks.
3. If Rimi is barking then Raja is angry.

Explain statements in predicate logic. Convert them into CNF form.

Prove that Raja is angry using resolution.

Soln. :

Step 1 : Converting given facts to FOL.

1. Hungry (Rimi)
2. Hungry (Rini) → barks (Rimi)
3. Barks (Rimi) → angry (Raja)

Step 2 : Converting FOL statements to CNF.

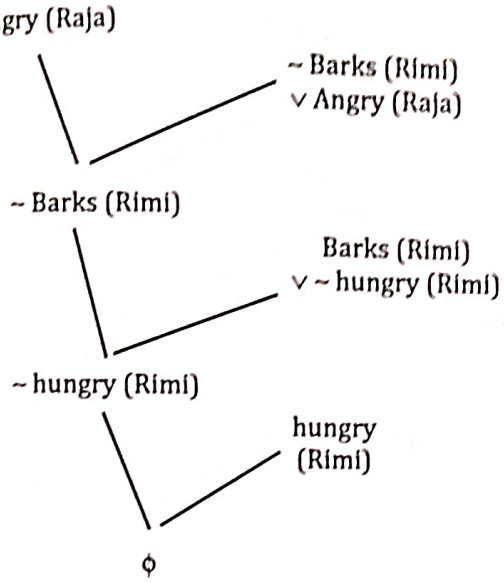
1. Hungry (Rimi)
2. ~ hungry (Rimi) ∨ barks (Rimi)
3. ~ barkes (Rimi) ∨ angry (Raja)

Step 3 : Negate the stmt to be proved

T.P.T. Angry (Raja)

Negation : ~ Angry (Raja)

Step 4 : Proof by resolution



This shows that our assumption is Wrong. Hence proved that **Raja is Angry.**

Ex. 4.11.5 : Consider following facts.

1. If maid stole the jewelry then butler was not guilty.
2. Either maid stole the jewelry or she milked the cow.
3. If maid milked the cow then butler got the cream.
4. Therefor if butler was guilty then he got the cream.

Prove the conclusion (step 4) is valid using resolution.

Soln. :

Step 1 : Converting given facts to FOL.

1. steal (maid, jwellary) → ~ guilty (butler)
2. steal (mail, jwellary) ∨ milk (maid, cow)
3. milk (maid, cow) → got_Cream (butter)

To prove that

4. guilty (butler) → got_cream (butter)

Step 2 : Converting FOL to CNF.

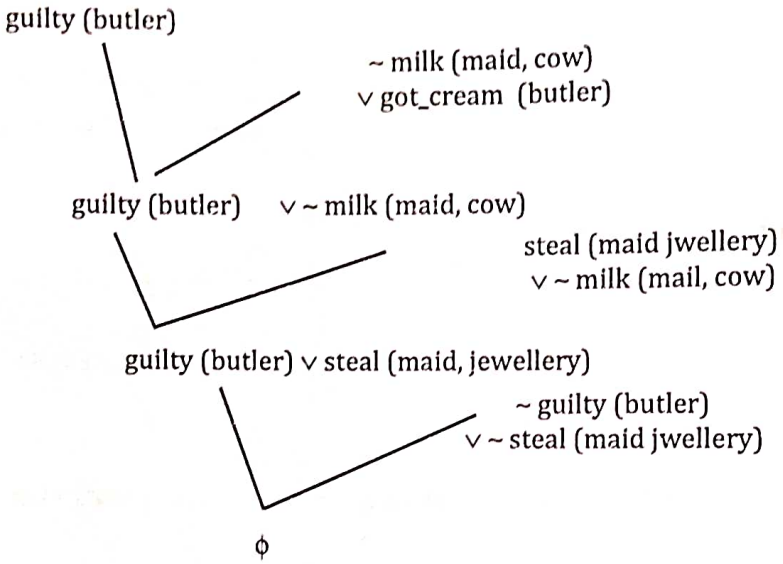
1. ~ steal (maid, jwellary) ∨ ~ guilty (butler)
2. steal (maid, jwellary) ∨ milk (maid, cow)
3. ~ milk (maid, cow) ∨ got_cream (butler)
4. ~ guilty (butler) ∨ got_cream (butler)

Step 3 : Negate the proof sentence

As sentence 4 is the one to be proved.

guilty (butler) $\wedge$ ~ got_ cream (butler)

Step 4 : Proof by resolution $\wedge\vee$ got_cream (butter)



Hence proved.

Ex. 4.11.6 : Consider following axioms.

All people who are graduating are happy.

All happy people smile.

Someone is graduating.

(i) Represent these axioms in FOL.

(ii) Convert each formula to CNF.

(iii) Prove that someone is smiling using resolution technique. Draw the resolution tree.

MU - Dec. 15, 12 Ma

Soln. :

Step 1 : Converting axioms to FOL.

(i) $\forall x$: graduating (x) $\rightarrow$ happy (x).

(ii) $\forall x$: happy (x) $\rightarrow$ smile (x)

(iii) $\exists x$: graduating (x)

Step 2 : Converting FOL to CNF.

(i) ~ graduating (x) $\vee$ happy (x)

(ii) ~ happy (x_1) $\vee$ smile (x_1)

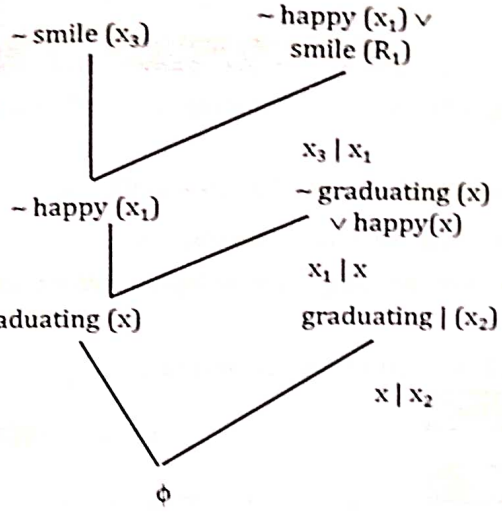
(iii) graduating (x_2)

Step 3 : T.P.T. x_3 smile (x_3)

Negating the stmt

~ smile (x_3)

Proof by resolution



Hence our assumption is wrong.

Hence proved.

4.12 Programming Logic

Languages like C, C++, JAVA are used for creating intelligent softwares, But for artificial intelligence there are two specifically designed languages named : LISP and PROLOG.

LISP

- LISP stands for **LIST Programming**. LISP is a programming language which was invented by John McCarthy in 1958.
- As the name suggests LISP manipulates lists.

PROLOG

- PROLOG stands for **Programming** in **LOGIC**. PROLOG is a programming language which was invented by Alain Colmerauer in 1970s.
- PROLOG program can use logical reasoning in order to answer questions which can be inferred from the knowledge base.
- Prolog was initially intended for programming of NLP applications, but it was further developed and now it is adapted in various areas like :
  - Planning.
  - Formal logic and associated forms of programming.
  - Reasoning modelling.

6 AI Applications

Syllabus

Introduction to NLP : Language models, Grammars, Parsing.

Robotics : Robots, Robot hardware, Problems Robotics can solve AI applications in Healthcare, Retail, Banking.

6.1 Natural Language Processing (NLP)

University Questions

Q. Write short note on natural language Processing. **MU - May 10**

Q. Explain different components of natural language processing. **MU - Dec. 19**

As the name suggests, Natural Language Processing, involves machines or robots to understand and process the language that human speak, and infer knowledge from the speech input. It also involves the active participation from machine in the form of dialog i.e. NLP aims at the text or verbal output from the machine or robot. The input and output of an NLP system can be speech and written text respectively.

6.1.1 Components of NLP

Mainly there are two components of NLP.

1. **Natural Language Understanding (NLU)**

In this part of the process, the speech input gets transformed into the useful representations in order to analyse various aspects of the language. As the natural language is very reach in forms and structures, it is also very ambiguous. There can be different forms of ambiguities like **lexical ambiguity**, which is a very basic i.e. word level ambiguity. For example the "document" can be a noun or verb. It's a complicated process. Secondly, there can be **syntactical ambiguity**, which is about parsing the sentence. For example, a sentence like "Madam said on Monday she would give an exam". Thirdly, there can be **referential ambiguity**. Check a sentence, "Meera went to Geeta. She said "I am Hungry"."Who is hungry, is not well referred from this sentence. In many cases we observe that one sentence can have meanings. And reversely, many sentences mean the same. Hence NLU a complicated process.

2. **Natural Language Generation (NLG)**

In order to generate the output text, the intermediate representation requires to be converted back to the natural language format. Hence, in this process there are multiple sub processes involves. They are as follow :

a) **Text Planning :** It includes extracting relevant contents from knowledge base.

b) **Sentence planning :** This process involves selecting correct words, forming meaningful sentence following language grammar and setting tone for the same.

c) **Text Realization :** This is the process of mapping the planned sentence into a structure.

6.2 The Steps Involved in NLP

University Question

Q. Explain the levels of knowledge used in NLP. **MU - Dec. 15, Dec. 19**

As natural language is very reach in forms and structures, NLP by default is a complicated process. In general, it can be divided into five steps as shown in Fig. 6.2.1.

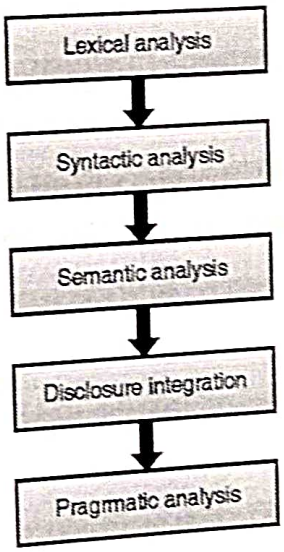


Fig. 6.2.1 : Steps of NLP process

1. Lexical Analysis

Lexicon is the words and phrases in language. Lexicon analysis deals with the recognition and identification of structure of the sentences. It divides the paragraphs in sentences, phrases and words.

2. Syntactic Analysis

In syntactic analysis the sentences are parsed as noun, verbs, adjectives, and other parts of sentences. In this phase the grammar of the sentence is analyzed in order to get the relationships among different words in the sentence. For example, "mongo eats me" will be rejected by syntactic analyzer.

3. Semantic Analysis

In this phase, the actual meaning of the sentence is extracted from the structure and the words used. It checks whether the sentence is meaningful. It maps the object with their syntactic structure to decide the correctness of the sentence. For example, "bitter sugar" will be termed as a wrong sentence by the analyzer.

6.6 Applications of NLP

NLP has a huge number of applications. Few to mention are as follow :

1. **Automatic summarization**

   NLP techniques are used to produce a readable summary of a chunk of text. This application is mainly found in summarizing of text of a known type such as articles in the financial section of a newspaper.

2. **Discourse analysis**

   In this application there are various related tasks. One is to identify the discourse structure of connected text. i.e. the nature of relationship among sentences.(e.g. explanation, contrast, elaboration, etc.) Another possible task is recognizing and classifying the speech acts in a chunk of text (e.g. yes-no question, content question, statement, assertion, etc.).

3. **Machine translation**

   This is the most difficult application of NLP. Machine translation is automatically translating text from one human language to another. It requires all of the different types of knowledge that humans possess (e.g. grammar, semantics, facts about the real world, etc.) in order to carry out the translation in a proper manner.

4. **Named entity recognition (NER)**

   In NER, given a stream of text, NLP system determines which items in the text map to proper names, such as people or places, and what the type of each such name is (e.g. person, location, organization). Although capitalization can aid in recognizing named entities in languages such as English, this information cannot aid in determining the type of named entity. For example, the first word of a sentence is also capitalized, and named entities often span several words, only some of which are capitalized. Furthermore, many other languages in non-Western scripts (e.g. Chinese or Arabic) do not have any capitalization at all, and even languages with capitalization may not consistently use it to distinguish names.

5. **Natural language generation**

   In this application of NLP, information from computer databases is converted into readable human language.

6. **Natural language understanding**

   In Natural language understanding, chunks of text is converted into more formal representations such as first-order logic structures that are easier for computer programs to manipulate. It involves the identification of the intended semantic from the multiple possible semantics which can be derived from a natural language expression.

7. **Part-of-speech tagging**

   In this application, the part of speech for each word is determined for given sentences. Many words, especially common ones, can serve as multiple parts of speech. For example, "book" can be a noun ("the book on the table") or verb ("to book a flight"); "set" can be a noun, verb or adjective; and "out" can be any of at least five different parts of speech. Some languages have more such ambiguity than others. Languages with little inflectional morphology, such as English are particularly prone to such ambiguity.

8. **Question answering**

Using NLP, one can determine answers of human-language questions. Typical questions have a specific right answer (such as "What is the capital of India?"), but sometimes open-ended questions are also considered (such as "What is the meaning of life?"). Recent works have looked at even more complex questions.

9. **Sentence breaking or sentence boundary disambiguation**

As the name suggest in this application of NLP, sentence boundaries are found from the given chunk of text. Sentence boundaries are often marked by periods or other punctuation marks, but these same characters(i.e. period or any punctuation mark) can serve other purposes (e.g. marking abbreviations). In such cases there can be ambiguities.

10. **Sentiment analysis**

In sentiment analysis, subjective information is extracted usually from a set of documents, to determine "polarity" about specific objects. It is especially useful for identifying trends of public opinion in the social media, for the purpose of marketing. It is one of the big applications of NLP.

11. **Speech recognition**

In this application, for a given sound clip of a person or people speaking, the textual representation of the speech is generated. This is the opposite of text to speech and is one of the extremely difficult problems.

12. **Speech segmentation**

In speech segmentation, from the given sound clip of a person or people speaking, words are separated. Speech segmentation is a subtask of speech recognition and typically grouped with it.

13. **Topic segmentation and recognition**

In this application, from the given a chunk of text, segments are separated, each of which is devoted to a topic, and thus the topic of the segment is identified.

14. **Word segmentation**

This is a significant application for foreign languages like Chinese, Japanese and Thai which do not have word separators (like space) used. Word segmentation separates the words in order to extract knowledge.

15. **Word sense disambiguation**

While using words that have more than one meaning; we have to select the meaning which makes the most sense in context. For this problem, we are typically given a list of words and associated word senses, e.g. from a dictionary or from an online resource such as WordNet.

6.7 Robotics – Robots

- **Robot** is any automatically operated machine that replaces human effort and perform functions in a humanlike manner. Robots were originally built to handle monotonous tasks (like building cars on an assembly line), but have since expanded well beyond their initial uses to perform tasks like fighting fires, cleaning homes and assisting with incredibly intricate surgeries.

- Each robot has a differing level of autonomy, ranging from human-controlled bots that carry out tasks that a human has full control over to fully-autonomous bots that perform tasks without any external influences.
- Robotics is the intersection of science, engineering and technology that produces machines, called robots, that substitute for or replicate human actions. By extension, robotics is the engineering discipline dealing with the design, construction, and operation of robots.
- The modern term *robot* derives from the Czech word *robota* ("forced labour" or "serf"), used in Karel Čapek's play R.U.R. (1920). The mechanical alternative inspired generations of inventors to build electrical humanoids.
- The word *robotics* first appeared in Isaac Asimov's science-fiction story *Runaround* (1942). It also contained Asimov's famous Three Laws of Robotics :
  1. A robot may not injure a human being, or, through inaction, allow a human being to come to harm.
  2. A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
  3. A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.
- Robots are gaining intellectual and mechanical capabilities that don't put the possibility of a R2-D2-like machine out of reach in the future.
- As technology progresses, so too does the scope of what is considered robotics. In 2005, 90% of all robots could be found assembling cars in automotive factories. These robots consist mainly of mechanical arms tasked with welding or screwing on certain parts of a car. Today, we're seeing an evolved and expanded definition of robotics that includes the development, creation and use of bots that explore Earth's harshest conditions, robots that assist law-enforcement and even robots that assist in almost every facet of healthcare.

6.7.1 Characteristics of Robot

1. Robots all consist of some sort of mechanical construction. The mechanical aspect of a robot helps it complete tasks in the environment for which it's designed. For example, the Mars 2020 Rover's wheels are individually motorized and made of titanium tubing that help it firmly grip the harsh terrain of the red planet.
2. Robots need electrical components that control and power the machinery. Essentially, an electric current (a battery, for example) is needed to power a large majority of robots.
3. Robots contain at least some level of computer programming. Without a set of code telling it what to do, a robot would just be another piece of simple machinery. Inserting a program into a robot gives it the ability to know when and how to carry out a task.

In the near future, with the advancements in artificial intelligence and software, robots will continue getting smarter, more flexible and more energy efficient. They'll also continue to be a main focal point in smart factories, where they'll take on more difficult challenges and help to secure global supply chains. From the deepest depths of our oceans to thousands of miles in outer space, robots will be found performing tasks that humans couldn't dream of achieving alone.

6.7.2 Robot hardware

Robot has the following hardware and software requirements :

HARDWARE / SOFTWARE	REQUIREMENTS
Operating system	Microsoft® Windows® 7 Enterprise, Ultimate, Professional Microsoft® Windows® 8 Pro, Enterprise
Browser	Microsoft ® Internet Explorer ® 8.0 or later
Processors	*Detailed list not defined (required support of SSE2 or higher instruction set)*
Memory	3 GB RAM for 32-bit OS 8 GB RAM for 64-bit OS
Display	1280 x 1024 monitor and display adapter capable of 24-bit color Dedicated video card with hardware support for OpenGL ® spec 1.4 or later and support for DirectX ® 9 or later
Disk Space	1 GB free disk space for the installation (8 GB if not installed from DVD) + 5 GB free disk space left after installation
Pointing Device	MS-Mouse compliant
Media (DVD)	Download and Installation from DVD
Internet connection	Internet connection for registration and cloud licensing/analysis if used
Additional requirements	For Robot™ Extension Spreadsheet Calculator additionally Microsoft ® Office Excel ® 2007 or 2010, 32-bit.

Robot has several modules that are each responsible for a specific step in the structure design: creating the model, analyzing the structure, and designing.

6.7.3 Problems Robotics can solve

1. **Security**

Robots are being proposed as security agents as they can protect humans. Currently, robotics companies are working on pairing robot guards with human security consultants. A very famous company in this field is Knightscope in the United States that has autonomous security robots capable of assisting human security guards with real-time, actionable intelligence. These robots can help with crimes such as armed robberies, burglaries, domestic violence, fraud, hit, and runs, etc.

2. **Space Exploration**

There are many things in space that are very dangerous for astronauts to do. Humans can't roam on Mars all day to collect soil samples or work on repairing a spaceship from the outside while it's in deep space! In these situations, robots are a great choice because there are no chances for the loss of human life then. So space institutions like NASA frequently use robots and autonomous vehicles to do things that humans can't. For example, Mars Rover is an autonomous robot that travels on Mars and takes pictures of Martian rock formations that are interesting or important and then sends them back on Earth for the NASA scientists to study.

3. Entertainment

Robots can play a big role in the entertainment industry. They can be used behind the sets in movies and serials to manage the camera, provide special effects, etc. They can be used for boring repetitive tasks that are not suitable for a human as cinema is, after all, a creative industry. Robots can also be used to do stunt work that is very dangerous for humans but looks pretty cool in an action movie. Theme parks like Disney World are also using autonomous robots to enhance the magical experience of their customers.

4. Agriculture

Agriculture is the sector that is the basis of human civilization. However, agriculture is also a seasonal sector that is dependent on ideal weather conditions optimal soil, etc. Moreover, there are many repetitive tasks in agriculture that are just a waste of farmer's time and can be performed more suitable by robots. These include seeding, weed control, harvesting, etc. Robots are usually used for harvesting the crops which allow farmers to be more efficient. An example of a robot that is used to remove weeds in farms is the Ecorobotix. It is powered by solar energy and can be used to target and spray weeds using a complex camera system.

5. Health Care

Robots can help doctors in performing operations more precisely, be used as prosthetic limbs, provide therapy to patients, etc. The possibilities are limitless. One example of this is the **da Vinci robot** that can help surgeons in performing complex surgeries relating to the heart, head, neck, and other sensitive areas. There are other robotic devices that are created like exoskeletons that can be used to provide additional support for people undergoing rehabilitation after spinal injuries, strokes, etc.

6. Underwater Exploration

Robots are a great option for exploring places that humans cannot reach easily, like the depths of the ocean. There is a lot of water pressure deep in the ocean which means humans cannot go that down and machines such as submarines can only go to a certain depth as well. A deep underwater is a mysterious place that can finally be explored using specially designed robots. These robots are remote-controlled, and they can go into depths of the ocean to collect data and images about the aquatic plant and animal life.

7. Food Preparation

There are robots that even can cook and create complete meals for you! These robot chefs can create food using hundreds of different recipes. All humans need to do is choose the recipe they want and provide the robot with pre-packaged containers of all the ingredients that are needed for that recipe. The robot can then cook the food on its own. Moley Robotics is one such robotics company that has created a robotic kitchen with a robot that can cook like a master chef! So no worries if you can't cook food. Because now a robot can!

8. Manufacturing

There are many repetitive and common tasks in the manufacturing industry that don't require any usage of the mind like welding, assembly, packing, etc. These tasks can be easily done by robots while leaving the mentally challenging and creative tasks to humans. These robots can be trained to perform these repetitive and monotonous tasks with precision under the guidance and supervision of a human. This option is also best for the manufacturing processes that are dangerous and may be harmful to humans.

9. Military

Robots also have many applications in the military. They can be used as drones to keep surveillance on the enemy, they can also be used as armed systems to attack the opposing forces or as Medicare agents to help friendly forces. Some of the popular robots used in the Military sector include MAARS (Modular Advanced Armed Robotic System) which looks like a tank and contains tear gas and lasers to confuse enemies and even grenade launcher for desperate situations. DOGO is also a tactical combat robot that has a camera for spying on the activities of the enemy and a 9-millimeter pistol for emergency situations!

10. Customer Service

- There are robots that are developed to look exactly like humans for cosmetic purposes. These robots are primarily used in the field of customer service in high visibility areas to promote robotics. One such example is Nadine, a humanoid robot in Singapore that can recognize people from previous visits, make eye contact, shake hands, continue chatting based on previous meetings, etc. Another such customer service robot is Junko Chihira in Japan, a humanoid robot working at the tourist information center in Aqua City Odaiba, a shopping center on Tokyo's waterfront.
- Robots are used for everything ranging from security guards, chefs, doctor's assistants, customer service agents, and even a one-man army in war! There are many other applications of robots as well because of their precision and programming to perform various tasks that are dangerous, boring, or repetitive to humans. All in all, robots can be the perfect helper for humans and solve many problems in different industries.

6.8 AI applications in Healthcare, Retail, Banking

6.8.1 Healthcare

- When we talk about human lives and health, any technologies that can give more efficient, helpful, and faster analysis to hand out a proper treatment plan in time are tremendously valuable. Artificial Intelligence and its subdivision Machine Learning is taking over the world right now. Speaking of AI in the medical field, we must realize the tremendous potential and the changes Machine Learning can bring to the healthcare industry
- Let's understand how ML can transform both patient care and the administrative processes of the Healthcare industry. Research has already proven that Machine Learning often surpasses humans in the diagnosis of disease. Algorithms are already doing a better job spotting malignant tumors than actual radiologists. Machine Learning for healthcare technologies provides algorithms with self-learning neural networks that are able to increase the quality of treatment by analyzing external data on a patient's condition, their X-rays, CT scans, various tests, and screenings. Also, worth mentioning, deep learning is now largely used for detecting cancer cells.
- One of the best Machine Learning for Healthcare applications is a bot system that makes the treatment period much easier. A virtual nurse for patients acts as a voice-controlled healthcare assistant that provides information on many illnesses, health disorders, and medicines. An AI assistant is a very handy thing if the patient needs real-time advice and it might be difficult for him or her to get to the doctor. Data engineers are working on solutions for all medical activities that deal with overall health monitoring as well as helping cure or even prevent disease.
- There is huge application of Machine Learning in Healthcare algorithms in the fields of Oncology, Pathology and Rare diseases.

Planning Problem

University Question

- What is planning problem? **MU - Dec. 12, Dec. 13**

We have seen in above section what information is available while formulating a planning problem and what results are expected. Also it is understandable here that, states of an agent correspond to the probable surrounding environments while the actions and goals of an agent are specified based on logical formalization.

Also we have learnt about various types of intelligent agents in chapter 2. Which shows that, to achieve any goal an agent has to answer few questions like "what will be the effect of its actions", "how it will affect the upcoming actions", etc. This illustrates that an agent must be able to provide a proper reasoning about its future actions, states of surrounding environments, etc.

Consider simple Tic-Tac-Toe game. A Player cannot win a game in one step, he/she has to follow sequence of actions to win the game. While taking every next step he/she has to consider old steps and has to imagine the probable future actions of an opponent and accordingly make the next move and at the same time he/she should also consider the consequences of his/her actions.

- **Classical planning has the following assumptions about the task environment :**
  - **Fully Observable :** Agent can observe the current state of the environment.
  - **Deterministic :** Agent can determine the consequences of its actions.
  - **Finite :** There are finite set of actions which can be carried out by the agent at every state in order to achieve the goal.
  - **Static :** Events are steady. External event which cannot be handled by agent is not considered.
  - **Discrete :** Events of the agent are distinct from starting step to the ending(goal) state in terms of time.
- So, basically a planning problem finds the sequence of actions to accomplish the goal based on the above assumptions.
- Also note that, goal can be specified as a union of sub-goals.
- Take example of ping pong game where, points are assigned to opponent player when a player fails to return the ball within the rules of ping-pong game. There can a best 3 of 5 matches where, to win a match you have to win 3 games and in every game you have to win with a minimum margin of 2 points.

2.1 Problem Solving and Planning

University Questions

- How planning problem differs from search problem? **MU - Dec. 12, Dec. 13**
- Compare and contrast problem solving agent and planning agent. **MU - Dec. 15, May 16,**

- Generally problem solving and planning methodologies can solve similar type of problems. Main difference between problem solving and planning is that planning is a more open process and agents follow logic-based representation. Planning is supposed to be more powerful than problem solving because of these two reasons.

- Planning agent has situations (i.e. states), goals (i.e. target end conditions) and operations (i.e. actions performed). All these parameters are decomposed into sets of sentences and further in sets words depending on the need of the system.
- Planning agents can deal with situations/states more efficiently because of its explicit reasoning capability also it can communicate with the world. Agents can reflect on their targets and we can minimize the complexity of the planning problem by independently planning for sub-goals of an agent. Agents have information about past actions, presents actions and the important point is that it can predict the effect of actions by inspecting the operations.
- Planning is a logical representation, based on situation, goals and operations, of problem solving.

Planning = Problem solving + Logical representation

5.3 Goal of Planning

- We plan activities in order to achieve some goals. Main goal can be divided into sub-goals to make planning more efficient.
- Take example of a grocery shopping at supermarket, suppose you want to buy milk, bread and egg from supermarket, then your initial state will be – "at home" and goal state will be – "get milk, bread and egg".
- Now if you look at the Fig. 5.3.1 you will understand that branching factor can be enormous depending upon the set of actions, for e.g. Watch TV, read book, etc., available at that point of time.

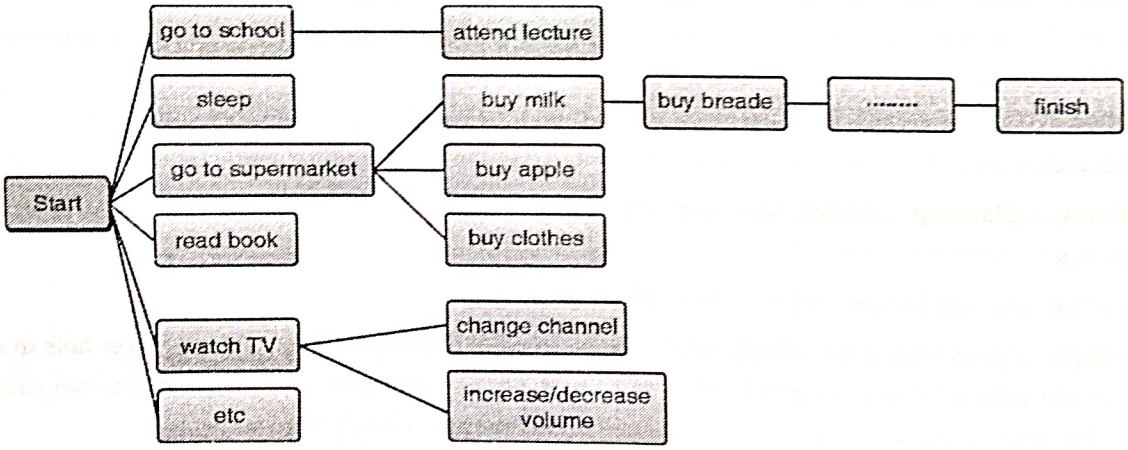


Fig. 5.3.1 : Supermarket examples to understand need of planning

- Thus **branching factor** can be defined as a set of all probable actions at any state, set can be very large, such as in the supermarket example or block problem. If the domain of probable actions increases, the branching factor will also increase as they are directly proportional to each other, this will result in the increase of search space.
- To reach the goal state you have to follow many steps, if you consider using heuristic functions, then you have to remember that it will not be able to eliminate states; these functions will be helpful only for guiding the search of states.
- So it becomes difficult to choose best actions. (i.e. even if we go to supermarket we need to make sure that all three listed items are picked, only then goal state can be achieved).

- As there are many possible actions and it is difficult to describe every state, and there can be combined goals (as seen in supermarket example) searching is inadequate to achieve goals efficiently. In order to be more efficient planning is required.
- In above sections, we discussed that planning requires explicit knowledge that means in case of planning we need to know the exact sequence of actions which will be useful in order to achieve the goal.
- Advantage of planning is that, the order of planning and the order of execution need not be same. For example, you can plan how to pay bills for grocery before planning to go to supermarket.
- Another advantage of planning is that you can make use of divide and conquer policy by dividing /decomposing the goal into sub goals.

5.3.1 Major Approaches

- There are many approaches to solve planning problems. Following are few **major approaches** used for planning :
  - Planning with state space search.
  - Partial ordered planning.
  - Hierarchical planning / hierarchical decomposition (HTN planning).
  - Planning situation calculus/ planning with operators.
  - Conditional planning.
  - Planning with operators.
  - Planning with graphs.
  - Planning with propositional logic.
  - Planning reactive.
- Out of these major approaches we will be learning about following approaches in detail :
  - Planning with state space search
  - Partial ordered planning and
  - Hierarchical planning / Hierarchical decomposition (HTN planning).
  - Conditional planning.
  - Planning with operators.

5.4 Planning Graphs

- Planning graph is a special data structure which is used to get better accuracy. It is a directed graph and is useful to accomplish improved heuristic estimates. Any of the search technique can make use of planning graphs. Also GRAPHPLAN can be used to extract a solution directly.
- Planning graphs work only for propositional problems without variables. You have learnt Gantt charts. Similarly, in case of planning graphs there are series of levels which match to time ladder in the plan. Every level has set of literals and a set of actions. Level 0 is the initial state of planning graph.

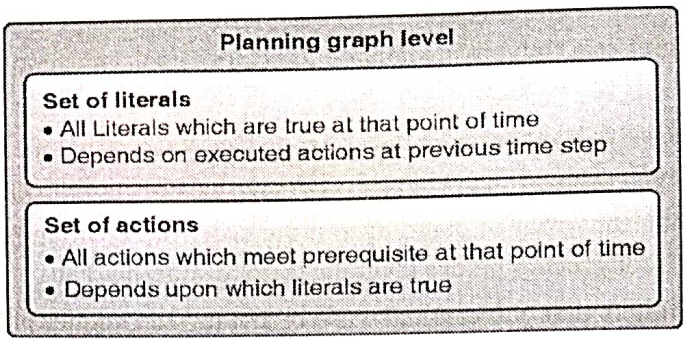


Fig. 5.4.1 : Planning graph level

- Example
  - Init (Have(Apple))
  - Goal(Have(Apple) ∧ Ate(Apple))
  - Action(Eat(Apple), PRECOND : Have(Apple)
  - EFFECT : ¬Have(Apple) ∧ Ate(Apple))
  - Action(Cut(Apple), PRECOND : ¬ Have(Apple)
  - EFFECT : Have(Apple))

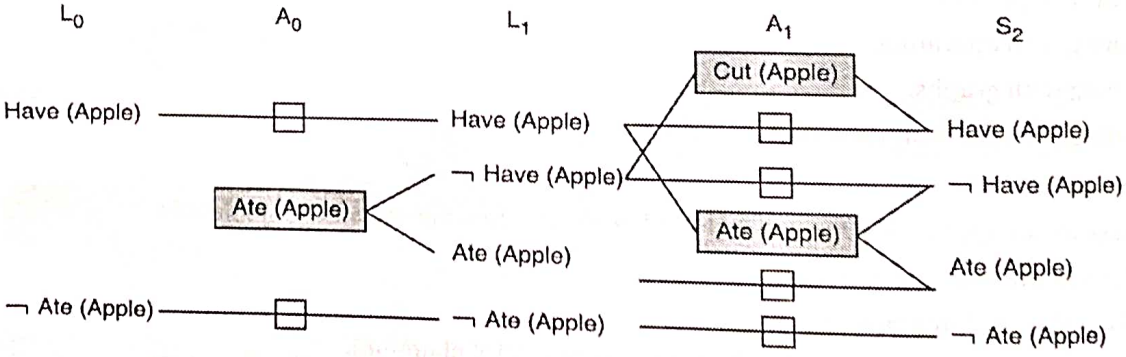


Fig. 5.4.2

- Start at level L0 and determine action level A0 and next level L1.
  - A0 >> all actions whose prerequisite is satisfied at previous level.
  - Connect precondition and effect of actions L0 – > L1.
  - Inaction is represented by persistence actions.
  - Level A0 contains the possible actions.
  - Conflicts between actions are shown by mutual exclusion links.
  - Level L1 contains all literals that could result from picking any subset of actions in Level A0.
  - Conflicts between literals which cannot occur together (as a effect of selection action) are represented by mutual exclusion links.
  - L1 defines multiple states and the mutual exclusion links are the constraints that define this set of states.

- Continue until two consecutive levels are the same or contain the same amount of literals.

A mutual exclusion relation holds between two actions when :	A mutual exclusion relation holds between two literals when :
• One action cancels out the effect of another action. • One of the effects of action is negation of preconditions of other action. • One of the preconditions of one action is mutually exclusion with the precondition of the other action.	• If one literal is the negation of the other literal OR. • If each possible action pair that could achieve the literal is mutually exclusive.

GRAPH PLAN algorithm

GRAPH PLAN can directly extract solution from planning graph with the help of following algorithm :

```
function GRAPHPLAN(problem) return solution or failure
    graph ← INITIAL-PLANNING-GRAPH(problem)
    goals ← GOALS[problem]
    loop do
        if goals all non-mutex in last level of graph then do
            solution ← EXTRACT-SOLUTION(graph, goals,
LENGTH(graph))
            if solution ≠ failure then return solution
            else if NO-SOLUTION-POSSIBLE(graph) then return failure
        graph ← EXPAND-GRAPH(graph, problem)
```

Properties of planning graph

- If goal is absent from last level then goal cannot be achieved!.
- If there exists a path to goal then goal is present in the last level.
- If goal is present in last level then there may not exist any path.

5.5 Planning as State-Space Search

- We have seen example of an agent that can perform three tasks of printing, sending a mail and making coffee namely lets' call this agent as office agent.
- When this office agent gets order from three people at a same time to perform these three different tasks then, let us see how planning with state space search problem will look if we have a finite space.
- You can understand from Fig. 5.5.1 that the office agent is at location 250 on the state space grid. When he gets a task he has to decide which task can be performed more efficiently in lesser time.

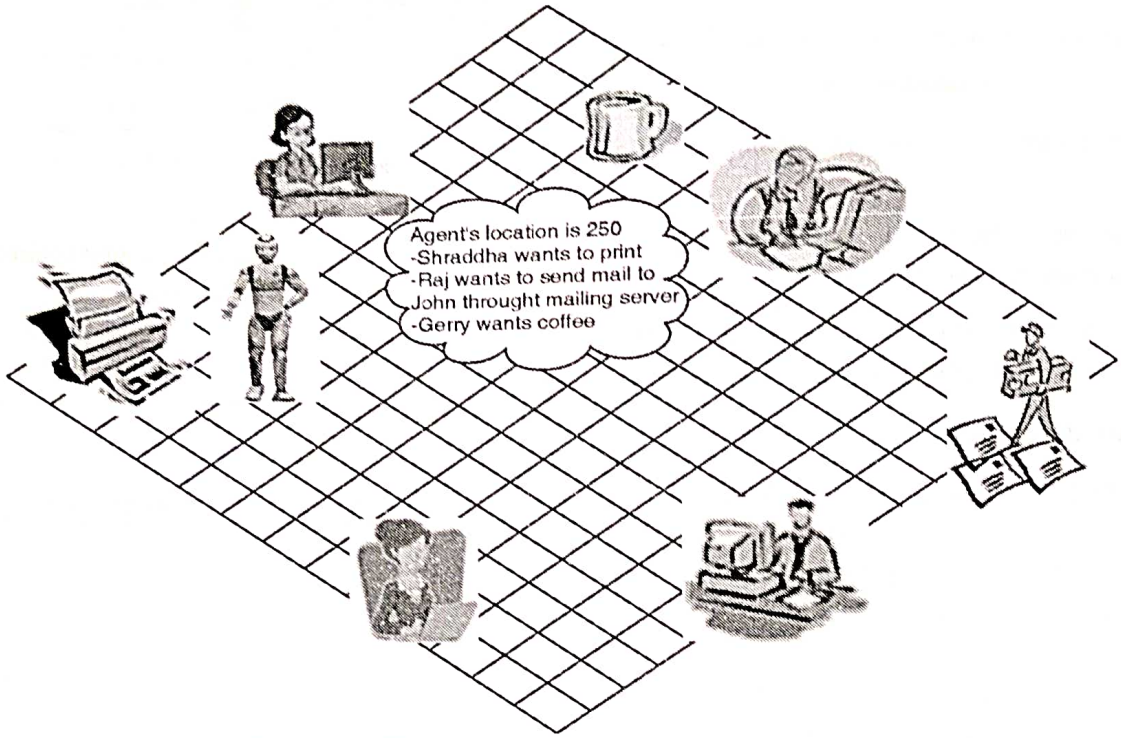


Fig. 5.5.1 : Office agent example with finite state space

- If it finds some input and output locations nearer on state space grid for example in case of printing task then the probability of performing that task will increase.
- But to do this it should be aware of it's own current location, the locations of people who are assigning tasks and the locations of the required devices.
- State space search is unfavourable for solving real-world problems because, it requires complete description of every searched state, also search should be carried out locally.
- There can be two ways of representations for a state :

1. Complete world description
2. Path from an initial state

1. Complete world description

- Description is available in terms of an assignment of a value to each previous suggestion.
- Or we can say that description is available as a suggestion that defines the state.
- Drawback of this types is that it requires a large amount of space.

2. Path from an initial state

- As per the name, path from an initial state gives the sequence of actions which are used to reach a state from an initial state.
- In this case, what holds in a state can be deduced from the axiom which specifies the effects of an actions.

- Drawback of these types is that it does not explicitly specify "What holds in every state". Because of this it can be difficult to determine whether two states are same.

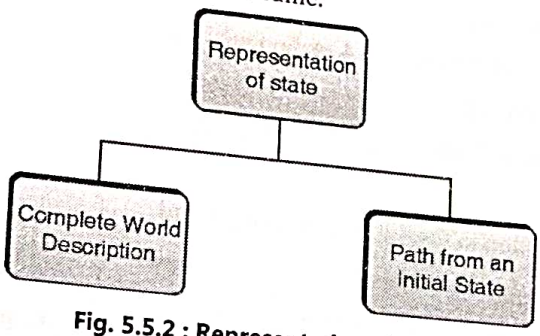


Fig. 5.5.2 : Representation of states

5.5.1 Example of State Space Search

Let us take an example of a water jug problem

- We have two jugs, a 4 - gallon one and a 3- gallon one. Neither has any measuring markers on it. There is a pump that can be used to fill the jugs with water how can you get exact 2 gallons of water into the 4 - gallon jug?
- The state space for this problem can be described as of ordered pairs of integers (x, y), such that x = 0,1,2,3 or 4, representing the number of gallons of water in the 4- gallon jug and y = 0,1,2 or 3, representing the quantity of water in the 3- gallon jug. The start state is (0, 0). The goal state is (2, n) for any value of n, since the problem does not specify how many gallons need to be in the 3- gallon jug.
- The operators to be used to solve the problem can be described as shown bellow. They are represented as rules whose left side are matched against the current state and whose right sides describe the new state that results from applying the rule.

Rule set

1. $(x,y) \longrightarrow (4,y)$ fill the 4- gallon jug
   If $x < 4$
2. $(x,y) \longrightarrow (x,3)$ fill the 3-gallon jug
   If $x < 3$
3. $(x,y) \longrightarrow (x-d,y)$ pour some water out of the 4- gallon jug
   If $x > 0$
4. $(x,y) \longrightarrow (x-d,y)$ pour some water out of the 3- gallon jug
   If $y > 0$
5. $(x,y) \longrightarrow (0,y)$ empty the 4- gallon jug on the ground
   If $x > 0$
6. $(x,y) \longrightarrow (x,0)$ empty the 3- gallon jug on the ground
   If $y > 0$
7. $(x,y) \longrightarrow (4,y-(4-x))$ pour water from the 3- gallon jug into the 4- gallon
   If $x + y >= 4$ and $y > 0$ jug until the 4-galoon jug is full
8. $(x,y) \longrightarrow (x-(3-y),3))$ pour water from the 4- gallon jug into the 3-gallon
   If $x + y >= 3$ and $x > 0$ jug until the 3-gallon jug is full

9. (x,y) ⟶ (x+y,0) pour all the water from the 3 -gallon jug into
   If x + y <= 4 and y > 0 the 3-gallon jug
10. (x,y) ⟶ (0,x+y) pour all the water from the 4 -gallon jug into
    If x + y <= 3 and x > 0 the 3-gallon jug
11. (0,2) ⟶ (2,0) pour the 2-gallon from the 3 -gallon jug into
    the 4-gallon jug
12. (2,y) ⟶ (0,x) empty the 2 gallon in the 4 gallon on the ground

Production for the water jug problem

Gallons in the 4- gallon Jug	Gallons in the 3- gallon	Rule Applied
0	0	
0	3	2
3	0	9
3	3	2
4	2	7
0	2	5 or 12
2	0	9 or 11

One solution to the water jug problem.

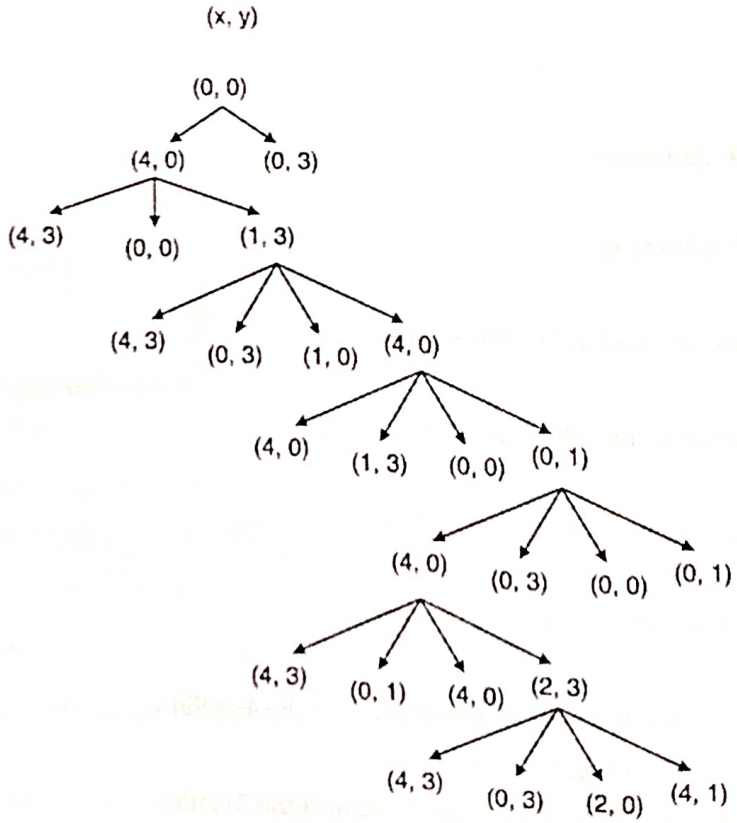
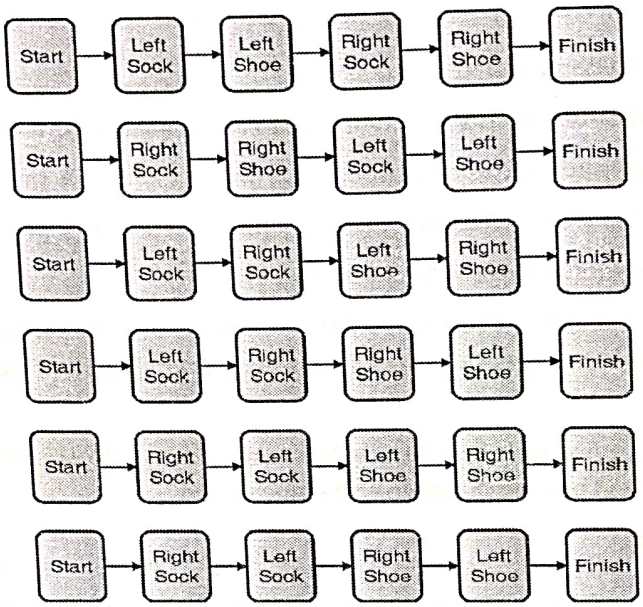


Fig. 5.5.3



Fig, 5.6.1 : Total order planning of Wearing Shoe

5.7 Partial Order Planning

University Questions

Q. Discuss partial order planning giving suitable example. **MU - May 15, Dec. 15**

Q. Compare partial order planning with conditional planning. **MU - Dec. 19**

- In case of **Partial Ordered Planning (POP)**, ordering of the actions is partial. Also partial ordered planning does not specify which action will come first out of the two actions which are placed in plan.
- With partial ordered planning, problem can be decomposed, so it can work well in case the environment is non-cooperative.
- Take same example of wearing shoe to understand partial ordered planning.

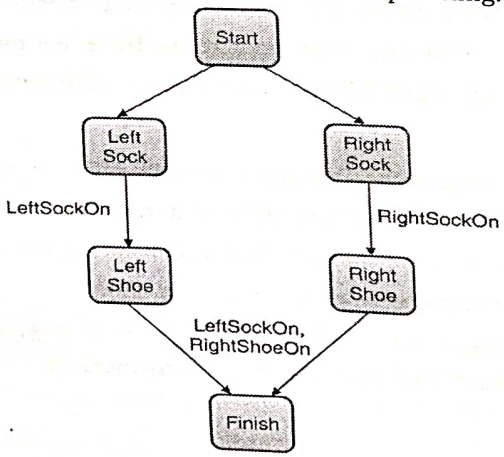


Fig. 5.7.1 : Partial order planning of Wearing Shoe

- **A partial order planning combines two action sequences**
  - First branch covers left-sock and left-shoe.
  - In this case to wear a left shoe, wearing left sock is the precondition, similarly.
  - Second branch covers right-sock and right-shoe.
  - Here, wearing a right sock is the precondition for wearing the right shoe.
- Once these actions are taken we achieve our goal and reach the finish state.

5.7.1 POP as a Search Problem

University Question

Q. Define partial order planner. **MU - May 13, Dec. 14**

- If we considered POP as a search problem, then we say that states are small plans.
- States are generally unfinished actions. If we take an empty plan then, it will consist of only starting and finishing actions.
- Every plan has four main components, which can be given as follows :

Set of actions

- These are the steps of a plan. Actions which can be performed in order to achieve goal are stored in set of actions component.
- **For example :** Set of Actions = { Start, Rightsock, Rightshoe, Leftsock, Leftshoe, Finish}
- Here, wearing left sock, wearing left shoe, wearing right sock, wearing right shoe are set of actions.

2. Set of ordering constraints/ preconditions

- Preconditions are considered as ordering constraints. (i.e. without performing action "x" we cannot perform action "y")
- For example : Set of Ordering ={Right-sock < Right-shoe; Left-sock < Left-shoe} that is In order to wear shoe, first we should wear a sock.
- So the ordering constraint can be Wear Left-sock < wear Left-shoe (Wearing Left-sock action should be taken before wearing Left-shoe) Or Wear right-sock < wear right-shoe (Wearing right-sock action should be taken before wearing right-shoe).
- If constraints are cyclic then it represents inconsistency.
- If we want to have a consistent plan then there should not be any cycle of preconditions.

3. Set of causal links

- Action A achieves effect "E" for action B

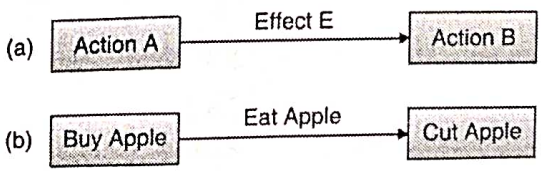


Fig. 5.7.2 : (a) Causal Link Partial Order Planning (b) Causal Link Example

- From Fig. 5.7.2(b) you can understand that if you buy an apple it's effect can be eating an apple and the precondition of eating an apple is cutting apple.
- There can be conflict if there is an action C that has an effect ¬ E and, according to the ordering constraints it comes after action A and before action B.
- Say we don't want to eat an apple instead of that we want to make a decorative apple swan. This action can be between A and B and It does not have effect "E".
  - **For example :** Set of Causal Links = {Right-sock->Right-sock-on → Right-shoe, Leftsock → Leftsockon → Leftshoe, Rightshoe → Rightshoeon → Finish, leftshoe → leftshoeon → Finish }.
  - To have a consistent plan there should not be any conflicts with the causal links.

4. Set of open preconditions

- Preconditions are called open if it cannot be achieved by some actions in the plan. Least commitment strategy can be used by delaying the choice during search.
- To have a consistent plan there should not be any open precondition

5.7.2 Consistent Plan is a Solution for POP Problem

- As consistent plan does not have cycle of constraints; it does not have conflicts in the causal links and does not have open preconditions so it can provide a solution for POP problem.
- While solving POP problem operators can add links and steps from existing plans to open preconditions in order to fulfil the open preconditions and then steps can be ordered with respect to the other steps for removing the potential conflicts. If the open precondition is unattainable, then backtrack the steps and try solving the problem with POP.
- Partial ordered planning is a more efficient method, because with the help of POP we can progress from vague plan to complete and correct solution in a faster way. Also we can solve a huge state space plan in less number of steps, this is because search takes place only when sub-plans interact.

5.8 Hierarchical Planning

University Question

Q. Explain the real time application of hierarchical planning. **MU - Dec. 19**

- Hierarchical planning is also called as plan decomposition. Generally plans are organized in a hierarchical format.
- Complex actions can be decomposed into more primitive actions and it can be denoted with the help of links between various states at different levels of the hierarchy. This is called as operator expansion.
- **For example**

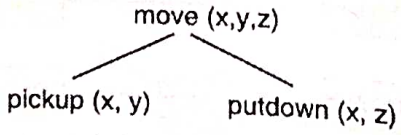


Fig. 5.8.1 : Operator expansion