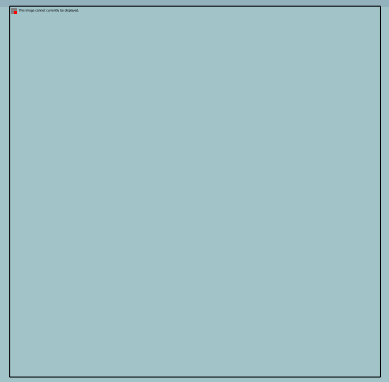
Genetic Algorithm

...

In the field of artificial intelligence...

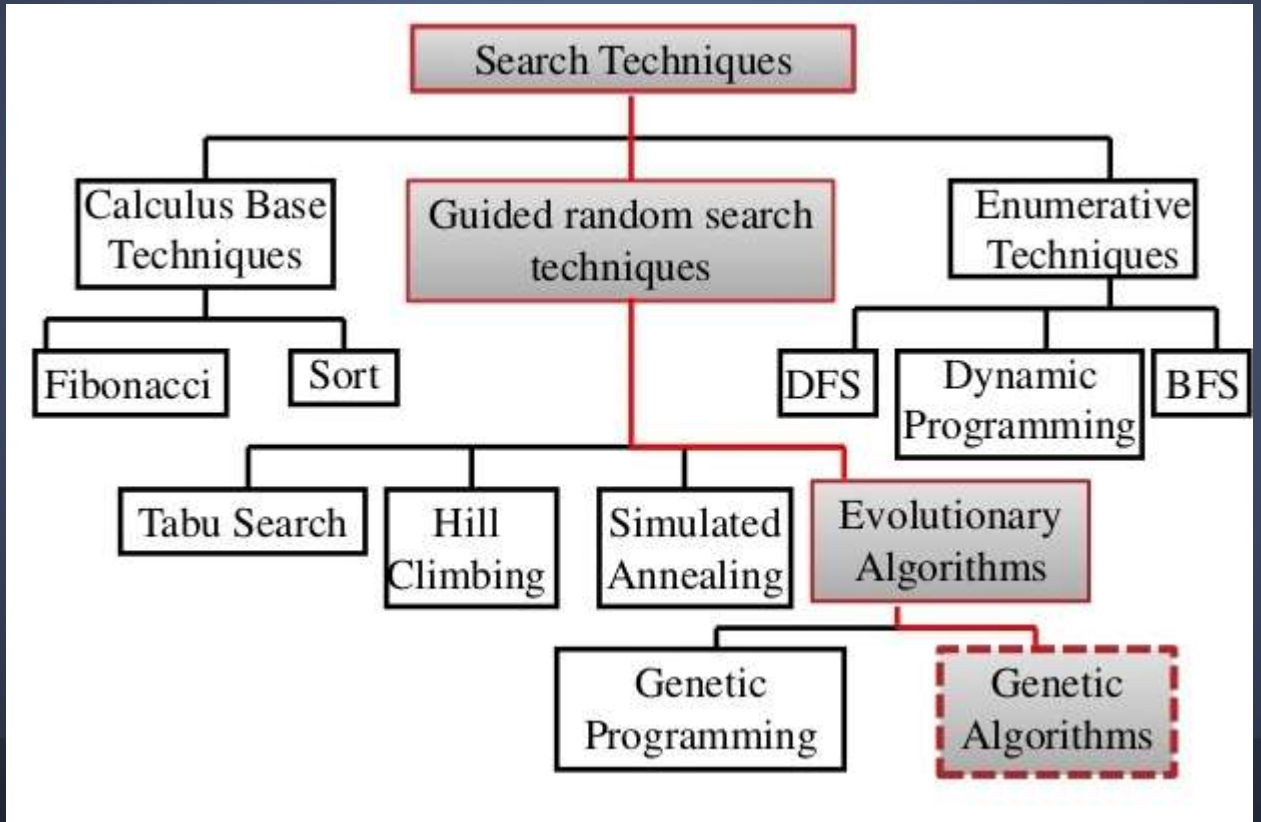
Presenters’



Abu Sayed
ID : 011 111 009

Tauhidul Khandaker
ID : 011 111 010

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

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.*

    3. *[Mutation] With a mutation probability mutate new children at each position in chromosome*

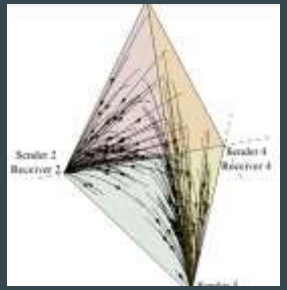
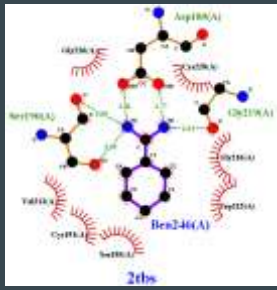
Next Page To Continue……

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.
8. Game theory equilibrium resolution.



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

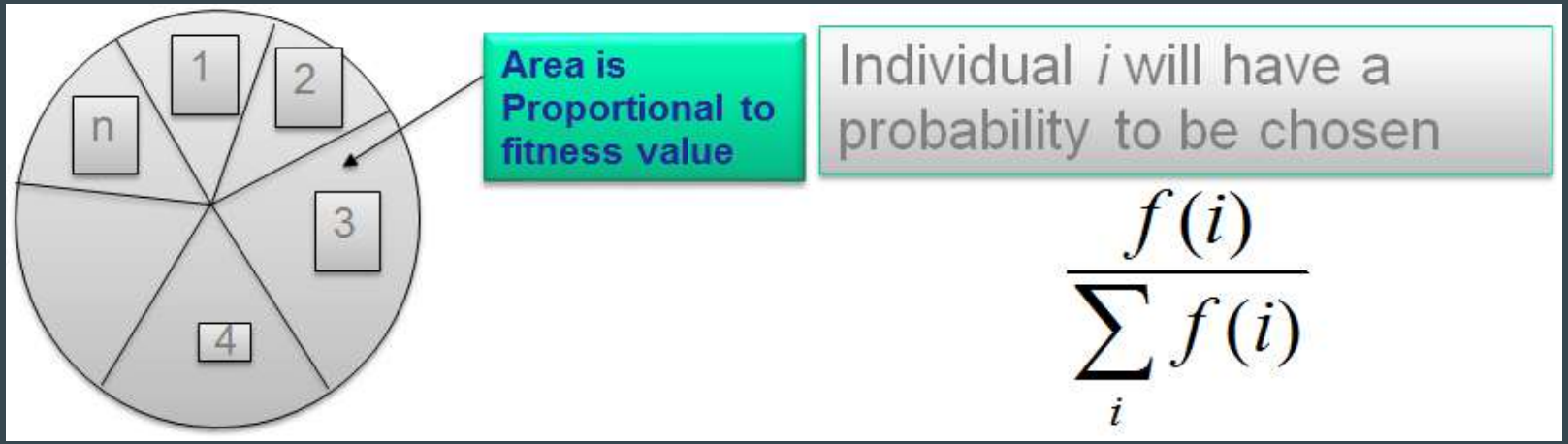
s6 = 0100110000

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:



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)

s5’ = 0100010011 (s4)

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 s2” = 1111010101 s3” = 1110111101 s4” = 0111000101 s5” = 0100011101 s6” = 1110110011	s1”’ = 1110100101 s2”’ = 1111110100 s3”’ = 1110101111 s4”’ = 0111000101 s5”’ = 0100011101 s6”’ = 1110110001

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

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

Conclusion

Inspired from Nature

Has many areas of Applications

GA is powerful


Thank
You
Merci
Gracias
TerimaKasih
Hvala
Arigatô
KamSahHamnida
Spacibo
Danke
Mèsi
Xièxie
Chokrane
Kiitos
Takk
aDank
BarakAllahuFiik
Grazie
Grazzi
Dziakuju
GratiasAgimus
GratiasAgo
Tak
Dankie
Motashakkeram
KopKhunKha
Gracie
CámOn
Multumesc
Saha
Salamat
Obrigado
Bayarlalaa
Nirringrazzjak
NäjisTuke
Efharisto
Faleminderit
Gràcies
Grandmercé
Tanemirt
Taiku
Mercé
Matondo
Blagodaram
ThintKo
Buznyg
WaadMahadsantahay
TapadhLeat
Köszönöm
Murakoze
Mauruuru
Rahmat
Tenki
Aabhar
Barkal
Chnorakaloutioun
TänanVäga
Ačiū
Sagolun
Dakujem
Gratzias
KhobChaiDeu
Mamnun
aNiKie