NAME :- DUBEY KARAN SANJEEV

CLASS :- B.E - 4

ROLL NO :- 04

BATCH :- A.

SUBJECT :- AISC ASSIGNMENT - 2

Q.1. Consider the following axioms.

(i) All hounds howl at night.

(ii) Anyone who has any cat will not have any mice.

(iii) light sleepers do not have anything which howls at night.

(iv) John has either a cat or a hound.

(a) Predicate logic statement.

(b) Clausal Normal form.

(c) Apply resolution technique.

Ans

(a) The first step is to write each sentence as a well formed formula in first-order predicate logic calculus. The formulae written for the above axioms are shown below, using predicate LS(x) to represent the property 'light sleeper'.

(i) All hound howl at night.

$\forall x\,(Hound(x) \rightarrow Howl(x))$

(ii) Anyone who has any cats will not have any mice

$\forall x \forall y\,(Have(x,y) \wedge Cat(y) \rightarrow \neg \exists z\,(Have(x,z) \wedge Mouse(z)))$.

(iii) Light sleepers do not have anything which howls at night

$\forall x\,(LS(x) \rightarrow \neg \exists y\,(Have(x,y) \wedge Howl(y)))$

(iv) John has either a cat or hound.

$\exists x\,(Have(John, x) \wedge (Cat(x) \vee Hound(x)))$

$LS(John) \rightarrow \neg \exists z\,(Have(John, z) \wedge Mouse(z))$.

(b) The next step is to transform each wff into Prenex Normal form, skolemize and rewrite as clause in conjuctive normal form (CNF).

Below we show these transformations for each first order formula.

(i) $\forall x \, (Hound(x) \rightarrow Howl(x))$

$\neg Hound(x) \vee Howl(x)$

(ii) $\forall x \forall y \, (Have(x,y) \wedge Cat(y) \rightarrow \neg \exists z \, (Have(x,z) \wedge Mouse(z)))$.

$\forall x \forall y \, (Have(x,y) \wedge Cat(y) \rightarrow \forall z \, \neg (Have(x,z) \wedge Mouse(z)))$.

$\forall x \forall y \forall z \, (\neg (Have(x,y) \wedge Cat(y)) \vee \neg (Have(x,z) \wedge Mouse(z)))$.

$\neg Have(x,y) \vee \neg (Cat(y)) \vee \neg Have(x,z) \vee \neg Mouse(z)$.

(iii) $\forall x \, (LS(x) \rightarrow \neg \exists y \, (Have(x,y) \wedge Howl(y)))$.

$\forall x \, (LS(x) \rightarrow \forall y \, \neg (Have(x,y) \wedge Howl(y)))$.

$\forall x \forall y \, (LS(x) \rightarrow \neg Have(x,y) \vee \neg Howl(y))$

$\forall x \forall y \, (LS(x) \vee \neg Have(x,y) \vee \neg Howl(y))$.

$LS(x) \vee \neg Have(x,y) \vee \neg Howl(y)$.

(iv) $\exists x \, (Have(John, x) \wedge (Cat(x) \vee Hound(x)))$.

$Have(John, a) \wedge (Cat(a) \vee Hound(a))$.

$\neg [LS(John) \rightarrow \neg \exists z \, (Have(John, z) \wedge Mouse(z))]$

(negated conclusion).

$\neg [LS(John) \vee \neg \exists z \, (Have(John, z) \wedge Mouse(z))]$

$LS(John) \wedge \exists z \, (Have(John, z) \wedge Mouse(z))]$

$LS(John) \wedge Have(John, b) \wedge Mouse(b)$.

The set of CNF clauses for this problem is thus as follows:-

(i) $\neg Hound(x) \vee Howl(x)$

(ii) $\neg Have(x,y) \vee \neg Cat(y) \vee \neg Have(x,z) \vee \neg Mouse(z)$

(iii) $\neg LS(x) \vee \neg Have(x,y) \vee \neg Howl(y)$

(iv) (a) $Have(John, a)$

(b) $Cat(a) \vee Hound(a)$.

(a) $LS(John)$

(b) $Have(John, b)$

(c) $Mouse(b)$.

(c) Now we proceed to prove the conclusion by resolution using the above clauses. Each result clause is numbered the number of its parent clause are shown in at the right hand side.

(vi) Cat (a) ∨ Howl (a) [1, 4(b)]

(vii) ¬ Have (x,y) ∨ ¬ Cat ∨ ¬ Have (x, b) [2, 5(c)]

(viii) ¬ Have (John , x) ∨ ¬ Cat (y) [7, 5(b)]

(ix) ¬ Have (John, a) ∨ Howl (a) [6, 8]

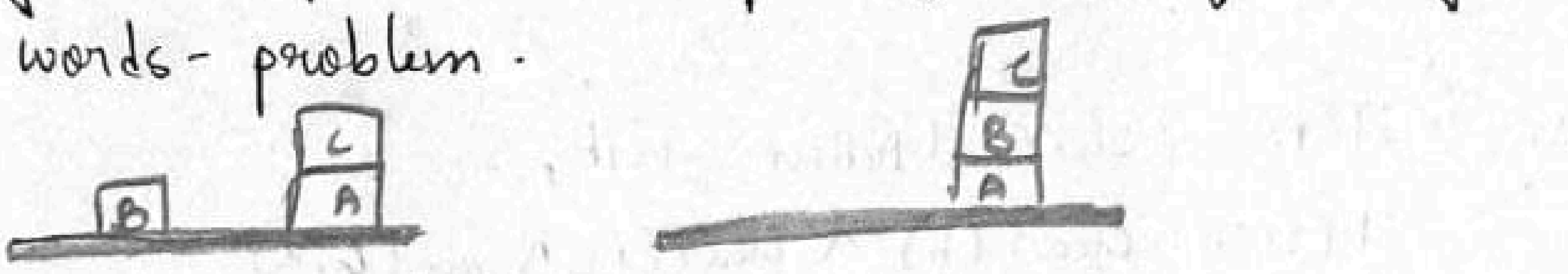
(x) Howl (a) [4(a), 9]

(xi) ¬ LS(X) ∨ ¬ Have (x, a) [4(a), 11]

(xii) ¬ LS (John) [5(a), 12]

(xiii) □

Q2. Give the partial order plan for the following block-words-problem.



(i) Give initial and goal state description.
(ii) Provide definition of each discription.
(iii) Define the operators.
(iv) Create a sample plan.

Ans

Partial - Order planner (POP) is a regression planner, it uses problem decomposition, it searches plan space rather than the state space

A plan in POP (whether it be a finished one or an unfinished one) comprises.

↳ A set of plan steps. Each of these is a STRIPS operator

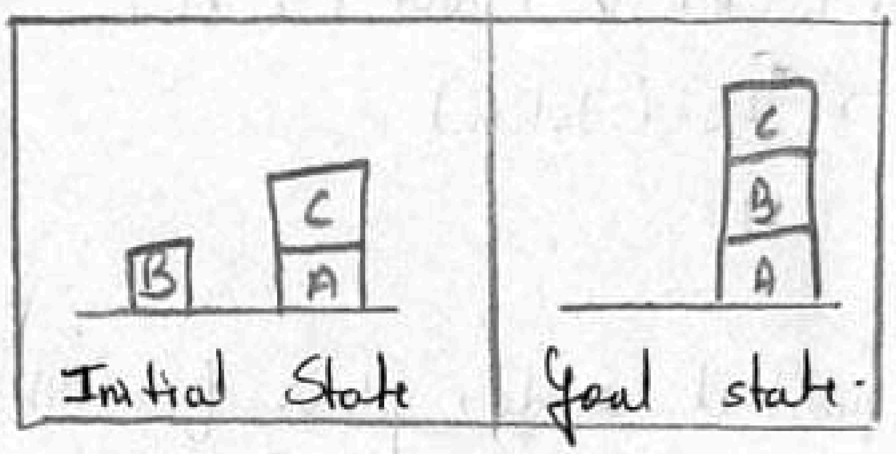
↳ A set of ordering constants: $S_i < S_j$ means step S_i occur sometimes before S_j.

↳ A set of casual link $S_i \xrightarrow{c} S_j$ means step S_i achieves precondition c of step S_j.

So, it comprises action (steps) with constraints (for ordering and causalities) on them.

The algorithm need to start off with an initial plan. This is an unfinished plan, which we will refine until we reach a solution plan.

Start is a step with no precondition, only effects: the effects are the initial state of the world, finish is a step with no effects only precondition: the precondition are the goal.



Plan (STEPS: { S1 : Op (Action : Start,
EFFECT : Clear (b) ∧ Clear (c) ∧ on (c, a)
∧ ontable (a) ∧ ontable (b) ∧ armempty)

S2 : Op (ACTION : Finish,
PRECOND : on (c, b) ∧ on (a, c))},

ORDERINGS : { S1 -< S2 },

LINKS : { }).

This initial plan is refined using POP's plan refinement operators. As we apply them, they will take us from an unfinished plan to a less and less unfinished plan.

Goal achievement operator.

1. Step addition :- Add a new step si which has an effect c that can achieve an as yet unachieved precondition.
Constraint :- Si -< Sj and = " " Si = " ", C = " "
Sj = " " and = " " start = " " " -< Si

=" " –< =" "

finish, <=" " L_i=" ">.

2. Use an effect c of an existing step si to achieve an as yet ~~yet~~ unachieve precondition.

Constraints: $s_i \prec s_j$ and $s_i \xrightarrow{c} s_j$.

Casual link must be protected from threats, i.e steps that delete (or negate or clobber) the protected condition. If s threatens link $s_i \xrightarrow{c} s_j$;

1. Promote: add the constraint $s \prec s_i$, or
2. Demote: add the constraint $s_j \prec s$.

The goal achievement operators ought to be obvious enough. They find preconditions of steps in the unfinished-plan that are not yet achieved.

The promotion and demotion operations may be less clear. Why are these needed? POP uses problem-decomposition: faced with a conjuctive precondition it uses goal achievement on each conjuct separately.

Finally, we have to be able to recognise when we have reached a solution plan: a finished plan.

A solution plan is one in which

- every precondition of every step is achieved by the effect of some other step.
- There are no contradictions in the ordering constraint eg. disallowed is $s_i \prec s_j$ and $s_j \prec s_i$ also =" " disallowed =" " is =" " s_i =" " –< s_j =" " =" " –< =" " s_k =" " –< =" " s_i, <=" " L_i=" ">

Note that solution may still be partially ordered. This retains flexibility for as long as possible. Only immediately prior to execution will the plan need linearisation.

If there's a single agent but if it is capable of multitasking, then some linearisation can be avoided steps can be carried out in parallel.

Q. 3. Design a fuzzy controller for a train approaching a station. The i/p are distance from station and speed of the train. The o/p is brake power used.

Use:-

1. Triangular membership function
2. Four descriptor for each variable.
3. Appropriate defuzzification method.

Ans

Step 1:- Identifying input and output variable along with linguistic description.

Input:-

Speed {S, F} (0-100%)

S - Slow

F - Fast

Distance {C, F} (0-100 feet)

C - Close

F - Far.

Output:-

Brake Power {L, M, H} (0-100%)

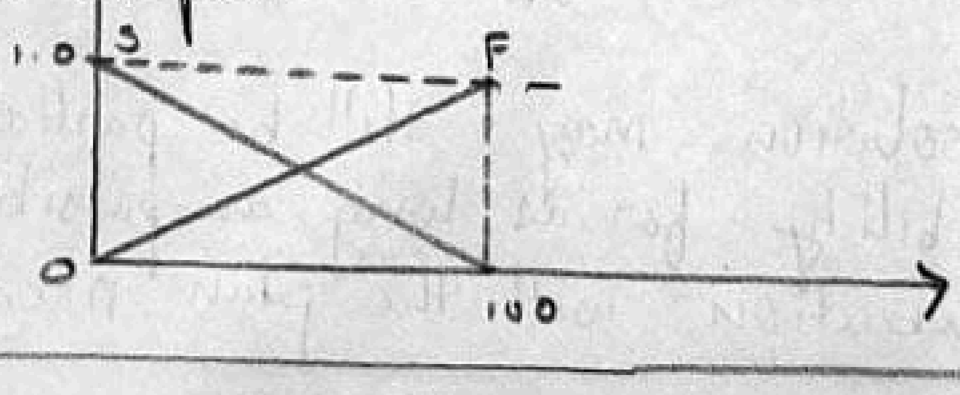
L - Light

M - Medium

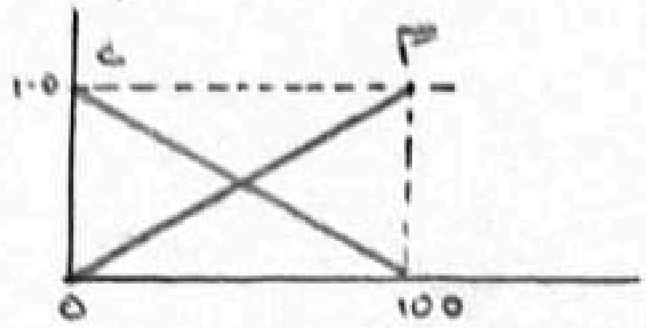
H - Heavy

The input and output variable can be plotted as follows:-

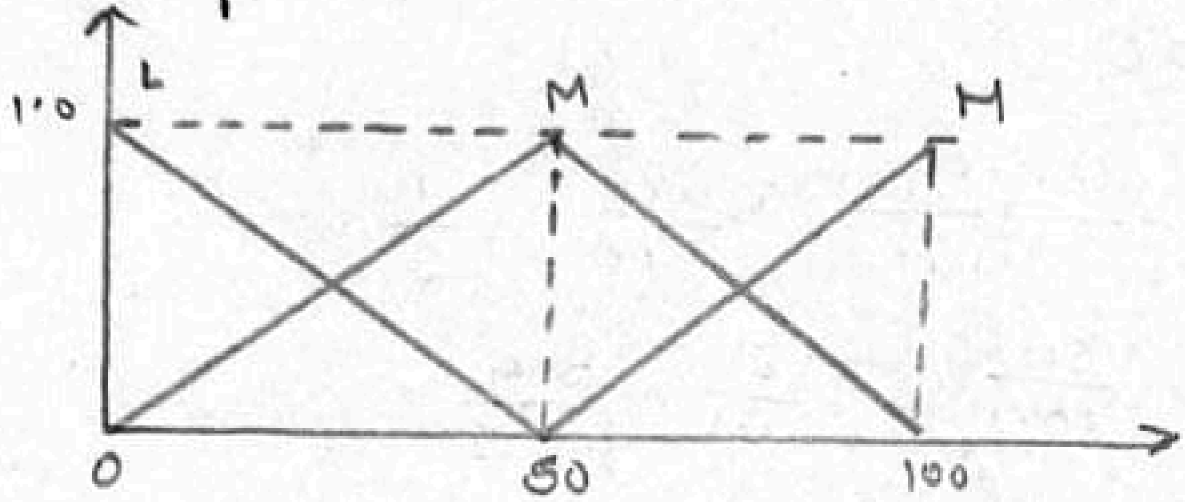
Input Speed.



Input : Distance



Output : Break Power.



Step 2 : Assign each membership value for each input and output variable.

$$\mu_S(x) = \begin{cases} \mu_S(x) = \dfrac{100-x}{100} & , 0 <= x <= 100 \\ \mu_F(x) = \dfrac{x}{100} & , 0 <= x <= 100. \end{cases}$$

$$\mu_D(y) = \begin{cases} \mu_C(y) = \dfrac{100-y}{100} & , 0 <= y <= 100 \\ \mu_F(y) = \dfrac{y}{100} & , 0 <= y <= 100 \end{cases}$$

$$\mu_{BP}(z) = \begin{cases} \mu_L(z) = \dfrac{50-z}{50} & , 0 <= z <= 50 \\ \mu_M(z) = \begin{cases} \dfrac{z}{50} & , 0 <= z <= 50 \\ \dfrac{100-z}{50} & 50 <= z <= 100 \end{cases} \\ \mu_H(z) = \dfrac{z-50}{50} & , 50 <= z <= 100 \end{cases}$$

Step 3

3.1 Build Rule Base.

x/y	C	F
S	L	L
F	H	M

3.2 Rule Evaluation

Speed = 80

Distance = 20

$N_S(80) = \frac{100-80}{100} = \frac{20}{100} = \frac{1}{5}$

$N_F(80) = \frac{80}{100} = \frac{8}{10} = \frac{4}{5}$

$N_C(20) = \frac{100-20}{50\cancel{100}} = \frac{80}{50} = \frac{8}{5}$

$N_F(20) = \frac{20}{100} = \frac{2}{10} = \frac{1}{5}$

3.3 Rule Decision Table.

	$N_C(y)$	$N_F(y)$
$N_S(x)$	$N_L(z)$	$N_L(z)$
$N_F(x)$	$N_H(z)$	$N_M(z)$

Step 4: Defuzzification.

4.1 Min-Max method.

$N_S(80) \cap N_C(20) = \frac{1}{5} \cap \frac{8}{5} = \frac{1}{5}$

$N_S(80) \cap N_F(20) = \frac{1}{5} \cap \frac{1}{5} = \frac{1}{5}$

$N_F(80) \cap N_C(20) = \frac{4}{5} \cap \frac{8}{5} = \frac{4}{5}$

$N_F(80) \cap N_F(\cancel{80}) = \frac{4}{5} \cap \frac{1}{5} = \frac{1}{5}$

$\text{Max}\left(\frac{1}{5}, \frac{1}{5}, \frac{4}{5}, \frac{1}{5}\right) = \frac{4}{5}$

4.2 Rule Strength Table

	$\mu_c(y)$	$\mu_F(y)$
$\mu_S(x)$	1/5	1/5
$\mu_F(x)$	4/5	1/6

4.3 Mapping RST with RDT

$$\mu_H(z) = \frac{z-50}{50}$$

$$\frac{4}{5} = \frac{z-50}{50}$$

$$\frac{4 \times 50}{5 \times \cancel{50}} = z-50$$

$$40 = z-50$$

$$z = 90\%.$$

∴ 90 % break power is required when train speed is 80 and distance is 20, means train is fast and distance is close then obviously break power will be more-

Q.4. Explain an application of ANFIS.

Ans

ANFIS :- An Adaptive Neuro - Fuzzy interference system or Adaptive network - Based Fuzzy System (ANFIS) is a kind of artificial neural networks that is based on Takagi - Sugeno fuzzy interference system. Since it integrates both neural networks and fuzzy logic principles, it has potential to capture the benefits of both in a single frameworks.

Its inference system corresponds to a set of fuzzy If - Then rule that have learning capability.

Representing Mandani Fuzzy Model:

1. For the Madmdani fuzzy interference system with max-min composition, a corresponding ANFIS can be constructed if discrete approximation are use to replace the integrals in the centroid.
2. However, the resulting ANFIS is much more complicated than either TS ANFIS or Tsukamoto ANFIS. The extra complexity in structure and computation of Mamdani ANFIS with max-min composition.
3. If we adopt sum-product composition and centroid defuzzification for a mamdani fuzzy models, a corresponding ANFIS can be constructed. ~~easily~~ based on Theorem.

Applications.

1. ANFIS controller is widely use for controlling, the non-linear system.
2. As this is the best controller as compared to conventional PID controller, and other controller.
3. This controller is used in Temperature water bath controller.
4. Also this controller is used in planes to control them now a days research is going on for Intelligent planes which learn by themselves and do false take off and landing so ~~that~~ there are the applications.