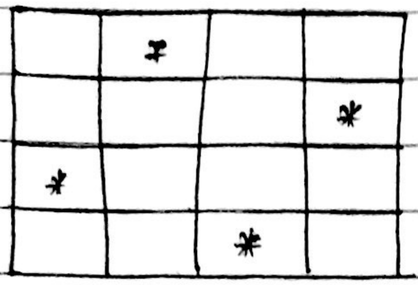
Date

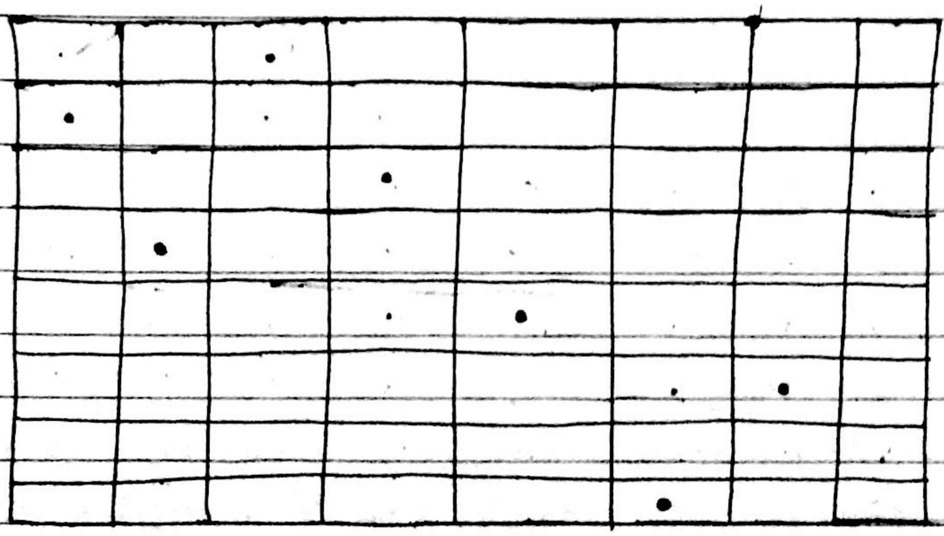
Page

A I N N

8 queen problem.



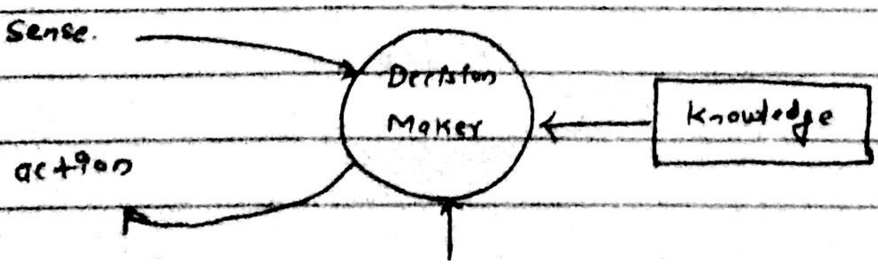
4 queen problems.



8 queen problems.

fox → goose → grain

Knowledge Representation



Chapter : 2

Intelligence:

Ability ~~of~~ a system to calculate, reason, perceive relationships.

Types

- Linguistic
- Musical
- Logical - Mathematical.
- Spatial
- Bodily - Kinesthetic
- ~~In~~ Intra-personal
- Inter-personal.

propositional logic :-

It is the simpler form of a logic. Proposition is a statement. for example, Ram is intelligent, Ram is hardworking, If Ram is intelligence and hardworking then Ram scores high marks.

intelligent(Ram)

hardworking(Ram)

Syntax :-

let, P stands for intelligent(Ram)

q stands for hardworking(Ram)

$P \wedge Q$ and $P \vee Q$ are called compound proposition.

Syntatic elements of propositional logic

- Vocabulary :-

It is a set of propositional symbols(p, q, r, etc). which can be true or false.

- Set of logical operators :-

  AND($\wedge$), OR($\vee$), NOT($\neg$), implies($\rightarrow$)

- Parenthesis :-

  Parenthesis is used for grouping.

  Two special symbols : T and F.

<u>How to form propositional Sentences?</u>

Each symbol (a proposition or a constant) is a sentence.

If (P) is a sentence and (Q) is a sentence, then

1> (P), (Q) is a sentence

2> $P \wedge Q$, $P \vee Q$ is a sentence.

3> $\neg P$, $\neg Q$ is a sentence.

4> $P \rightarrow Q$, $Q \rightarrow P$ is a sentence.

Sentence is also known as well formed formula. (WWF)

<u>Implication</u>

$$p \rightarrow Q$$

if p is true then Q is true.

<u>Bijection or Equivalence</u>

$$p \leftrightarrow Q \quad \text{or} \quad p \Leftrightarrow Q$$

e.g. If two sides of a triangle is equal then two base angles of the triangle are equal.

$$P \rightarrow Q$$

$$Q \rightarrow P$$

p : two sides of a triangle is equal.

q : two base angles of a triangle is equal

then, $P \leftrightarrow Q$

$p \rightarrow Q$

$Q \rightarrow P$

Date

Page

How do we get the meaning ?

→ A sentence can be compound by using connective like and, or, ~~not~~ implies, etc.

→ ~~Interpeal~~ Interpret each atomic proposition in the same world.

→ Assign ~~truth~~ values to each interpretation.

→ Compute the truth value of compound proposition.

p: likes (Sumit, Sunil).

q : knows (Jyoti, Sudhir).

World: Sumit and Sunil are friends and Jyoti and Sudhir are known to eachother.

$$p \land q = T$$

$$p \land (\neg q) = F$$

Validity of a sentence

If a propositional sentence always evaluates to true under all possible interpretation, it is valid.

Tautology

$$p \lor (\neg p)$$

Quiz :-

Express the following English language statement in the language of propositional logic

① It rains in July.

② The book is not costly.

③ If it rains today and one doesnot carry an umbrella, ye he will be drenched.

- Rains (July)
- ¬ costly (Book)
- Rains(today) ∧ ¬ carry (Ram, umbrella) → getdrenched (Ram)

¬ carry (Ram, umbrella).

<u>Quiz 2 :-</u>

If P is true and Q is true, then are ~~the~~ following sentences are true or false

1. $P \rightarrow Q$ T.
2. $(\neg P) \vee Q \rightarrow Q$.   T.
3. $(\neg P) \vee Q \rightarrow P$.   T.
4. $P \vee (\neg P) \rightarrow T$.   T

<u>Truth Table :-</u>

P	Q	$P \rightarrow Q$
T	T	T
T	F	F
F	T	F
F	F	T

<u>De-morgan's Theorem</u>

$\neg(P \wedge Q) = \neg P \vee \neg Q$

$\neg(P \vee Q) = \neg P \wedge \neg Q$.

(I) $\neg(P \wedge Q) = \neg P \vee \neg Q$.

P	Q	$\neg P$	$\neg Q$	$P \wedge Q$	$\neg(P \wedge Q)$	$\neg P \vee \neg Q$
T	T	F	F	T	F	F
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	F	T	T	F	T	T

$\neg(P \vee Q) = \neg P \wedge \neg Q$

P	Q	$P \vee Q$	$\neg(P \vee Q)$	$\neg P$	$\neg Q$	$\neg P \wedge \neg Q$
T	T	T	F	F	F	F
T	F	T	F	F	T	F
F	T	T	F	T	F	F
F	F	F	T	T	T	T

Reasoning in terms of propositional logic

P : It is the month of July.
Q : It rains.
R : $P \rightarrow Q$ [Rule]
If it is the month of July then it rains.

Given,
It is the month of July conclude it rains.

Modus Pones (Inference Rule)

$$\frac{P \rightarrow Q \;,\; P}{Q}$$

$$\boxed{P \rightarrow Q \Rightarrow \neg P \vee Q}$$

$P \rightarrow Q \Rightarrow \neg P \vee Q$
Conjoining with P.
$P \wedge (\neg P \vee Q)$
$((P \wedge \neg P) \vee Q)$
$F \vee Q$.
Q.//

Modus pones is an inference rule that allows us to deduce the truth of a consequent depending on the truth of the

Date
Page

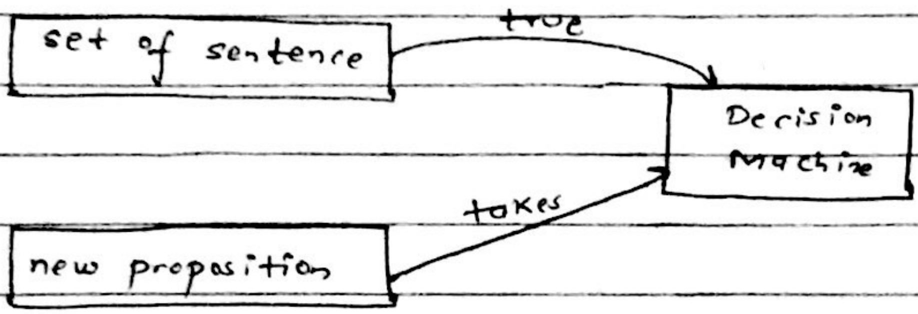
antecedent.

Satisfiability

A sentence is satisfiable by an interpretation if under that interpretation the sentence evaluates to true. If no interpretation makes all the sentences in the set to be true then the set of sentences is unsatisfiable or inconsistence.

Entailment ($\models$)

$P \models Q$, Q logically follows P.



Resolution

Clause :-

Clause is a special form of a Sentence. Basically, not and or are only used. It is a disjunction of literals.

morgan's law).

$(A \wedge \neg B) \vee (\neg C \wedge A)$.

III) Convert to conjuctive normal form by using distributive and associative laws.

$= (A \vee (\neg C \vee A)) \wedge (\neg B \vee \neg C \vee A)$

$= (A \vee \neg C) \wedge (\neg B \vee \neg C \vee A)$

IV) get set of clauses.

$(A \vee \neg C)$ $(\neg B \vee \neg C \vee A)$

clausal form allows us to use Resolution.

Resolution by Refutation

~~Pd((prolchlls~~

Procedure ~~of~~ for Resolution:-

I> Convert the given proposition into clausal form.

II> Convert the negation of sentence to be proved. into clausal form.

III> Combine the clause into set.

IV> Iteratively apply resolution to the set and add ~~resolved~~ resolvent to the set.

V> Continue until no further resolvents can be obtained or a null clause is obtained.

~~VI)~~

Q. ~~One mammal drinks milk~~

1. Mammals drinks milk.
2. Man is mortal
3. Man is a mammal.
4. Tom is a Man.

prove :- A> Tom drinks milk.

Date
Page

8) Tom is mortal.

A : Mammals drinks milk
B : Man is mortal
C : Man is mammal
D : Tom is a Man.

mammal (Tom) $\rightarrow$ drinks (Tom, milk) P
Man (Tom) $\rightarrow$ mortal (Tom) Q
Man (Tom) $\rightarrow$ mammal (Tom). R
man (Tom) S.

mammal (Tom) $\rightarrow$ drinks (Tom, milk)
conjoing with mammal (Tom)
mammal (Tom) $\wedge$ ($\neg$ mammal (Tom) $\vee$ drinks (Tom, milk))
((mammal (Tom) $\wedge$ $\neg$ mammal (Tom) $\vee$ drinks (Tom, milk))
drinks (Tom, milk).

man (Tom) $\rightarrow$ mortal (Tom)
conjoing with man (Tom)
man (Tom) $\wedge$ ($\neg$ man (Tom) $\vee$ ~~drinks~~ mortal (Tom))
((man (Tom) $\wedge$ $\neg$ man (Tom) $\vee$ mortal (Tom))
mortal(Tom)

mammal (Tom) $\rightarrow$ drinks (Mammal, milk) — P
Man (Tom) $\rightarrow$ mortal (Man) $\Rightarrow$ Man (Tom) $\rightarrow$ mortal (Tom). — Q
Man (Tom) $\rightarrow$ mammal (Man) $\Rightarrow$ Man (Tom) $\rightarrow$ mammal (Tom). $\rightarrow$ R
Man (Tom). — S

I. Applying modus ponens on R and S, we get
mammal (Tom) — 5

II. Applying modus ponens on 5 and P.
drinks (Tom, milk).

Date
Page

Applying modous ponen's on g and s
mortal(Tom).

Converting to clausal form,
mammal(Tom) → drinks(Tom, milk)

(I) Eliminate implication sign
¬mammal(Tom) ∨ drinks(Tom, milk)
¬man(Tom) ∨ mortal(Tom)
¬man(Tom) ∨ mammal(Tom)
man(Tom)

Q. Number of propositions
1. If a triangle is equilateral then it is isoceles.
2. If a triangle is isoceles, then two sides AB and AC are equal.
3. If AB and AC is equal, then angle B and angle C are equal.
4. Triangle ABC is an equalateral triangle.

prove :-
∠B = ∠C.

Solution :-
Steps :-
I> Converting the given propositions into clausal form as follows:-

equilateral (Triangle) ⟶ isoceles(Triangle) — (1)
isoceles (Triangle) ⟶ equal (Triangle, two sides AB and AC)
equal (Triangle, two sides AB and AC) ⟶ equal(Triangle, two angles B and C)

equilateral (Triangle ABC) — (4)

Date	
Page	

II> Applying modus ponen's on ① and ④, we get
isoceles (Triangle) — ⑤

III> Applying modus ponen's on ⑤ and ②
equal (Triangle, twosides AB and BC) — ⑥

IV> Applying modus ponen's on ③ and ⑥
equal (Triangle, two angles B and C).

Hence, we get
Two angles B and C are equal
i.e $\angle B = \angle C$.

① equilateral (ΔABC) $\longrightarrow$ isoceles (ΔABC)
② isoceles (ΔABC) $\longrightarrow$ equal (AB, AC)
③ equal (AB, AC) $\longrightarrow$ equal ($\angle B, \angle C$)
④ equilateral (ΔABC)
⑤ equal ($\angle B, \angle C$) — prove.

Converting these sentences into clausal form:
$\neg$equilateral (ΔABC) $\vee$ isoceles (ΔABC)
$\neg$ isoceles (ΔABC) $\vee$ equal (AB, AC)
$\neg$equal (AB, AC) $\vee$ equal ($\angle B, \angle C$)
equilateral (ΔABC)

—②

—③ let us disprove ⑤
$\neg$equal ($\angle B, \angle C$)

$\neg$ equal (AB, AC) — 7

from 7 and 2

$\neg$ equal (AB, AC) $\vee$ $\neg$ isoceles (ΔABC) $\vee$ equal (AB, AC)

$\neg$ isoceles (ΔABC) — 8

from 8 and 1

$\neg$ isoceles (ΔABC) $\vee$ $\neg$ equilateral (ΔABC) $\vee$ isoceles (ΔABC)

$\neg$ equilateral (ΔABC) — 9

from 9 and 4

$\neg$ equilateral (ΔABC) $\vee$ equilateral (ΔABC)

ϕ [null].

Since, we arrived at the contradictory situation i.e. null clause. ~~There~~ Therefore, our assumption $\neg$ equal (AB, AC) is false. Therefore, the given statement equal (AB, AC) is true.

Date
Page

1. Mammal (Tom) ⟶ drinks (Tom, milk)
2. Man (Tom) ⟶ Mortal (Tom).
3. Man (Tom) ⟶ Mammal (Tom)
4. Man (Tom).

~~we prove~~
Converting these into clausal form.

1. ¬Mammal (Tom) V drinks (Tom, milk)
2. ¬Man (Tom) V Mortal (Tom)
3. ¬Man (Tom) V Mammal (Tom)
4. Man (Tom)
5. drinks (Tom, milk)

we prove ~~¬Man (Tom)~~ ¬drinks (Tom, milk) — 5

from 5 and 1
¬mammal (Tom) V drinks (Tom, milk) V ¬drinks (Tom, milk)
¬mammal (Tom) — 6
from 6 and 3
¬man (Tom) V Mammal (Tom) V ¬mammal (Tom)
¬man (Tom) — 7
from 7 and 4
¬man (Tom) V man (Tom)
ϕ [null].

Since, we arrived at the contradictory situation i.e. null clauses. Therefore our assumption ¬drinks (Tom, milk) is false. Therefore given statement drinks (Tom, milk) is true.

First order Logic / First order predicate Logic (FOL or FOPL)

Limitations of Propositional Logic :

* Consider following arguments.

A> All dogs are faithful.

B> Tommy is a dog.

Therefore, Tommy is faithful.

Suppose,

p : all dogs are faithful.

Q : Tommy is a dog.

$P \land Q \Rightarrow$ Tommy is faithful ?

We can't infer such in a propositional logic.

* Tom is hardworking.

hardworking (Tom).

Tom is an intelligent Student.

intelligent(Tom)

If Tom is hardworking and Tom is an intelligent student, then Tom scores high marks.

hardworking (Tom) $\land$ intelligent (Tom) $\xrightarrow{\text{scores}}$ $\land$ highmarks (Tom) ~~scored~~

Converting to the clausal form.

$\neg$ (hardworking (Tom) $\land$ intelligent(Tom)) $\lor$ scores highmarks (Tom).

$\neg$ (hardworking (Tom)) $\lor$ $\neg$(intelligent(Tom)) $\lor$ scoreshighmarks (Tom).

We cannot depict ~~these~~ ~~into~~ ~~so~~ such relationship for plethora of instances. But, we could write insteadd :-

Date
Page

All students who are hardworking and intelligent scores high marks.

For, All X such that, X is a student and X is intelligent and X is hardworking then X scores highmarks.

Therefore, FOL is generalization of propositional logic.

The problem of infinite model of propositional logic is addressed by predicate logic.

Syntax of FOPL

FOPL can be defined in terms of :-

1) Terms
2) Predicates
3) Quantifiers

1) Terms :-

It denotes some objects other than true or false (constants). e.g. Tommy is a dog. All man are Mortal. Terms can be constants or variables.

* Constants of type W is a name that denotes a particular objects in set W. e.g. 5, Tommy; etc.
* Variable of type W is a name that denotes any element in the set W. e.g. $X \in N$. N is a set of Natural numbers. $x \in D$, D belongs to set of dogs.

Function :- A functional term of arity n takes an object of type $W_1, W_2, \ldots, W_n$ as n input and returns an object W. e.g. $F(W_1, W_2, \ldots, W_n)$.

Sum (5, 6).

2) Predicates :-

Like function but return type is either true or false. e.g. greater than (X, Y), is true iff ~~(if and only if)~~

Date
Page

iff $x > y$.

Types of predicates :-

1) A predicate with no variable is a proposition. e.g Tommy is a dog.
2) A predicate with one variable is called property. e.g. X is a dog.

dog(x), iff x is dog.
→ T

Formulation of Predicates.

let, $P(x_1, y_1, \ldots)$ and $Q(x_2, y_2, \ldots)$ be two predicates, then $P \vee Q$, $P \wedge Q$, $\neg P$, $\neg Q$, $P \rightarrow Q$, $P \leftrightarrow Q$ are valid predicates.

Predicate examples:-

If Man is x, then x is Mortal.

Man(x) → Mortal(x)

¬Man(x) ∨ Mortal(x)

If n is a natural number, then n is either odd or even.

natural_number(n) ⟶ even(n) ∨ odd(n)

¬natural_number(n) ∨ even(n) ∨ odd(n)

Quantifier

It is to represent the number. There are two types of quantifier.

1) Universal quantifier (∀)
Read as for all.
2) Existential quantifier (∃)
Read as there exists.

Date

Page

example :-

x is a set of days of a week.

$\exists_x$: ~~day~~ x is days of a week.

Universal Quantifier :-

* All dogs are faithful.

$\forall_x : dog(x) \rightarrow faithful(x)$.

* All mammal drinks milk

$\forall_x : mammal(x) \rightarrow$ ~~dri drinkmilk (x), drink)~~ $drink(x, milk)$

* All man are mortal

$\forall_x : man(x) \longrightarrow mortal(x)$

* All birds cannot fly.

$\neg(\forall_x : bird(x) \rightarrow fly(x))$

* All man are mammal.

$\forall_x : man(x) \longrightarrow mammal(x)$.

Existential Quantifier :-

Represents some numbers in the set.

* At least one planet has a life in it.

$\exists_x : (planet(x) \wedge haslife(x))$

Date
Page

* There exists a bird, that can't fly.

$$\exists x : (bird(x) \wedge \neg fly(x))$$

Duality of a Quantifier

All man are mortal $\Leftrightarrow$ No man is immortal.

$\forall x : man(x) \rightarrow mortal(x) \longleftrightarrow \neg[\exists x (man(x) \wedge \neg mortal(x))]$

Sentences

A predicate is a sentence.

If P and $\neg P$ are sentences and x is a variable then

$P, \neg P, \exists x(P), P \wedge \neg P, P \vee \neg P, P \rightarrow \neg P$ are sentences.

Quiz

1> Some dog bark.
2> All dogs have four legs.
3> All barking dogs are irritating.
4> No dogs purr.
5> Fathers are male parents with children.
6> Students are people who are enrolled in a course.
7> Markus was a man.
8> Markus was a Pompeian.
9> All pompeian were Roman.
10> Caeser was a Ruler.
11> All Roman were either loyal to Caeser or hated him.
12> Everyone is loyal to someone.
13> People only try to assascinate Rulers they are not loyal too
14> Markus tried to assascinate Caeser.
15> All men are people.
16> There is a person who loves everyone in the world.

17> Everyone in this world is loved by at least one person.

18> John likes all kind of food.

19> Steps likes easy courses.

20> Everybody loves somebody.

21> Abhram is the father of Issac.

22> John gave the book to Cena.

23> Science courses are hard.

24> Anyone passing his history exam and wining the lottery is happy.

25> Anyone who studies or is lucky can pass all his exams.

26> John didnot study but he is lucky.

27> Anyone who is lucky wins the lottery.

28> Apples are food.

29> Chicken is food.

30> Anything anyone eats and is not killed by is food.

31> Bill eats peanuts and is still alive.

32> Sue eats everything bill eats.

1> $\exists x : (dog(x) \land bark(x))$

2> $\forall x : (dogs(x) \longrightarrow fourlegs(x))$

3> $\forall x : (\cancel{barking\ dogs(x)}\ dog(x) \land bark(x) \longrightarrow irritating(x))$

4> $\neg[\exists x : dog(x) \land \neg purr(x)]$ / $\forall x : dog(x) \rightarrow \neg purr(x)$

5> $\forall x : (father(x) \longrightarrow male(x) \land has\_children(x))$

6> $\exists x : (people(x) \land enrolled\ in\ course(x) \longrightarrow students(x))$

7> $\cancel{Markus(x) \Longrightarrow Man(x)}$ $Man(Markus)$

8> $Pompein(Markus)$

~~8> Ruler(Caeser) ∀x : Roman(x) → Pompeian~~

9> $\forall x : Pompeian(x) \longrightarrow Roman(x)$.

10> $ruler(Caeser)$.

11> $\forall x : Romans(x) \longrightarrow Loyal\_to(x, Caesor) \lor Hate(x, Caeser)$

12> $\forall x \exists y : loyal(x, y)$

17) Everyone in this world is loved by at least one person
18) John likes all kind of food.
19) Steve likes easy courses.
20) Everybody lows somebody.
21) Abhram is the father of Issac.
22) John gave the book to Cena.
23) Science courses are hard.
24) Anyone passing his history exam and wining the lottery is happy.
25) Anyone who studies or is lucky can pass all his exams.
26) John didnot study but he is lucky.
27) Anyone who is lucky wins the lottery.
28) Apples are food.
29) Chicken is food.
30) Anything anyone eats and is not killed by is food.
31) Bill eats peanuts and is still alive.
32) Sue eats everything bill eats.

1) $\exists x : (dog(x) \land bark(x))$
2) $\forall x : (dogs(x) \longrightarrow fourlegs(x))$
3) $\forall x : (\overset{dog(x) \land bark(x)}{\cancel{barking\_dogs(x)}} \longrightarrow irritating(x))$
4) $\neg[\exists x : dog(x) \land \neg purr(x)]$ / $\forall x : dog(x) \longrightarrow \neg purr(x)$
5) $\forall x : (father(x) \longrightarrow male(x) \land has\_children(x))$
6) $\exists x : (people(x) \land enrolled\_incrourse(x) \longrightarrow students(x))$
7) $\cancel{Markus(x) \longrightarrow Man(x)}$ $Man(Markus)$
8) $Pompein(Markus)$
8) $\cancel{Ruler(Caeser) \quad \forall x : Roman(x) \longrightarrow Pompeian}$
9) $\forall x : Pompeian(x) \longrightarrow Roman(x)$.
10) $ruler(Caeser)$.
11) $\forall x : Romans(x) \longrightarrow Loyal\_to(x, Caesar) \lor Hate(x, Caeser)$
12) $\forall x \exists y : loyal(x, y)$

Date
Page

13>

14> Tryassinate (Mercus, Caesar).

15> $\forall x$: man(x) $\longrightarrow$ people(x)

~~16> $\exists x \forall y$: person(x) $\longrightarrow$ loves everyon~~ loves(x, y).

16> ~~$\forall x \exists y$~~ $\exists x \forall y$: loves(x, y)

17> $\forall x \exists y$: Loves(x, y)

18> $\forall x$: food(x) $\longrightarrow$ likes(John, x).

19> ~~$\exists x$: easy_course(x) $\longrightarrow$ likes(Stepes, x)~~ ~~course(x) $\longrightarrow$ likes(steve, x) $\wedge$ easy(x)~~ ~~course(science) $\rightarrow$ hard(science)~~ course(x) $\wedge$ easy(x) $\longrightarrow$ likes(steve, x)

20> $\forall x \exists y$: loves(x, y).

21> Father(Abhram, Issac).

22> gave_bookto(John, Cena)

23> course(science) $\longrightarrow$ hard(science)

24> $\forall x$: (pass(x, history) $\wedge$ win(x, lottery)) $\rightarrow$ happy(x)

25> ~~$\neg$study(John) $\wedge$ lucky(John)~~. $\forall x \forall y$ study(x) $\vee$ lucky(x) $\rightarrow$ pass(x, y)

26> $\neg$study(John) $\wedge$ lucky(John)

27> $\forall x$ lucky(x) $\rightarrow$ win(x, lottery)

28> food(Apple)

29> food(chicken)

30>

Date

Page

Resolution in first order predicate logic

All people who are graduating are happy.
All happy people smile.
Someone is graduating.

prove:
Is someone smiling ?

Rules

Encoding / Converting into predicate logic.

$\forall x : graduating(x) \longrightarrow happy(x)$

$\forall x : happy(x) \longrightarrow smile(x)$

$\exists x : graduating(x)$

To prove:-

$\neg \exists x : Smile(x)$

Eliminate implication operator.

$\forall x : \neg graduating(x) \vee happy(x)$

$\forall x : \neg happy(x) \vee smile(x)$

$\exists x : graduating(x)$

$\neg \exists x : smile(x)$

Convert it into Canonical or normal form (Reduce scope of negation).

$\forall x : \neg graduating(x) \vee happy(x)$

$\forall x : \neg happy(x) \vee smile(x)$

$\exists x : graduating(x)$

$\cancel{\exists x : \neg smile(x)}$ $\forall x : \neg smile(x)$

Standorize variable apart:

$\forall x : \neg graduating(x) \vee happy(x)$
$\forall y : \neg happy(y) \vee smile(y)$
$\exists z : graduating(z)$
$\forall w : \neg smile(w)$

Skolemization (Remove existential quantifier)

The process of finding constant or function for existentially quantified variables that will make the clause to be true.

$\forall x : \neg graduating(x) \vee happy(x)$
$\forall y : \neg happy(y) \vee Smile(y)$
$graduating(name1)$
$\forall w : \neg smile(w)$

5) drop universal quantifier.

$\neg graduating(x) \vee happy(x)$
$\neg happy(y) \vee smile(y)$
$graduating(name1)$
$\neg Smile(w)$

From 4 and 2

$\neg \cancel{smile}(w)$ $\quad$ $\neg happy(y) \vee \cancel{smile}(y)$
(y/w)

$\neg happy(w)$

from 1

$\neg \cancel{happy}(w)$ $\quad$ $\neg graduating(x) \vee \cancel{happy}(x)$
(x/w)

$\neg graduating(w)$

Date |
Page |

¬graduating(w) graduating(name1)

(w | name1)

ϕ(null).

Hence, contradiction of goal is disproved. So, goal is proved.

Q3)

I>

1. $\forall x$: pass(x, history) ∧ win(x, lottery) → happy(x).
2. $\forall x \forall y$: study(x) ∨ lucky(x) → pass(x, y).
3. ¬study(John) ∧ lucky(John).
4. $\forall x$ lucky(x) → win(x, lottery).

To prove:-

⑤ ¬happy ~~grain~~ (John)

II> Eliminate implication operator.

① $\forall x$: ¬(pass(x, history) ∧ win(x, lottery)) ∨ happy(x)
② $\forall x, \forall y$: ¬(study(x) ∨ lucky(x) ∨ pass(x, y))
③ ¬study(John)
④ lucky(John)
⑤ $\forall x$: ¬lucky(x) ∨ win(x, lottery).
⑥ ¬happy(John).

III> Reduce scope of negation

① $\forall x$: ¬pass(x, history) ∨ ¬win(x, lottery) ∨ happy(x)
② $\forall x \forall y$ [¬study(x) ∧ ¬lucky(x)] ∨ pass(x, y)
③ ¬study(John)
④ lucky(John)
⑤ $\forall x$: ¬lucky(x) ∨ win(x, lottery)
⑥ ¬happy(John).

Date
Page

$\forall x : \neg pass(x, history) \lor \neg win(x, lottery) \lor happy(x)$

$\forall x \forall y : (\neg study(x) \lor pass(x,y)) \land (\cancel{\neg study(x)}\ \neg lucky(x) \lor pass(x,y))$

$\neg study(john)$

$\neg lucky(john)$

$\forall x : \neg lucky(x) \lor win(x, lottery)$

$\neg happy(john)$

Skolemization:-

$\forall x : \neg pass(x, history) \lor \neg win(x, lottery) \lor happy(x)$

$\forall x \forall y : \neg study(x) \lor pass(x,y)$

$\forall u \forall v : \cancel{\neg study}\ \neg lucky(u) \lor pass(u,v).$

$\neg study(john)$

$lucky(john)$

$\forall w : \neg lucky(w) \lor win(w, lottery)$

$\neg happy(john)$

drop universal quantifier

$\neg pass(x, history) \lor \neg win(x, lottery) \lor happy(x)$

$\neg study(x) \lor pass(x,y)$

$\neg lucky(u) \lor pass(u,v)$

$\neg study(john)$

$lucky(john)$

$\neg lucky(w) \lor win(w, lottery)$

$\neg happy(john)$

from (4) and (1)

from (8) and (6)

$\neg pass(john, history) \vee \neg win(john, \cancel{lottery})$: $\neg lucky(w) \vee win(\cancel{w, lottery})$

$(w/john)$

$\neg pass(john, history) \vee \neg lucky(john)$ — (9)

from (9) and (5)

$\neg pass(john, history) \vee \neg(\cancel{lucky(john)})$ $\quad$ $\cancel{lucky(john)}$

$\neg pass(john, history)$ — (10)

from (10) and (3)

$\neg pass(john, history)$ $\quad$ $\neg lucky(u) \vee pass(u, v)$

$(u/john, v/history)$

$\cancel{\neg pass(john, history)}$ $\quad$ $\neg lucky(john) \vee \cancel{pass(john, history)}$

$\neg lucky(john)$ — (11)

from (11) and (5)

$\cancel{\neg lucky(john)}$. $\quad$ $\cancel{lucky(john)}$

ϕ [null]

Hence contradiction of the goal is disproved. So, goal i.e. john is happy is proved.

Date
Page

Q3)

1. Steve only like easy courses.
2. Science courses are hard.
3. all courses in basket-weaving are easy.
4. BK301 is a basket weaving course.
5. Which course would steve likes ?

(a) $\forall x$: easy(x) $\longrightarrow$ likes(steve, x)
(b) $\forall x$: science(x) $\longrightarrow$ $\neg$ easy(x)
(c) $\forall x$: basket-weaving(x) $\longrightarrow$ easy(x).
(d) basketweaving(BK301).
(e) likes(steve, x).

I> Converting to clausal form.

Date
Page

(d) Basketweaving(BK301)

(e) ¬likes(steve, x).

a, e → ¬easy(x) —— (f)

f, c → ¬ ~~Bas~~ Basketweaving(x) —— (g) (x/BK301)

g, d → Φ

Q>

A> Ram like all kind of food.

B> Oranges are food.

C> Rice is a food.

D> Krishna eats popcorn and is still alive.

E> Radha eats anything that Krishna eats.

F> Anything anyone eats and is not killed by is food.

G> Prove :: Ram likes popcorn.

I> Converting to clausal form.

A> ∀x : food(x) → ~~likes~~ eats(Ram, x).

B> food(orange).

C> food(Rice).

D> eats(Krishna, popcorn) ∧ ~~alive~~ ¬Killed(Krishna, popcorn).

E> ~~∀x : eats(Radha, x)~~ → ∀x : eats(Krishna, x) → eats(Radha, x).

F> ∀x ∀y : (eats(x, y) ∧ ¬Killed(y)) → food(y)

F> ¬eats(Ram, popcorn).

II> Eliminate implication operator.

A> ∀x : ¬food(x) ∨ eats(Ram, x).

B> food(orange)

C> food(Rice)

D> eats(Krishna, popcorn) ∧ ¬Killed(Krishna, popcorn).

E> ∀x : ¬eats(Krishna, x) ∨ eats(Radha, x).

F> $\forall x \forall y: \neg[eats(x,y) \land \neg Killed(x)] \lor food(y)$

G> $\neg eats(Ram, popcorn)$

III> Reduce scope of negation.

A> $\forall x: \neg food(x) \lor eats(Ram, x)$

B> $food(orange)$

C> $food(Rice)$

D> $\neg eats(Krishna, popcorn) \land \neg Killed(Krishna)$ ~~popcorn~~

E> $\forall x: \neg eats(Krishna, x) \lor eats(Radha, x)$

F> $\forall x \forall y: \neg eats(x,y) \lor \neg Killed(x) \lor food(y)$

G> $\neg eats(Ram, popcorn)$.

IV> Standarlize variable apart.

A> $\forall x: \neg food(x) \lor eats(Ram, x)$

B> $food(orange)$

C> $food(Rice)$

D> $\neg eats(Krishna, popcorn) \land \neg Killed(Krishna)$ ~~popcorn~~

E> $\forall y: \neg eats(Krishna, y) \lor eats(Radha, y)$.

F> $\forall u, \forall v: \neg eats(u,v) \lor \neg Killed(u) \lor food(v)$

G> $\neg eats(Ram, popcorn)$.

V> drop Universal quantifier.

A> $\neg food(x) \lor eats(Ram, x)$

B> $food(orange)$

C> $food(Rice)$

E ¬

D> $\neg eats(Krishna, popcorn) \land \neg Killed(Krishna)$ ~~popcorn~~

E> $\neg eats(Krishna, y) \lor eats(Radha, y)$

F> $\neg eats(u,v) \lor \neg Killed(u) \lor food(v)$.

G> $\neg eats(Ram, popcorn)$.

from (A) and (H)

~~Rntrl~~ : $\neg food(x) \lor eats(Ram, x)$ $\qquad$ $\neg eats(Ram, popcorn)$

$(x / popcorn)$

$\neg food(popcorn)$ —— (H)

from (H) and (G)

$\neg food(popcorn)$ $\qquad$ $\neg eats(u, v) \lor \neg killed(u) \lor food(v)$

$(v / popcorn)$

$\neg eats(u, popcorn) \lor \neg killed(\cancel{popcorn})$ u. —— (I)

from (I) and (D)

~~$\neg eats(u, popcorn) \lor \neg killed(popcorn)$ $\quad$ $\neg eats(krishna, y) \lor eats(Radha, y)$~~

$\neg eats(u, popcorn) \lor \neg killed(\cancel{popcorn})$ u $\qquad$ $eats(krishna, popcorn)$

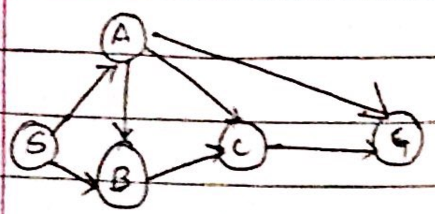
$(u / krishna)$

$\neg killed(\cancel{popcorn})$ krishna. —— (K)

from (K) and (F)

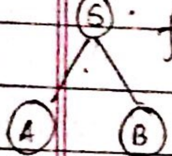
Frames

A* Search



	S	A	B	C	G
S		1	4		
A			2	5	12
B				2	
C					3
G					

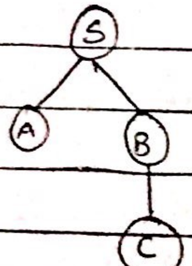
state	h(n)
S	7
A	6
B	2
C	1
G	0



$f(S) = g(S) + h(S) = 0 + 7 = 7$

$f(A) = g(A) + h(A) = 1 + 6 = 7$

$f(B) = g(B) + h(B) = 4 + 2 = 6$

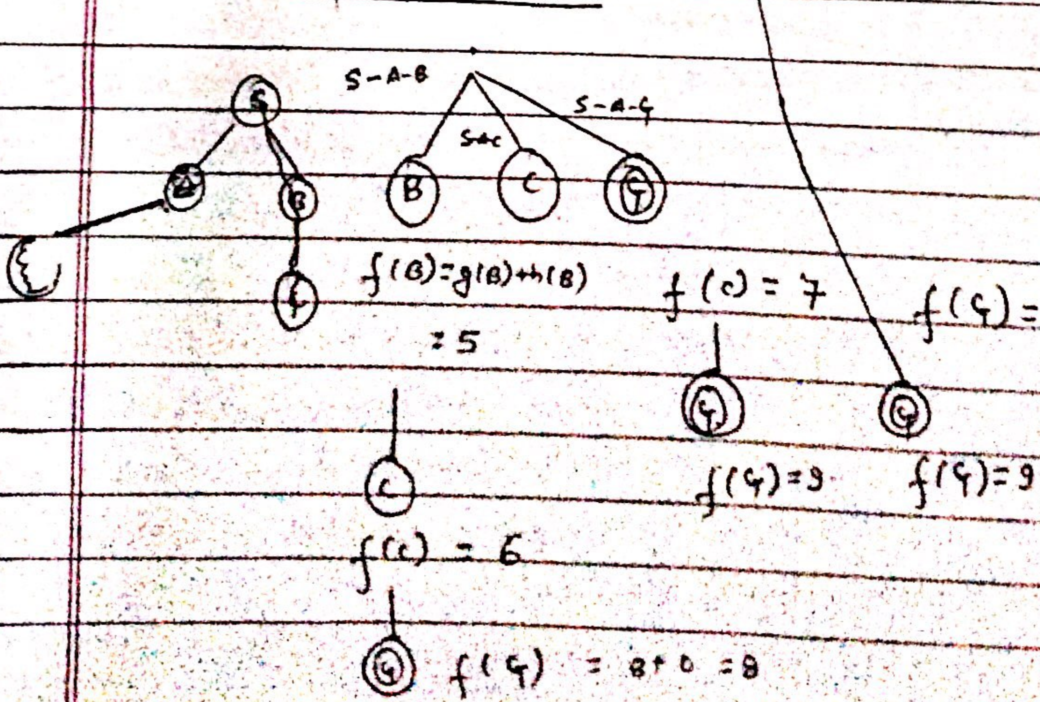


$f(C) = g(C) + h(C)$

$= [(S-B) + (B-C)] + 1$

$= 7$

Alphabetical order



S-A-B

S-A-C

S-A-G

$f(B) = g(B) + h(B)$

$= 5$

$f(C) = 7$

$f(G) = 13$

$f(G) = 9$

$f(G) = 9$

$f(C) = 6$

$f(G) = 8 + 0 = 8$

Hence, using A^* Search algorithm we can reach the goal in 4 different ways

i.e. S – A – B – C – G

S – A – C – G

S – A – G

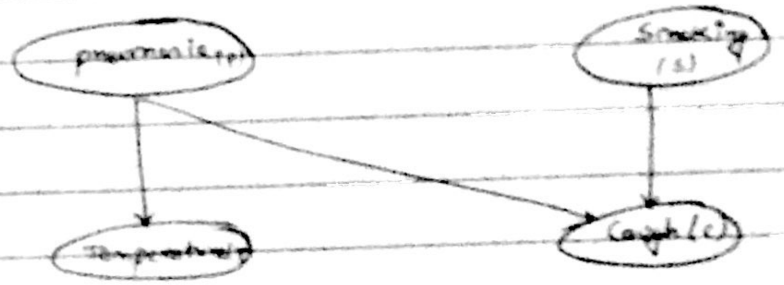
S – B – C – G

whose evaluation function ($f(n)$) equals to 8, 9, 13, 9 respectively.
Among these path S – A – B – C – G is the shortest and hence, this part is chosen.

pneumonia

True 0.1

False 0.9



Smoking

True 0.2

False 0.8

Pneumonia	Temperature	
	yes	No
yes	0.9	0.1
No	0.2	0.8

Pneumonia	Smoking	Cough True	Cough False
True	yes	0.95	0.05
True	No	0.8	0.2
False	Yes	0.6	0.4
False	No	0.05	0.95

(I) $P(c \mid S \wedge P) = 0.95$

(II) $P(s \mid c) = \dfrac{P(c \mid s) * P(s)}{P(c)}$

we have to calculate,

$$P(c \mid s) * P(s) = P(s) * [P(c \mid s \wedge P) * P(P) + P(c \mid s \wedge \neg P) * P(\neg P)]$$

$$= 0.2\,[0.95 * 0.1 + 0.6 * 0.9]$$

$$= 0.127$$

$$P(c) = P(c \mid P \wedge S) * P(P) * P(s) + P(c \mid P \wedge \neg S) * P(P) * P(\neg S) + P(c \mid \neg P \wedge S) * P(\neg P) * P(s) + P(c \mid \neg P \wedge \neg S) * P(\neg P) * P(\neg S)$$

$$= 0.95 * 0.1 \times 0.2 + 0.8 * 0.1 * 0.8 + 0.6 * 0.9 * 0.2 + 0.05 * 0.9 * 0.8$$

Unit 5 - Machine Learning

Learning

Types of Learning:-

1> Rote Learning
2> Learning by Examples.
3> Explanation based Learning.
4> Learning by taking advice.
5> Learning by analogy.
6> Inductive Learning
7> Deductive Learning

459-461 of AI, Rich and Knight.

Date
Page

Genetic Algorithms (by Elaine Rich, Kevin Knight, Shivashankar B. Nair).

- Search procedure based on a simple model of evolution
- Uses a "random" process to explore search space.
- Has been applied in many domains.
- Begin with a population of individuals. Each individual represents a solution to the problem we are trying to solve.
- A data structure describes the genetic structure of the individual. (Assume for initial discussion that this is a string of 0's and 1's).
- In genetics, the strings are called chromasomes and the bits are called genes.
- The string associated with each individual is it's genotype.
- Selection is based on fitness of individuals.
- Each evolving population of individuals is called a generation.
- Given a population of individuals corresponding to one generation the algorithm simulates natural selection and reproduction in order to obtain the next generation.

Three basic Operations:-

1) Reproduction

→ Individuals from one generation are selected for the next generation.

2) Crossover

→ Genetic material from one individual is exchanged with genetic material from another individual.

3) Mutation

Selection, crossover and mutation operations:

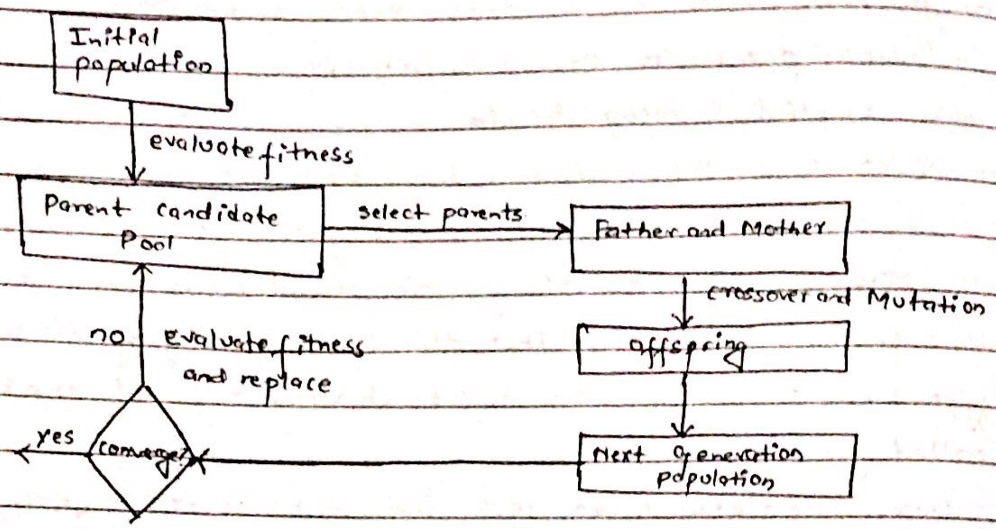
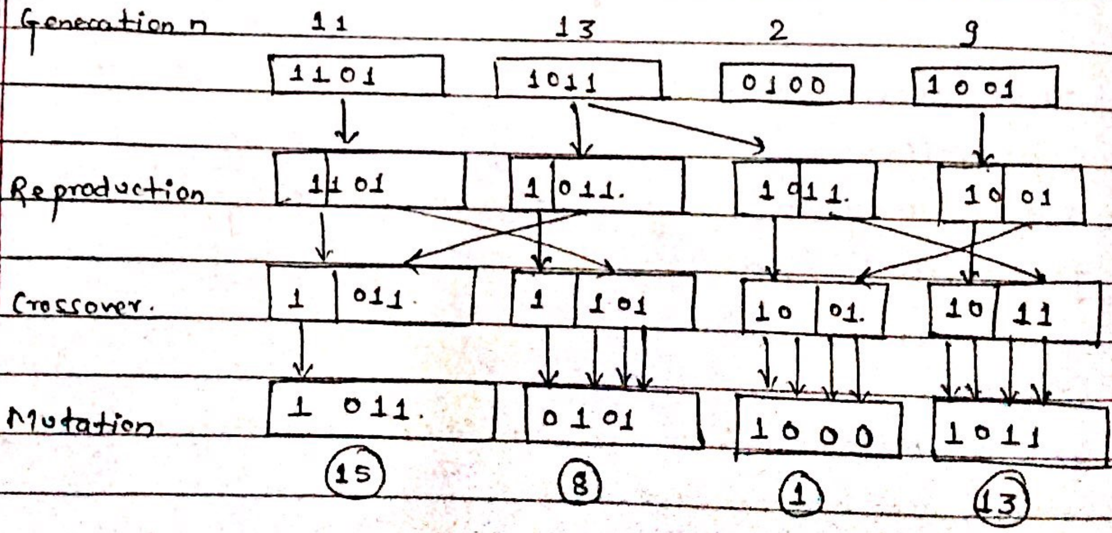


fig: Selection, crossover and mutation process

Example of General GA Procedure



Two keys to the success of a GA

- Data structure for
→ Genes
→ Chromosomes
→ Population
- Fitness evaluation function.

Date
Page

Example 7 :- While watching a game of champions League football in a cafe, you observe someone who is clearly supporting Manchester United in the game.
What is probability that they were actually born within 25 miles of Manchester?

Assume that
the probability that a randomly selected person in a typical local bar environment is born within 25 miles of Manchester is $\frac{1}{20}$ and

the chance that a person born within 25 miles of Manchester actually supports United is $\frac{7}{10}$.

the probability that a person not born within 25 miles of Manchester supports United with probability $\frac{1}{10}$.

Solution,

$P(B) = \frac{1}{20}$

$P(U/B) = \frac{7}{10}$

Now,

$$P(B/U) = \frac{P(U/B) * P(B)}{P(U)}$$

$$= \frac{\frac{7}{10} \times \frac{1}{20}}{P(U/B) * P(B) + P(U/\neg B) * P(\neg B)}$$

$$= \frac{\frac{7}{200}}{\frac{7}{10} \times \frac{1}{20} + \frac{1}{10} \times \frac{19}{20}}$$

$$= \frac{\frac{7}{200}}{\frac{26}{200}} = 0.26$$

Q. $15x - x^2$

[0 - 31]

n = 6

Solution:-

Steps:

→ Generate initial population at random. They are chromosomes or genotypes. e.g.

01101 → 13   01001 → 9.   01111 → 15

01011 → 11   10011 → 19.   10010 → 18

→ Calculate the fitness value for each individual.

ⓐ Decode the individual into an integer (called phenotypes).

01101 → 13   01011 → 11   01001 → 9   10011 → 19

01111 → 15   10010 → 18

ⓑ Evaluate the fitness according to $f(x) = 15x - x^2$.

13 → 26   11 → 44   9 → 54   19 → -76

15 → 210   18 → ~~324~~ -54

→ Select parents (two individuals) for crossover based on their fitness in P_i. Out of many methods for selecting the best chromosomes, if roulette wheel selection is used, then the probability of the i^{th} string in the population is $p_i = F_i / \left(\sum_{J=1}^{n} F_J\right)$.

String NO	Initial pop^n	X value	Fitness F_i $f(x) = 15x - x^2$	P_i	Expected count N* Prob i
1	01101	13	26	~~0.04~~ -0.10	-~~0.04~~ 0.60
2	01011	11	44	~~0.07~~ -0.17	- 1.03
3	01001	9	54	~~0.09~~ -0.21	- 1.26
4	10011	19	-76	~~-0.12~~ 0.29	1.78
5	01111	15	-210	0.38	5.85
6	10010	18	-584	~~0.54~~ 0.21	1.26
sum			-256	1.00	6.00

Average	-43	0.17	1
Max	54	-0.21	-1.26

The string no 3 has maximum chance of selection.

→ Produce a new generation of solutions by picking from the existing pool of solutions with a preference for solutions which are better suited than others.

We divide the range into six bins, sized according to the relative fitness of the solutions which they represent.

Strings	Prob i	Associated Bin
01101	-0.10	0.0 ----- -0.10
01011	-0.17	-0.10 ---- -0.27
01001	-0.21	-0.27 ---- -0.48
10011	0.29	-0.48 ---- -0.19
01111	0.98	-0.19 ---- 0.79
10010	0.21	0.79 ----- 1.00

By generating 6 uniform (0,1) random values and seeing which bin they fall into we pick the ~~four~~ six string that will form the basis for the next generation.

Random No	Falls into bin	Chosen string
-0.08	0.0 ··· -0.10	01101
-0.24	-0.10 ··· -0.27	01011
-0.35	-0.27 ···· -0.48	01001
-0.20	-0.10 ··· -0.27	~~01011~~ 01111
0.40	-0.27 ··· -0.48	01001
0.87	0.79 ··· 1.00	10010

→ Randomly pair the members of the new generation.

For the first pair of strings: 01101, 01011

→ We randomly select the crossover point to be after ~~fourth~~ third digit

01101 ⇒ 011|01 ⇒ 01111

01011 ⇒ 010|11 ⇒ 01001

For the second pair of strings: 01001, 01011.

→ We randomly select the crossover point to be after second digit

01|001 ⇒ 01 001 ⇒ 01 011.

01|011 ⇒ 01|011 ⇒ 01 001.

For the last pair of strings: 01001, 10010.

→ We randomly select the crossover point to be after fourth digit.

0100|1. ⇒ 01000

1001|0 ⇒ 10011.

→ Randomly mutate a very small fraction of genes in population.

→ Go back and re-evaluates fitness of the population.

String No	Initial popⁿ (chromosomes)	X value (phenotypes)	Fitness $f(x)=15x-x^2$	prob i (fraction of total)	Expected amount
1	01111				
2	01001				
3	01011				
4	01001				
5	01000				

Date
Page

$15x - x^2$.

$[0 - 15]$

① 0000 ...... 1111.

0001
0010
0011.

② 0010, 0110, 1100, 1010

③ $f(x) = 15x - x^2$

$f(2) = 26$

$f(6) = 54$

$f(12) = 36$

$f(10) = 50$

④

S.N	Chromosome	decimal	$f(x)$	P_i
1	0010	2	26	0.156
2	0110	6	54	0.325
3	1100	12	36	0.216
4	1010	10	50	0.301.
sum			166	1.00
avg			41.5.	

⑤

0 — 0010 | 0110 | 1100 | 1010 — 1

0.156 0.481 0.697

0 ··· 0.156 → 0010
0.156 ··· 0.481 → 0110
0.481 ··· 0.697 → 1100
0.697 ··· 1.000 → 1010

1 0 0 1 0 3 1 0 1 0

2 0 1 1 0 4 1 1 0 0

0 1 1 0 3' 1 0 0 0

0 0 1 0 4' 1 1 1 0

S.N.	Chromosomes	decimal	f(x)	p(i)
1	0110	6	54	0.56
2	0010	2	26	0.123

Date

Page

Assignments

1. Explanation Based Learning
2. Boltzmann machine

Neural Networks

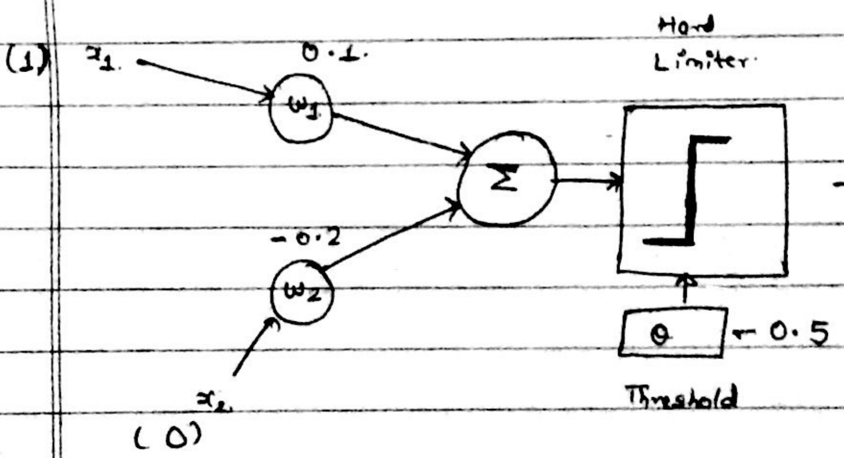
Perceptron

Step 1: Initialization.

step 2: Activation

step 3: Weight Training

step 4: Iteration



Realize AND operation using Perceptron training algorithm, Perform two iterations

1 AND 0 = 0 $\therefore Y_d(1) = 0$

$$Y_d'(p) = \text{Sigmoid}\left[\sum_{i=1}^{n} x_i(p)\, w_i(p) - \theta\right].$$

$$Y_d(1) = \text{Sigmoid}\left[1 \times 0.1 + 0 \times (-0.2) + 0.5\right]$$

$$Y_d(1) = \text{Sigmoid}\left[0.6\right].$$

$\alpha = 0.1, 0.01.$

Date
Page

$= \dfrac{1}{1+e^{-0.6}}$

$= \cancel{0.4\ldots}\ 0.6456$

$e(1) = Y_d(1) - Y(1)$

$= 0 - 0.6456$

$= -0.6456$

$w_i(p+1) = w_i(p) + \Delta w_i(p)$

$= w_i(p) + \alpha \cdot x_i(p) \cdot e(p)$

$w_1(2) = 0.1 + 0.01 * 1 * (-0.6456)$

$= 0.09354$

$w_2(2) = -0.2 + 0.01 * 0 * (-0.6456).$

$= -0.2.$

Iteration 2nd

$Y_d(2) = \text{Sigmoid}\left[\sum_{p=1}^{n} x_i(p) \cdot w_i(p) - \theta\right]$

$= \text{Sigmoid}\left[1 \times 0.09354 + 0 \times (-0.2) + 0.5\right]$

$= \text{Sigmoid}\left[0.59354\right]$

$= \dfrac{1}{1+e^{-0.59354}}$

$= 0.64417$

$e(2) = Y_d(2) - Y(2)$

$= 0 - 0.64417$

$= -0.64417$

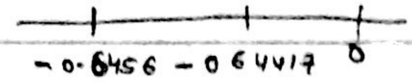
$w_i(p+1) = w_i(p+1) + \Delta w_i(p+1)$.

$w_1(3) = 0.09354 + 0.01 \times 1 \times (-0.64417)$

$= 0.08709$

$w_2(3) = -0.2 + 0.01 * 0 * (-0.64417)$

$= -0.2$



Adaline (Adaptive Linear Element)

Madaline (Multiple Adaline). → (Always Sigmoid function)

Multilayer Perceptron

Q. A factory production line is manufacturing bolts using three machines, A, B, C. ~~off~~ Of the total output, machine A is responsible for 25%, B for 35% and C for the rest. It is known from the previous experience with the machines that 5% of the output from machine A is defected; 4% from machine B, and 2% from machine C. A bolt is choosen at random from the production line and found to be defective. What is the probability that it came from

(a) Machine A

(b) Machine B

(c) Machine C.

Soln:-

$P(A) = \frac{25}{100} = 0.25$ $P(B) = \frac{35}{100} = 0.35$ $P(C) = 0.40$

$P(D/A) = \frac{5}{100} = 0.05$

$P(D/B) = \frac{4}{100} = 0.04$

$P(D/C) = 0.02$

$P(D) = P(A) * P(D/A) + P(B) * P(D/B) + P(C) * P(D/C)$

$= 0.0345.$

Now,

(I) $P(A/D) = \frac{P(D/A) * P(A)}{P(D)}$

$= \frac{0.05 \times 0.25}{0.0345}$

$= 0.3623$

(II) $P(B/D) = \frac{P(D/B) \times P(B)}{P(D)}$

$= 0.406$

(III) $P(C/D) = \frac{P(D/B) \times P(C)}{P(D)}$

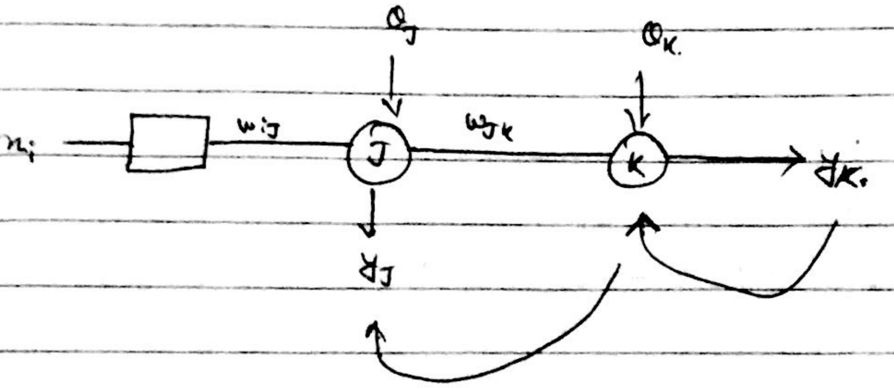
$= \frac{0.02 \times 0.40}{0.0345}$

$= 0.232.$

Date
Page

Back Propogation

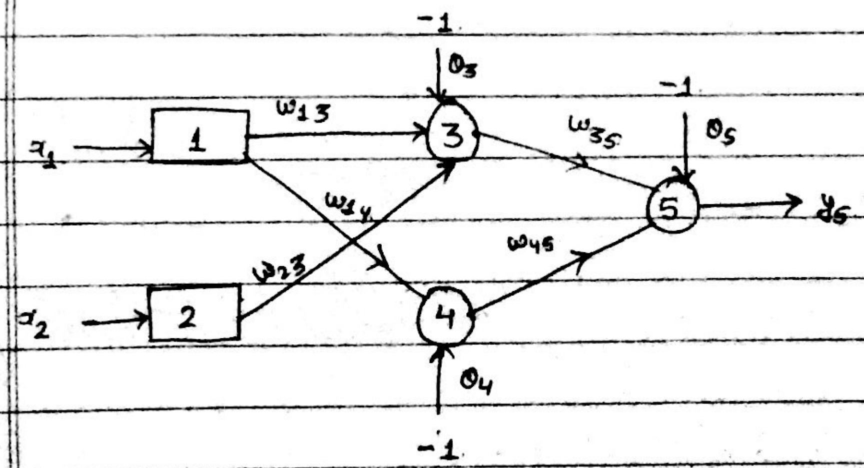
$$y_J(p) = \text{sigmoid}\left[\sum_{i=1}^{2} x_i(p)\cdot w_{iJ}(p) - \theta_J\right].$$



Imp

Kohonen Network

→ Self organized maps.
→ Competitive Learning (neurons compete among themselves to be activated).
→ Learning of Neural network without the presence of teacher.



$w_{13} = 0.5$, $w_{14} = 0.9$, $w_{23} = 0.4$, $w_{24} = 1.0$, $w_{35} = -1.2$,
$w_{45} = 1.1$, $\theta_3 = 0.8$, $\theta_4 = -0.1$ and $\theta_5 = 0.3$

We consider a training set where input x_1 and x_2 are equal to 1 and desired output $y_{d,5}$ is 0. The actual output of neurons 3 and 4 in the hidden layers are calculated as:

Date
Page

$y_3 = \text{sigmoid}(x_1 w_{13} + x_2 \cdot w_{23} - \theta_3)$
$= \text{sigmoid}(1 \times 0.5 + 1 \times 0.4 - 0.8)$
$= \text{sigmoid}(0.1)$
$= \dfrac{1}{1+e^{-(0.1)}}$
$= 0.5250$

$y_4 = \text{sigmoid}(x_2 w_{14} + x_2 w_{24} - \theta_4)$
$= \text{sigmoid}(1 \times 0.9 + 1 \times 1.0 + 0.1)$
$= \text{sigmoid}(2)$
$= \dfrac{1}{1+e^{-2}}$
$= 0.8808$

Now,
Actual output of neuron 5 in the output layer is determined as:
$y_5 = \text{sigmoid}(y_3 w_{35} + y_4 w_{45} - \theta_5)$
$= \text{sigmoid}(0.5250 \times (-1.2) + 0.8808 \times 1.1 - 0.3)$
$= \text{sigmoid}(0.0388)$
$= \dfrac{1}{1+e^{-0.0388}}$
$= 0.5097$

Thus, the following error is obtained as:
$e = y_{d,5} - y_5 = 0 - 0.5097 = -0.5097$

The next step is weight training. To update the weights and threshold levels in our network, we propagate the error, e from the output layer backward to the input layer.

First, we calculate the error gradient for neuron 5 in the output layer,

Date

Page

$\delta_5 = y_5(1-y_5)\, e$

$= 0.5097\,(1-0.5097) \times (-0.5097)$

$= -0.1274$

Then we determine the weight corrections assuming that the learning rate parameter, α is equal to 0.1.

$\Delta w_{35} = \alpha \cdot y_3 \cdot \delta_5 = 0.1 \times 0.5250 \times (-0.1274)$

$= 0.0067$

$\Delta w_{45} = \alpha \cdot y_4 \cdot \delta_5$

$= 0.1 \times 0.8808 \times (-0.1274)$

$= -0.0112$

$\Delta \theta_5 = \alpha \cdot (-1) \cdot \delta_5$

$= 0.1 \times (-1) \times (-0.1274)$

$= 0.0127$

Now, we calculate the error gradient for neuron 3 and 4 in hidden layer.

$\delta_3 = y_3(1-y_3) \cdot \delta_5 \cdot w_{35}$

$= 0.5250 \times (1-0.5250) \times (-0.1274) \times (-1.2)$

$= 0.0381.$

$\delta_4 = y_4(1-y_4) \cdot \delta_5 \cdot w_{45}$

$= 0.8808\,(1-0.8808) \times (-0.1274) \times 1.1$

$= -0.0147$

We, then determine the weight corrections.

$\Delta w_{13} = \alpha \cdot x_1 \cdot \delta_3 = 0.1 \times 1 \times 0.0381 = 0.0038$

$\Delta w_{23} = \alpha \cdot x_2 \cdot \delta_3 = 0.1 \times 1 \times 0.0381 = 0.0038$

$\Delta \theta_3 = \alpha \cdot (-1) \cdot \delta_3 = 0.1 \times (-1) \times 0.0381 = -0.0038$

$\Delta w_{14} = \alpha \cdot x_1 \cdot \delta_4 = 0.1 \times (1) \times (-0.0147) = -0.0015$

$\Delta w_{24} = \alpha \cdot x_2 \cdot \delta_4$

$= 0.1 \times 1 \times (-0.0147)$

$= -0.0015$

$\Delta \theta_4 = \alpha \cdot (-1) \times \delta_4$

$= 0.1 \times (-1) \times (-0.0147)$

$= 0.0015$

At last, we update all weights and thresholds

$w_{13} = w_{13} + \Delta w_{13} = 0.5 + 0.0038 = 0.5038$

$w_{14} = w_{14} + \Delta w_{14} = 0.9 - 0.0015 = 0.8985$

$w_{23} = w_{23} + \Delta w_{23} = 0.4 + 0.0038 = 0.4038$

$w_{24} = w_{24} + \Delta w_{24} = 1.0 - 0.0015 = 0.9985$

$w_{35} = w_{35} + \Delta w_{35} = -1.2 - 0.0067 = -1.2067$

$w_{45} = w_{45} + \Delta w_{45} = 1.1 - 0.0112 = 1.0888$

$\theta_3 = \theta_3 + \Delta \theta_3 = 0.8 - 0.0038 = 0.7962$

$\theta_4 = \theta_4 + \Delta \theta_4 = -0.1 + 0.0015 = -0.0985$

$\theta_5 = \theta_5 + \Delta \theta_5 = 0.3 + 0.0127 = 0.3127$

Here the calling population is finite and consists of 40 machine.

ii) Infinite calling population

In systems with a large population of potential customer the calling population is assumed to be infinite. eg. the potential customer's arrival at opd billing ~~cust~~ counter.

patient → Customer

clerk → server

The main difference between finite and infinite calling population model is how the arrival rate is defined.

- In infinite population model, the arrival rate is not affected by the no of customers who have left the calling population and joined queuing system when arrival process is homogeneous over time.

→ The average time rate is usually assumed to be constant

→ for finite population model, the arrival rate to the queuing system ~~does not~~ depends on the number of customer being served and waiting.

and they are related as

$$\boxed{\lambda = \frac{1}{T_a}}$$

e.g. for an office working 8 hrs a day 5 day in a week. Suppose ia gets about 800 telephone calls in a week. Suppose is to be model of the office during working hours is to be constructed using time scale of minute; find avg. inter arrival time and arrival rate.

Soln:

total working hr = 8×5 = 40 hr.

Averge arrival time = $\frac{40 \times 60}{800}$ = 3 minute.

i.e. T_a = 3 min

Arrival rate(λ) = $\frac{1}{T_a}$ = $\frac{1}{3}$ = 0.333 call / min

The calling population

The customers or inputs requesting for service from a system is called calling population. The population of potential customer i.e. the popn or requesting entities for getting a service is known as calling population. It may be finite or infinite.

- finite calling population

The system in which requesting entities are finite are called finite calling population. e.g. ATM, ~~longue~~ lounge service for bank with 10 ATM machine cuing for money. After an interval of time money should put by the bank ~~long~~ ~~longue~~ lounge worker ~~should put~~ who fills the money into the money box. Here,

ATM machine → Customer

Bank lounge work → Server

i) Timely ii) Random iii) Batched

i) Timely arrival pattern

It is reflected as scheduled arrivals such as patient to a doctor's clinic, students at classes, etc.

On this case, inter-arrival time may be constant or (constant $\pm \alpha$)

where α is the small random amount of time representing early or late arrival.

ii) Random arrival pattern

In this class, the arrival may occur at any random time. The inter arrival time are usually characterised by a probability distribution in case of random time.

eg. arrival of customer at ATM.

iii) Batched arrival pattern :-

In this pattern, customer may arrive one at a time or in batches, the batch may be of constant size or random sized.

→ An additional situation of arrival pattern occurs when at least one customer is assumed always be present in the queue so that server is never idle due to lack of customer.

For example, the ~~custo~~ customer may represent raw material for a product and sufficient material is always available.

The most important model for a random arrivals are poisson arrival pattern.

The following notation are used for describing the arrival patterns

T_a → mean arrival time

λ — mean arrival rate

Generation of Arrival Pattern

Generation of exogeneous arrivals are necessary for testing the reliability of a system being simulated in a discrete system simulation. e.g.

testing the design of logic circuits such as components of digital computers. A particular sequence of signals might be designed as the simulation i/p to see if the design reach as expected or not.

- the sequence of inputs may also be generated from the observation of a system. This approach of generation of arrival pattern is known as trace driven simulation in which system being simulated is tested with the records collected from the running system. by program monitors while little or no disturbance in system operation.
- When there is no interaction between the exogeneous arrivals and endogeneous events, it is permissible to create a sequence of arrivals in preparation for the simulation.
- The process of generating arrival of successor entity by the predecessor is known as boot strapping.

The arrival process

The arrival process of an infinite population model is usually characterized in terms of inter-arrival time of successive i/p, customers /entity /or request.

The various arrival pattern process may be of following types

Discrete system:

The system in which changes are predominantly discontinuous are called discrete system.

e.g. telephone system, shopping system, etc.

Numerical computation technique for discrete models

In discrete model various parameters at time interval (i) is recorded and performance analysis for the given system is achieved for discrete time interval.

For example, in accounting system ~~an~~ accountant begins his work everyday work with a file of documents to be processed. The time taken to process them may varies. They work through the file, begining each document as soon as ~~possible~~. finished the previous job except that he can take a fixed time (5 min) break after finishing a job. It is an hour or more since he began a work or since he last has a break.

We assume that time to process the document are given. We may also keep count of how many documents are left. The count will be ~~set~~ initially set to the no of documents at the begining time of the day and assume that no additional document arrive on that day. The count will be documented for each complete job and wish that if count ~~so~~ goes to zero. The computation can be organized as shown in table below;

Doc no, i	st. time	work time	finish time	Cumulative time	Break time	No of job N
1	0	45	45	45	0	20
2	45	16	61	61	1	19
3	66	5	71	5	0	18
4	71	20	91	25	0	17
5	91	8	99	33	1	16
6	104	10	114	10	0	5