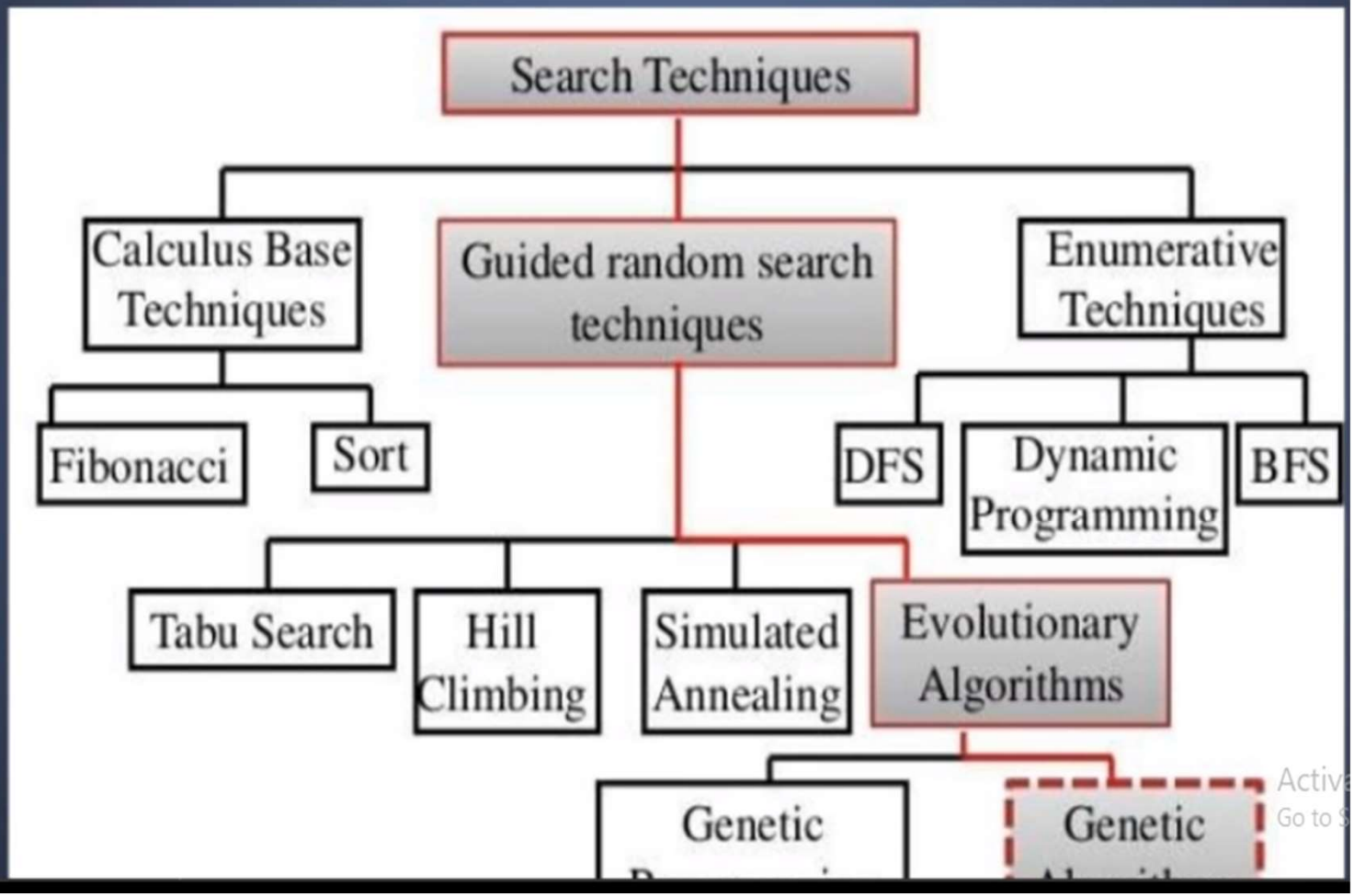
Application : Job Scheduling

Search Techniques



Why Genetic Algorithm?

- Widely-used in business, science and engineering
  - Optimization and Search Problems
  - Scheduling and Timetabling

Basic Algorithm of Genetic Algorithm

randomly initialize population(t)
determine fitness of population(t)
repeat
    select parents from population(t)
    perform crossover on parents creating population(t+1)
    perform mutation of population(t+1)
    determine fitness of population(t+1)
until best individual is good enough

Clip slide

Outline of the Basic Genetic Algorithm

1. [Start] Generate random population of n chromosomes (suitable solutions for the problem)
2. [Fitness] Evaluate the fitness f(x) of each chromosome x in the population
3. [New population] Create a new population by repeating following steps until the new population is complete
    1. *[Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)*
    2. *[Crossover] With a crossover probability crossover the parents to form a new children. If no crossover was performed, children is an exact copy of parents.*

Outline of the Basic Genetic Algorithm

4. [Replace] Use new generated population for a further run of algorithm
5. [Test] If the end condition is satisfied, stop, and return the best solution in current population
6. [Loop] Go to step 2

List of Genetic Algorithm applications

1. Airlines Revenue Management.
2. Wireless sensor/ad-hoc networks.
3. Audio watermark insertion/detection.
4. Protein folding and protein/ligand docking.
5. Pop music record producer.
6. Financial Mathematics.
7. Finding hardware bugs.

Initial Population

We start with a population of n random strings. Suppose that length is = 10 and total n = 6

s1 = 1111010101

s2 = 0111000101

s3 = 1110110101

s4 = 0100010011

s5 = 1110111101

s_6 = 0100110000

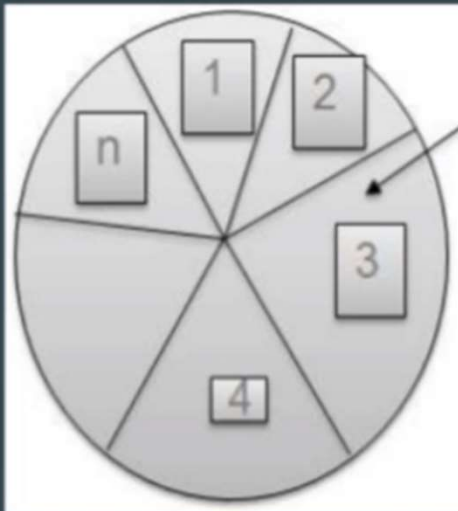
Fitness Function: f()

$s_1 = 1111010101$	$f(s_1) = 7$
$s_2 = 0111000101$	$f(s_2) = 5$
$s_3 = 1110110101$	$f(s_3) = 7$
$s_4 = 0100010011$	$f(s_4) = 4$
$s_5 = 1110111101$	$f(s_5) = 8$
$s_6 = 0100110000$	$f(s_6) = 3$

Total = 34

Selection (1)

Next we apply fitness proportionate selection with the roulette wheel method:



Area is Proportional to fitness value

Individual *i* will have a probability to be chosen

$$\frac{f(i)}{\sum_i f(i)}$$

We repeat the extraction as many times as the number of individuals

we need to have the same parent population size (6 in our case)

Selection (2)

Suppose that, after performing selection, we get the following population:

s1' = 1111010101 (s1)

s2' = 1110110101 (s3)

s3' = 1110111101 (s5)

s4' = 0111000101 (s2)

Basic Algorithm of Genetic Algorithm

randomly initialize population(t)
determine fitness of population(t)
repeat
    select parents from population(t)
    perform crossover on parents creating population(t+1)
    perform mutation of population(t+1)
    determine fitness of population(t+1)
until best individual is good enough

Perform Crossover

Before Crossover	
S1' = 1111010101 S2' = 1110110101	S5' = 0100010011 S6' = 1110111101
After Crossover	
S1' = 1110110101 S2' = 1111010101	S5" = 0100011101 S6" = 1110110011

Basic Algorithm of Genetic Algorithm

randomly initialize population(t)
determine fitness of population(t)
repeat
    select parents from population(t)
    perform crossover on parents creating population(t+1)
    perform mutation of population(t+1)
    determine fitness of population(t+1)
until best individual is good enough

Mutation

Before applying mutation:	After applying mutation:
s1” = 1110110101	s1”’ = 1110100101
s2” = 1111010101	s2”’ = 1111110100
s3” = 1110111101	s3”’ = 1110101111
s4” = 0111000101	s4”’ = 0111000101
s5” = 0100011101	s5”’ = 0100011101
s6” = 1110110011	s6”’ = 1110110001

Basic Algorithm of Genetic Algorithm

Clip slide

randomly initialize population(t)
determine fitness of population(t)
repeat
    select parents from population(t)
    perform crossover on parents creating population(t+1)
    perform mutation of population(t+1)
    determine fitness of population(t+1)
until best individual is good enough

Fitness of New Population

After Applying Mutation	
s1‴ = 1110100101	f(s1‴) = 6
s2‴ = 1111110100	f(s2‴) = 7
s3‴ = 1110101111	f(s3‴) = 8
s4‴ = 0111000101	f(s4‴) = 5
s5‴ = 0100011101	f(s5‴) = 5
s6‴ = 1110110001	f(s6‴) = 6
	Total = 37

Example (End)

- In one generation, the total population fitness changed from 34 to 37, thus improved by ~9%.
- At this point, we go through the same process all over again, until a stopping criteria is met.

Issues

Choosing basic implementation issues:

- representation
- population size, mutation rate, ...
- selection, deletion policies
- crossover, mutation operators

Termination Criteria

Performance, scalability

Solution is only as good as the evaluation function

When to Use a GA

Alternate solutions are too slow or overly complicated

Need an exploratory tool to examine new approaches

Problem is similar to one that has already been successfully solved by using a GA

Want to hybridize with an existing solution

Application : Job Scheduling

Applying Genetic Algorithm

- Represent a solution as a chromosome.

  -We have already chosen a representation like {1,0,1}.

- Define a fitness function.

  $f(solution) = [concat(G_1, G_2, G_3)]_{10}$ *G represents a gene*

  $f(\{1,0,1\}) = (101)_{10} = 5$

- Select a population of the solution and calculate their fitness values.

Solution	Fitness
{0,0,1}	1
{0,1,0}	2
{1,1,0}	6
{1,0,1}	5

For a single set, 4 solutions are obtained

Applying Reproduction

- Suppose, we randomly generate 500 sets of such solutions.
- Resulting population will have 2000 chromosomes.
- Among the 2000 chromosomes, the one having the maximum fitness value will be having the maximum appearance like as follows:

  {0,0,1},{0,1,0},{1,1,0},{1,0,1},{1,1,0},{0,1,0},{1,1,0},{1,0,1},...,

  {1,0,1},{0,1,0},{1,1,0},{0,1,0},{0,0,1},{1,1,0},{1,1,0},{0,1,0},...,

  {1,0,1},{1,1,0},....., up to 2000th chromosome.

For 500 sets, 500 x 4 = 2000 solutions are obtained

Job Shop Scheduling problem

- Suppose there are three manufacturing units M1, M2 and M3, each of which is capable of manufacturing a different product

- A schedule could be a set of numbers 20,35,15 where each number indicates the quantity of the concerned product manufactured by M1, M2, and M3 respectively

- The schedule could yield a profit say 40 computed by some equation

- The problem is to optimize (maximize) the profit
- The function used to compute the profit is called the fitness function

Job Shop Scheduling problem contd..

- The set {20,35,15} forms the chromosome; its members form the genes, Alleles are the values that these genes can assume

- The problem can be simplified in terms of binary numbers by assuming a unit is ON (manufacturing) or OFF (not manufacturing)

Job Shop Scheduling problem contd..

- A schedule { 1,0,1} yields a chromosome that indicates that M1 and M3 are on and M2 is off

The fitness function could be the binary number formed from the individual genes in the chromosome; for example the schedule {1,0,1} has fitness of $(101)_2$ or decimal 5

The problem is to find the best schedule to maximize the value of the numbers formed by the genes comprising the chromosome

The solution {1,1,1} could be found using genetic algorithm approach

Use Genetic Algorithm to solve Job Scheduling Problem

Using GA to solve Job Shop Scheduling problem

1. Represent a solution as a chromosome and find a suitable fitness function

   Define the chromosomes like {1,0,1}, {1,1,0} etc., and define the fitness function as

   $f(\text{chromosome}) = (\text{concat}(\text{Gene1},\text{Gene2}, \text{Gene3}))_{10}$

2. Choose a population of solutions and initialize it

   Let the population size N be 4; let the initialized population be [{0,0,1}, {0,1,0}, {1,1,0}, {1,0,1}]

3. Apply the genetic operators one by one

Using GA to solve Job Shop Scheduling problem contd..

4 . Determine the fitness of each solution fi and the total fitness of the population $\sum$fi as follows

Compute the normalized fitness or fitness probability pi, and the expected count of the solution (number of solutions that could occur) E_c

The computed values are shown in Table 1.

Using GA to solve Job Shop Scheduling problem contd..

Solution No.	Solution	fi=concat(Gene1,Gene2, Gene3))$_{10}$	%pi=fi/ ∑fi *100	E_c=N*%pi (N=4)
1	{0,1,0}	1	7.14	28.6
2	{0,1,0}	2	14.29	57.1
3	{1,1,0}	6	42.86	171.4
4	{1,0,1}	5	35.71	142.9
∑		14	100	400
Maximum		6	42.86	171.43

Table 1: Fitness and Expected Count of a generation

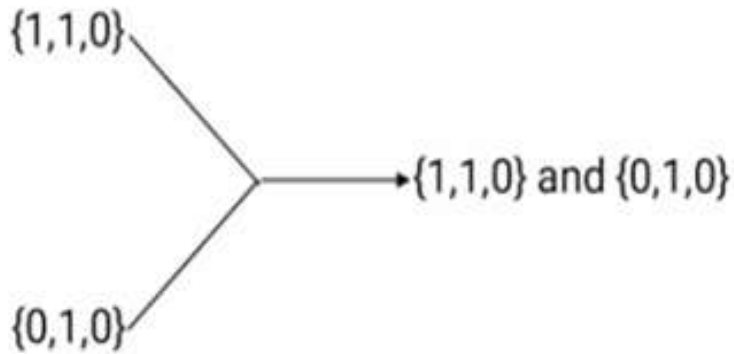
Solution	Fitness
{0,0,1}	1
{0,1,0}	2
{1,1,0}	6
{1,0,1}	5

Applying Crossover

- Let's select two solutions randomly from 2000 chromosomes.

  (a) {1,1,0} and {0,1,0}

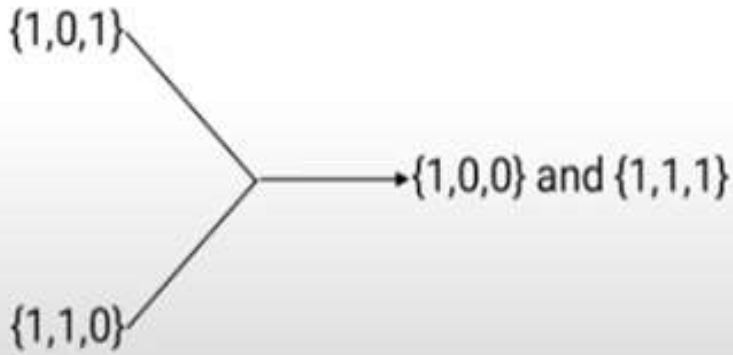
  (b) {1,0,1} and {1,1,0}



- Randomly choose a crossing site.

  1st bit for pair a

  0th bit for pair b



Applying Crossover

- Let's select two solutions randomly from 2000 chromosomes.

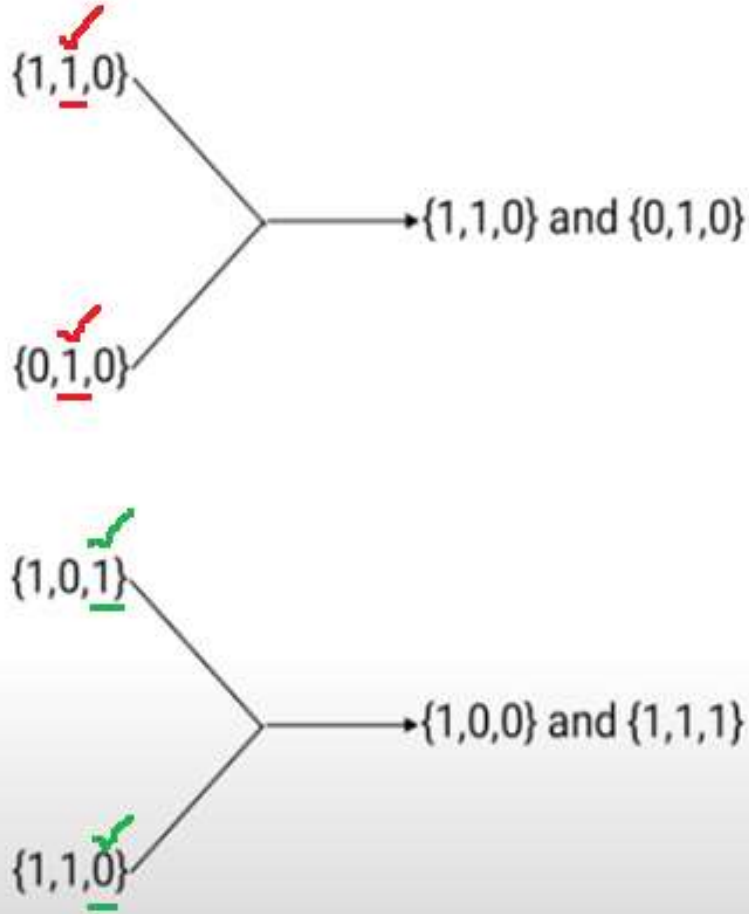
  (a) {1,1,0} and {0,1,0}

  (b) {1,0,1} and {1,1,0}

- Randomly choose a crossing site.

  1st bit for pair a

  0th bit for pair b



Applying Crossover

- Let's select two solutions randomly from 2000 chromosomes.

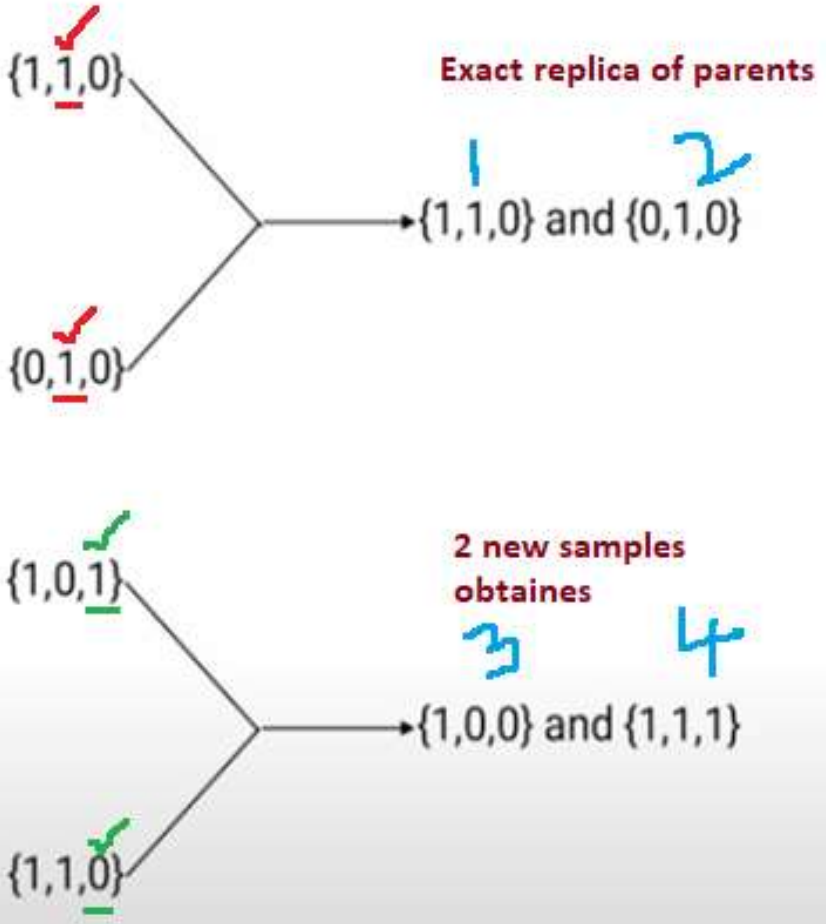
  (a) {1,1,0} and {0,1,0}

  (b) {1,0,1} and {1,1,0}

- Randomly choose a crossing site.

  1st bit for pair a

  0th bit for pair b



4 chromosomes are obtained by using Cross Over Operator

Solution	Fitness
{1,1,0}	6
{0,1,0}	2
{1,0,0}	4
{1,1,1}	7

NOTE: Slide No: 27, WKT,

The problem is to find the best schedule to maximize the value of the numbers formed by the genes comprising the chromosome

The solution {1,1,1} could be found using genetic algorithm approach

Solution	Fitness
{1,1,0}	6
{0,1,0}	2
{1,0,0}	4
{1,1,1}	7

Generated Solution is a better solution which is obtained with Cross Over Operation itself.

It is better than the previously existing solutions.
In the human perspective, it is the optimal solution because we can not have a larger value than {1 1 1} (better solution). So Crossover is successful.

This is the fitness values of the newly generated solutions.

Applying genetic operators

Reproduction or selection operator is applied on the mating pool and two solutions are selected. There are {1,0,1} and {1,1,0}

(c) Crossover is applied on {1,0,1} and {1,1,0} choosing crossing site 2 and the resultant chromosomes are {1,0,0} and {1,1,1}

(d) Mutation occurs rarely. Mutation flips the value of a randomly selected gene in a chromosome

For example the chromosome {0,1,0} could mutate to any of {1,1,0} or {0,0,0} or {0,1,1}

Applying genetic operators

Reproduction or selection operator is applied on the mating pool and two solutions are selected. There are {1,0,1} and {1,1,0}

(c) Crossover is applied on {1,0,1} and {1,1,0} choosing crossing site 2 and the resultant chromosomes are {1,0,0} and {1,1,1}

(d) Mutation occurs rarely. Mutation flips the value of a randomly selected gene in a chromosome

For example the chromosome {0,1,0} could mutate to any of {1,1,0} or {0,0,0} or {0,1,1}

Applying Mutation

- Let’s select a chromosome {0,1,0} to be mutated.

- After flipping either of the three bits, we can have the following possible chromosomes:

  {0,1,1}

  {0,0,0}

  {1,1,0}

Solution	Fitness
{0,1,1}	3
{0,0,0}	0
{1,1,0}	6

- Compared to first set(population) of solutions,
- After applying MUTATION, we have not read any better solution.
- So for this set of chromosome {0,1,0}, we can not generate a better solution
- It is resulting in the almost same fitness value

- By the application of Crossover we have reached the optimal solution for the JOB

Conclusion

Inspired from Nature

Has many areas of Applications

GA is powerful