Evolutionary Computation

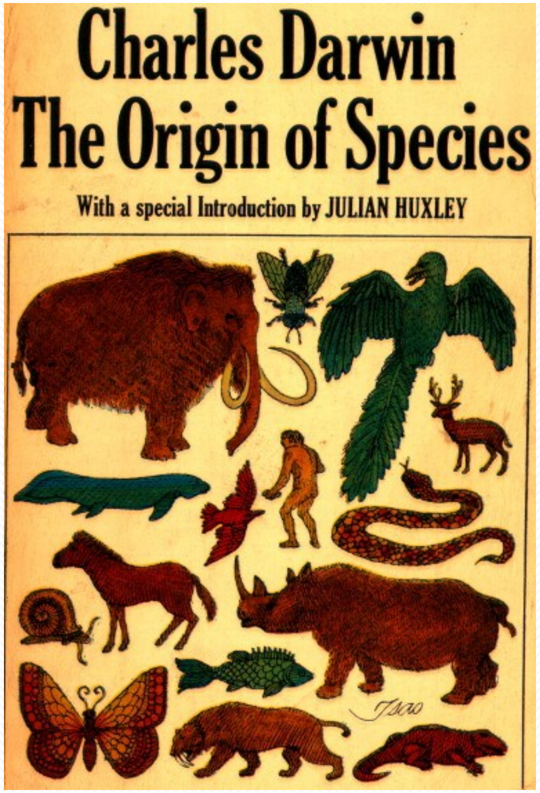
Asst.Prof. Dr.Supakit Nootyaskool
IT-KMITL ©2015

1.Overview GA

Darwin’s theory
Survival
History of Evolutionary computation
Optimization

Introduction to Genetic algorithm

- Genetic algorithms are a technique to solve problems by finding an optimum solution
- GA's are a subclass of Evolutionary Computation
- GA's are based on Charles Darwin's theory of evolution



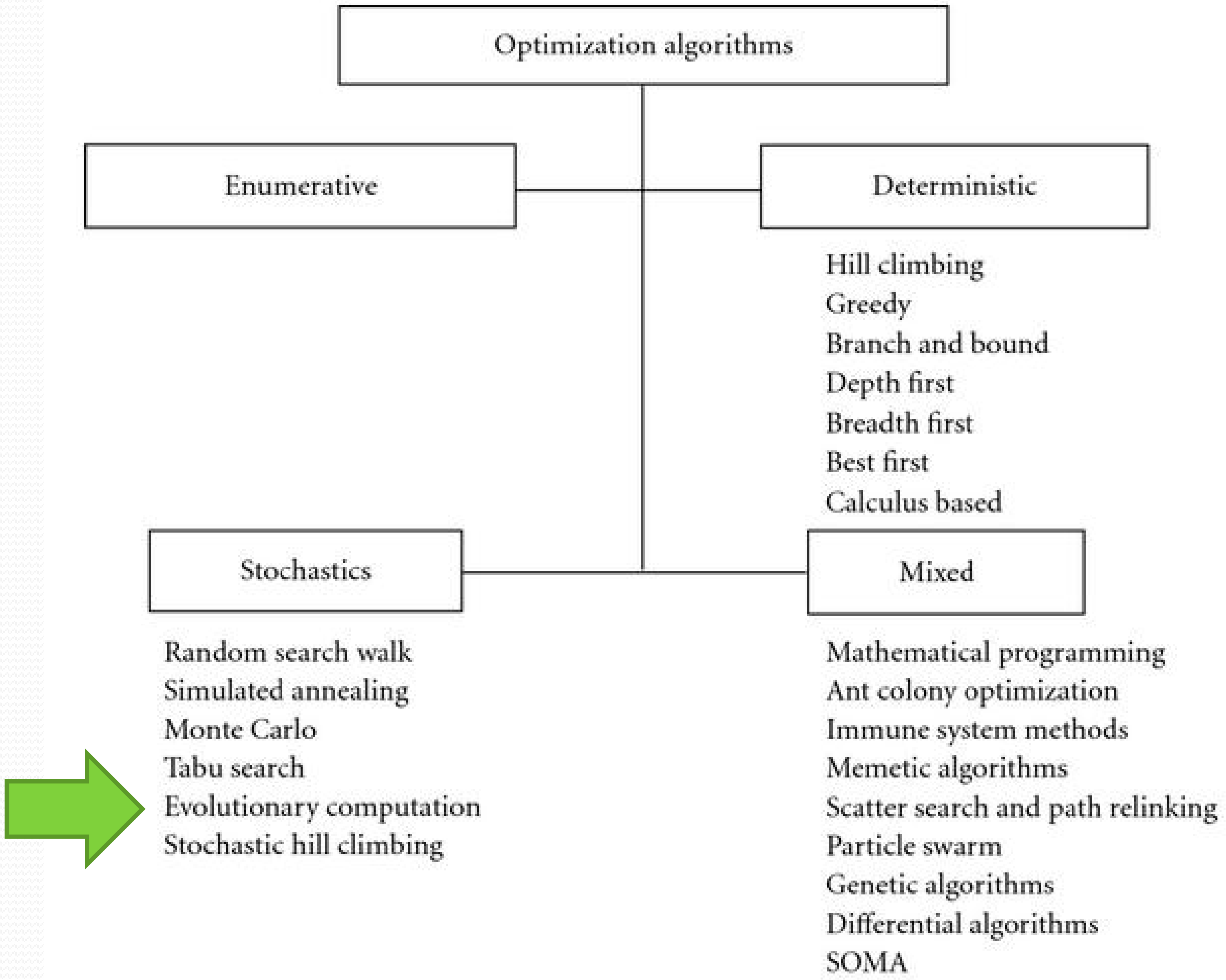
Introduction to Genetic algorithm

- The origin of species: "Preservation of favorable variations and rejection of unfavorable variations."
- There are more individuals born than can survive by self-adapt to a different environment.
- Environment changing -> Adaptation -> Survival

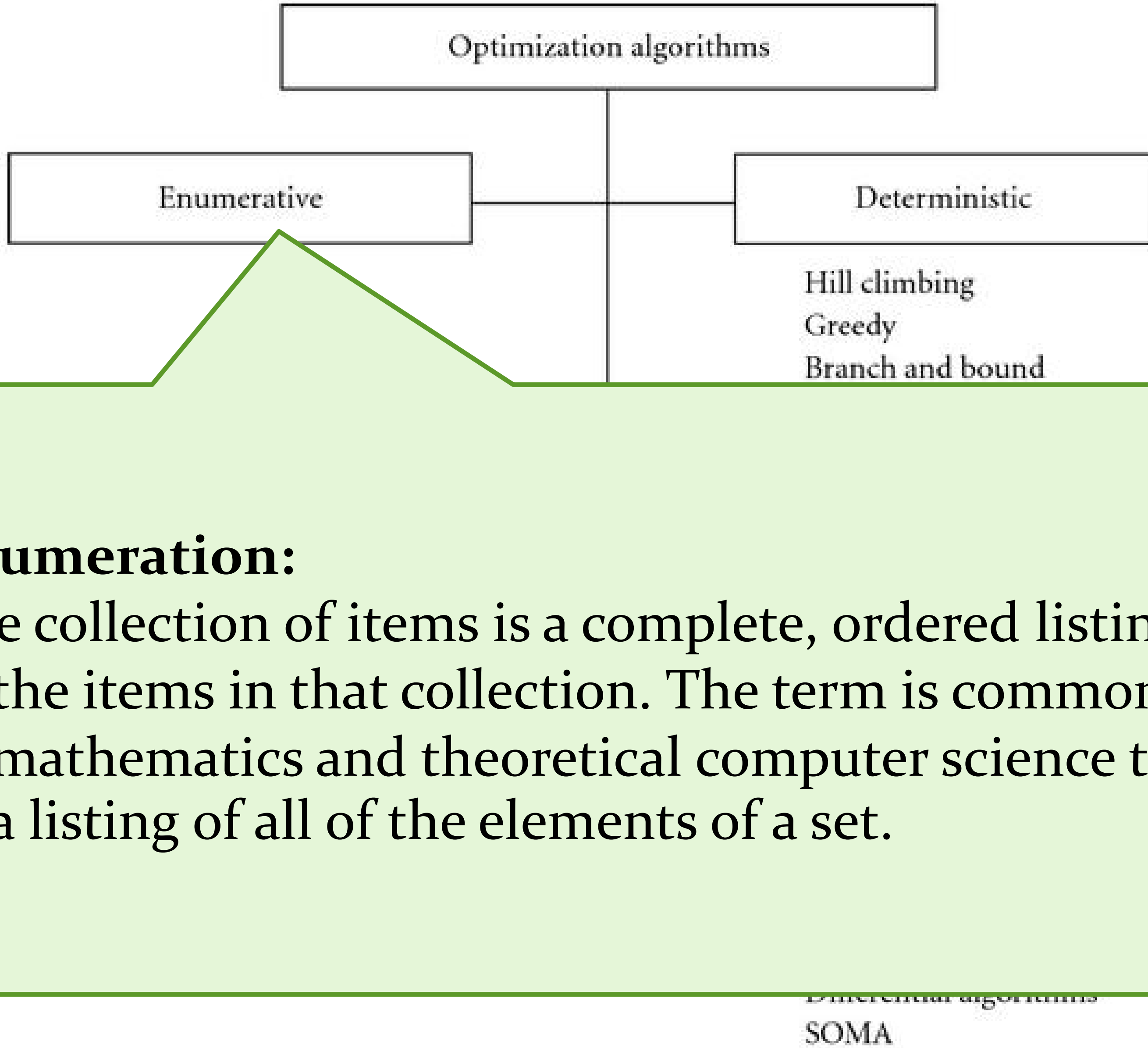
History of Evolutionary Computations

- 1948, Turing:
  proposes "Genetically or evolutionary search"
- 1962, Bremermann
  optimization through evolution and recombination
- 1964, Rechenberg
  introduces evolution strategies (ES)
- 1965, L. Fogel, Owens and Walsh
  introduce evolutionary programming (EP)
- 1975, John Holland
  introduces genetic algorithms (GA)
- 1992, Koza
  introduces genetic programming (GP)

Understand Optimization Algorithms

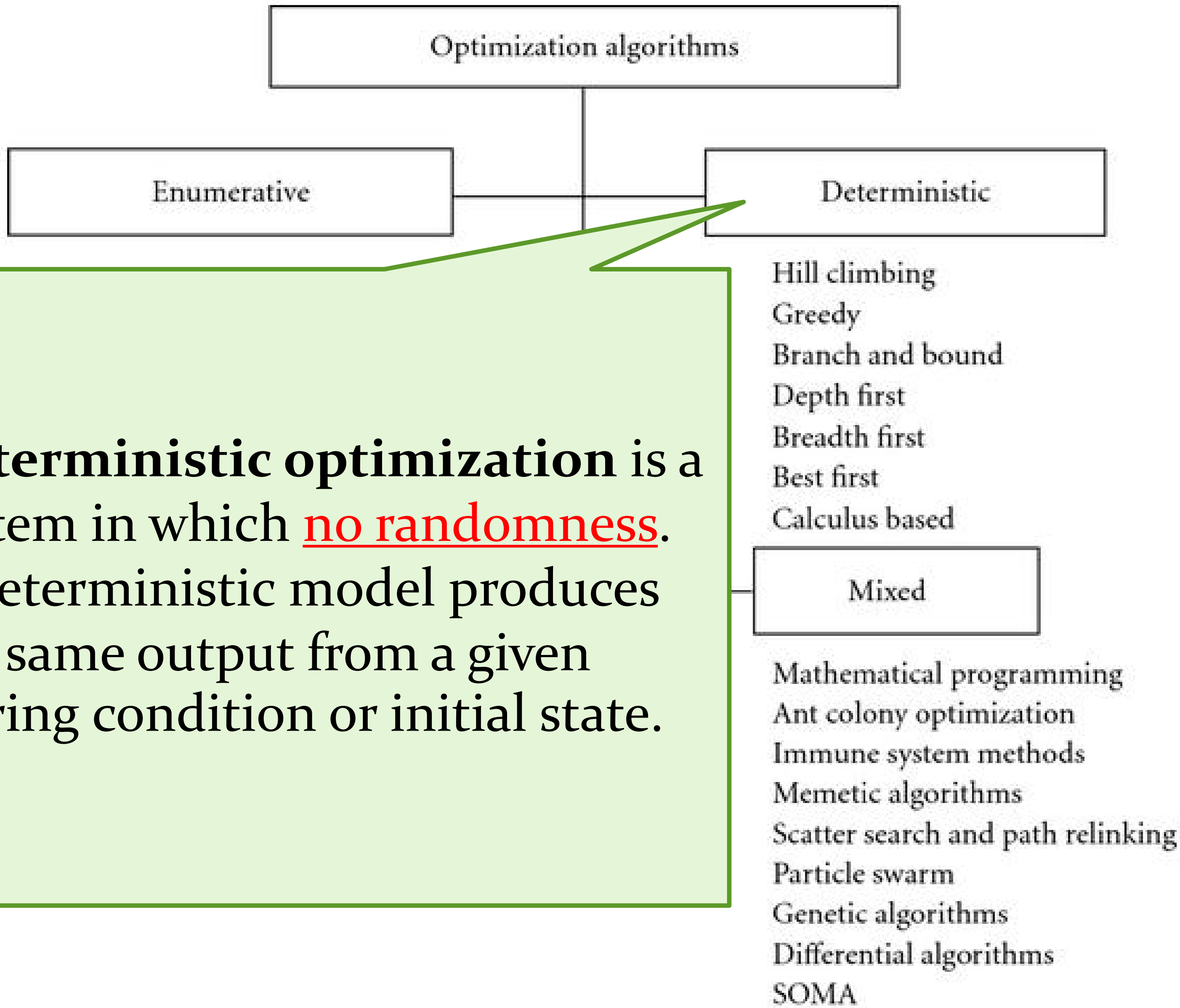


Understand Optimization Algorithm



Enumeration:
The collection of items is a complete, ordered listing of all of the items in that collection. The term is commonly used in mathematics and theoretical computer science to refer to a listing of all of the elements of a set.

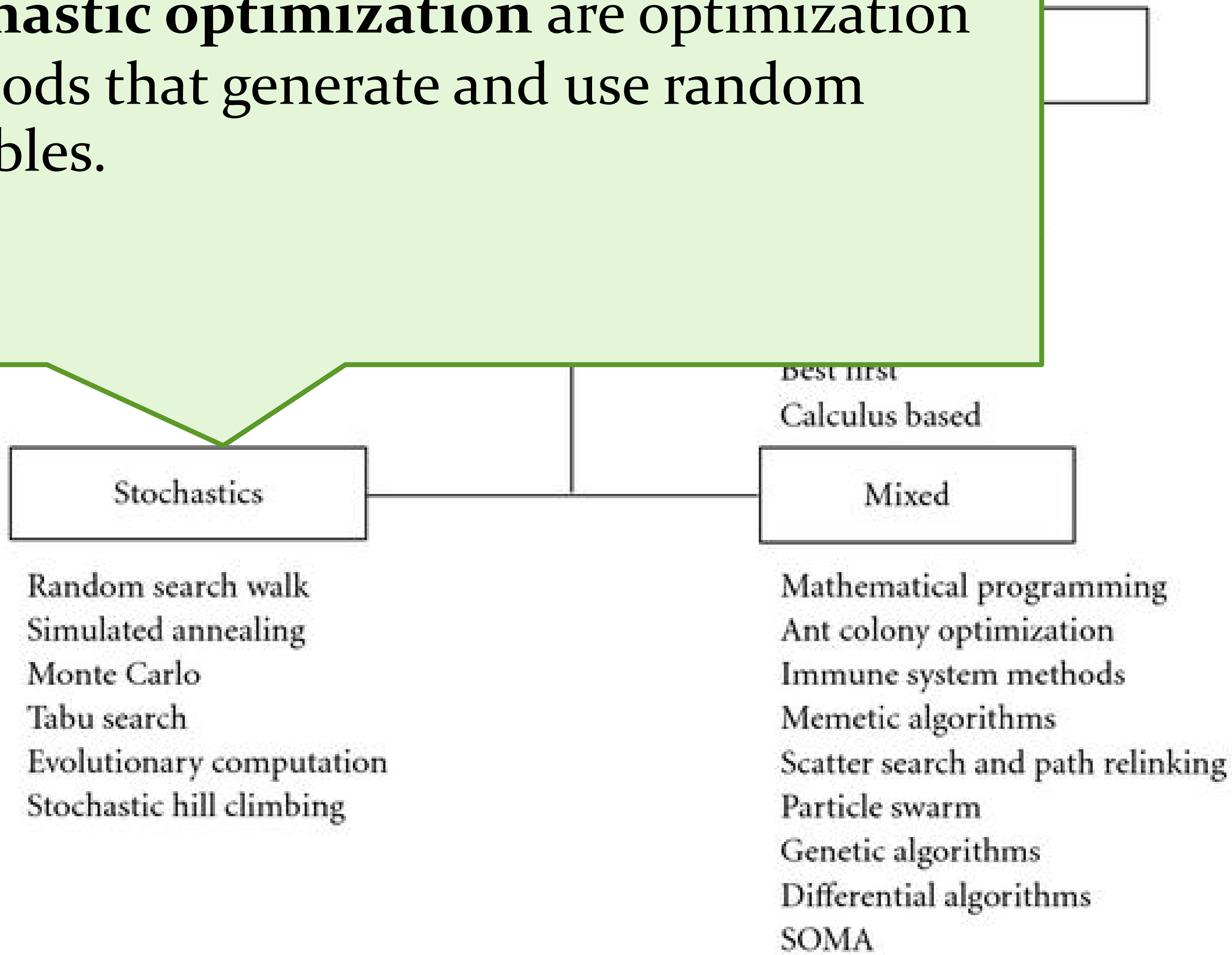
Understand Optimization Algorithm



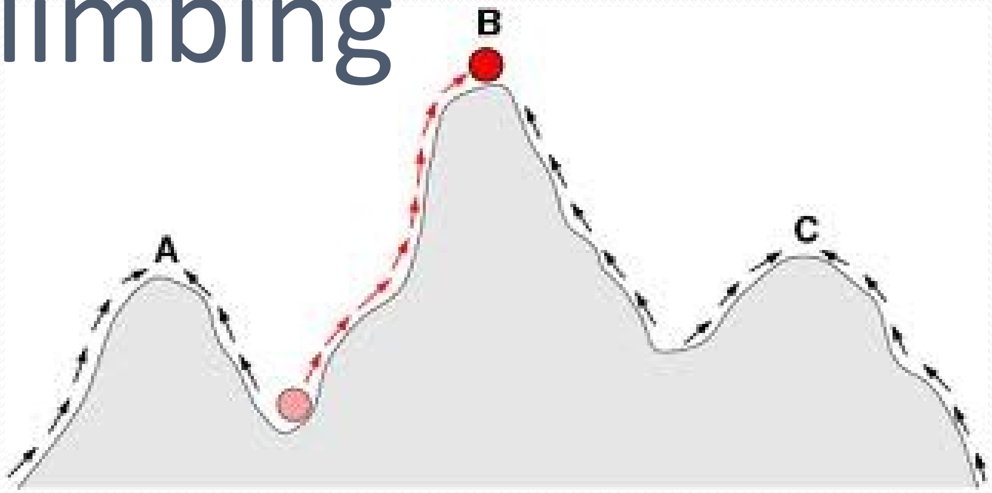
Deterministic optimization is a system in which no randomness. A deterministic model produces the same output from a given staring condition or initial state.

Understand Optimization Algorithm

Stochastic optimization are optimization methods that generate and use random variables.

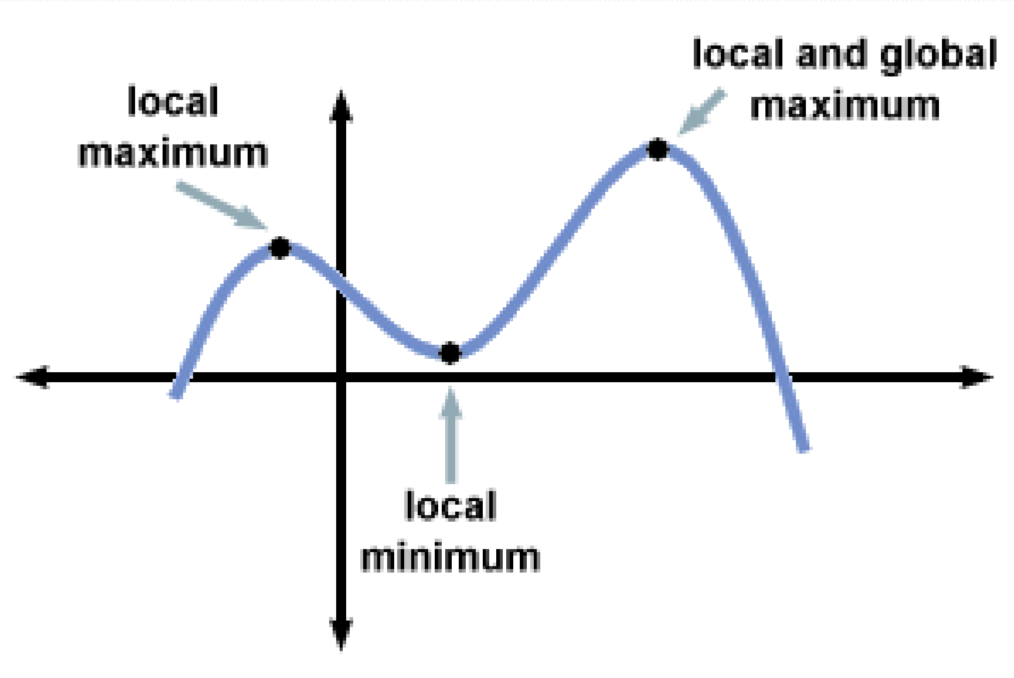


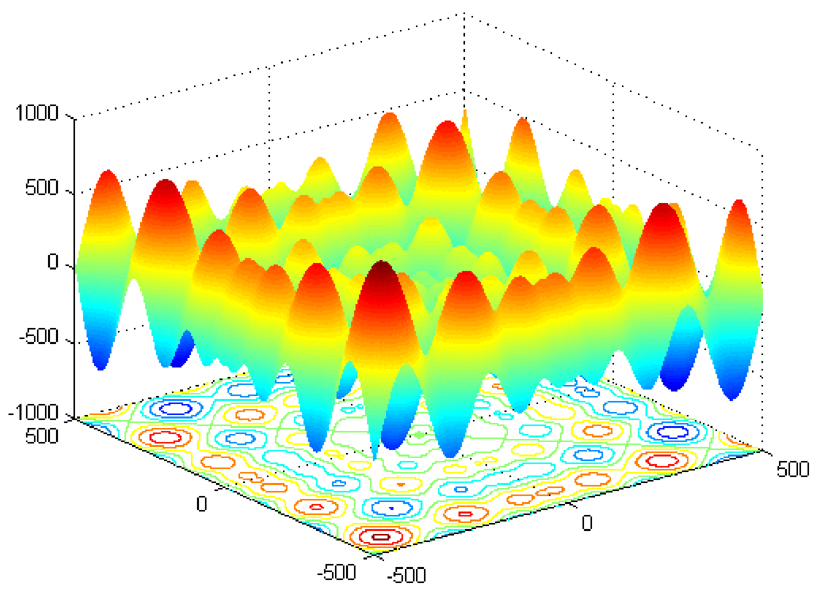
Hill Climbing



- The strategy of the Hill Climbing is iterate the process of select a neighbor for a candidate solution and only accept it if it results in an improvement. The strategy was proposed to address the limitations of deterministic hill climbing techniques that were likely to **get stuck in local optima** due to their greedy acceptance of neighboring moves.

- Local minima / Local maxima (Local solution)
- Global minimum / Global maximum (Optimum solution)





2.Evolutionary Computation Technique

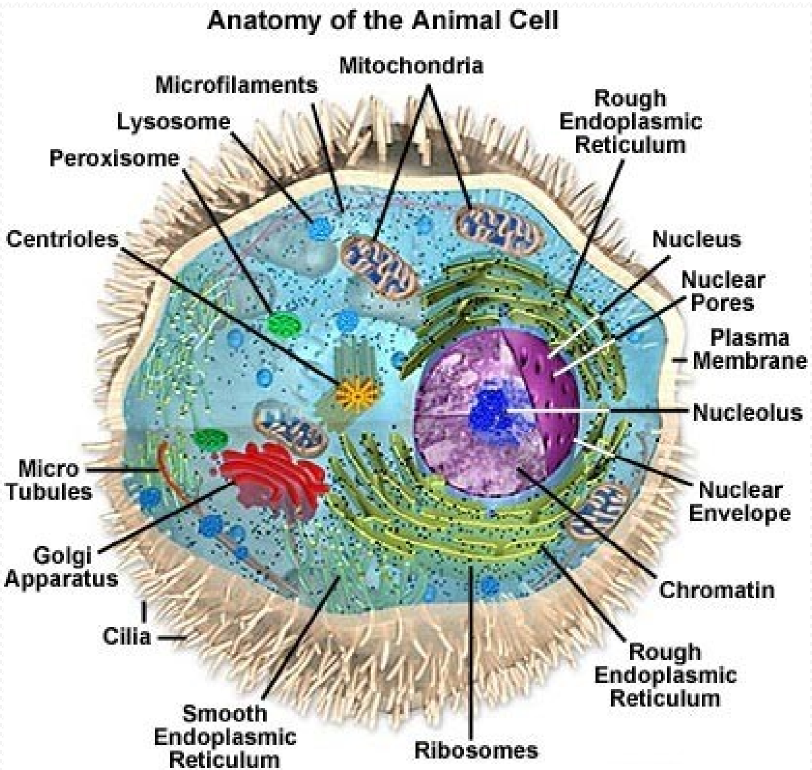
An example GA on peaks function

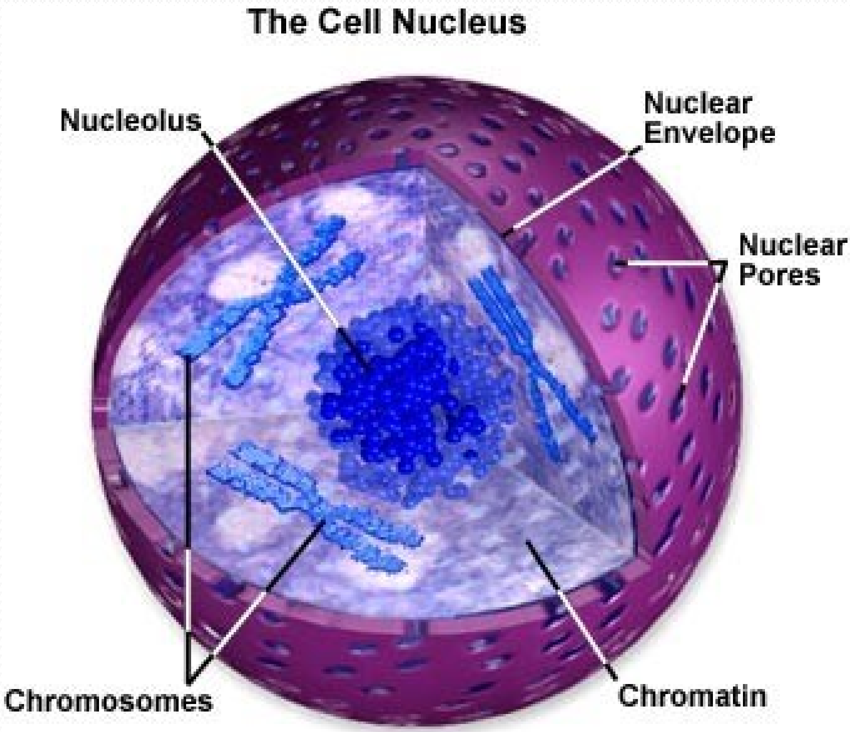
3.Background of GA

Biological background
Genotype
Phenotype
Crossover & Mutation
Selection
Chromosome representation

Biological background (1)

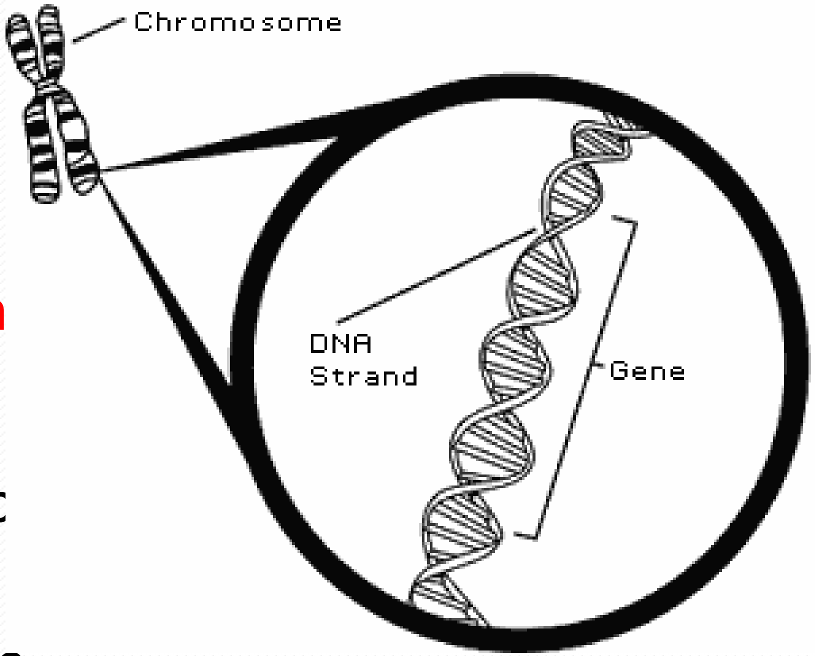
- Every animal cell is a complex of many small "factories" working together
- The center of this all is the cell nucleus
- The nucleus contains the genetic information

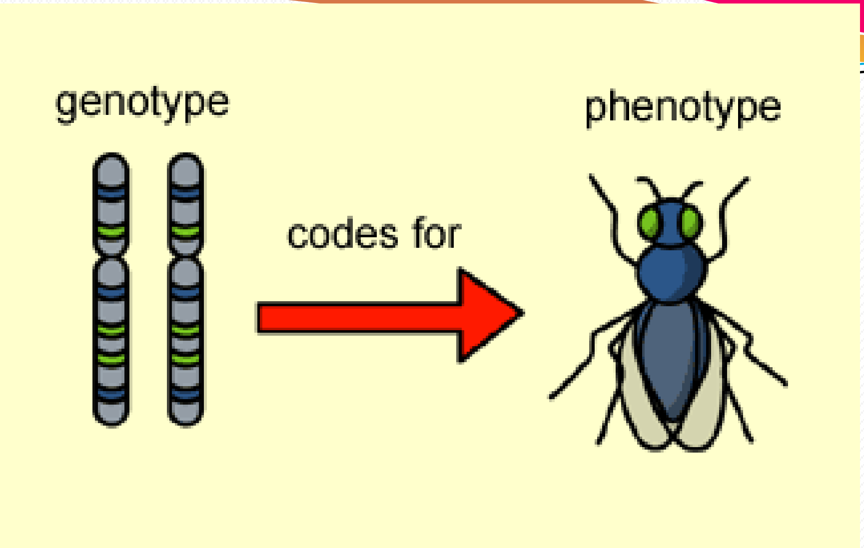




Biological background (2)

- Genetic information is stored in the chromosomes
- Each chromosome is build of DNA
- Chromosomes in humans form pairs
- There are 23 pairs
- The chromosome is divided in parts: gen
- Genes code for properties
- The posibilities of the genes for one prop is called: allele
- Every gene has an unique position on the chromosome: locus

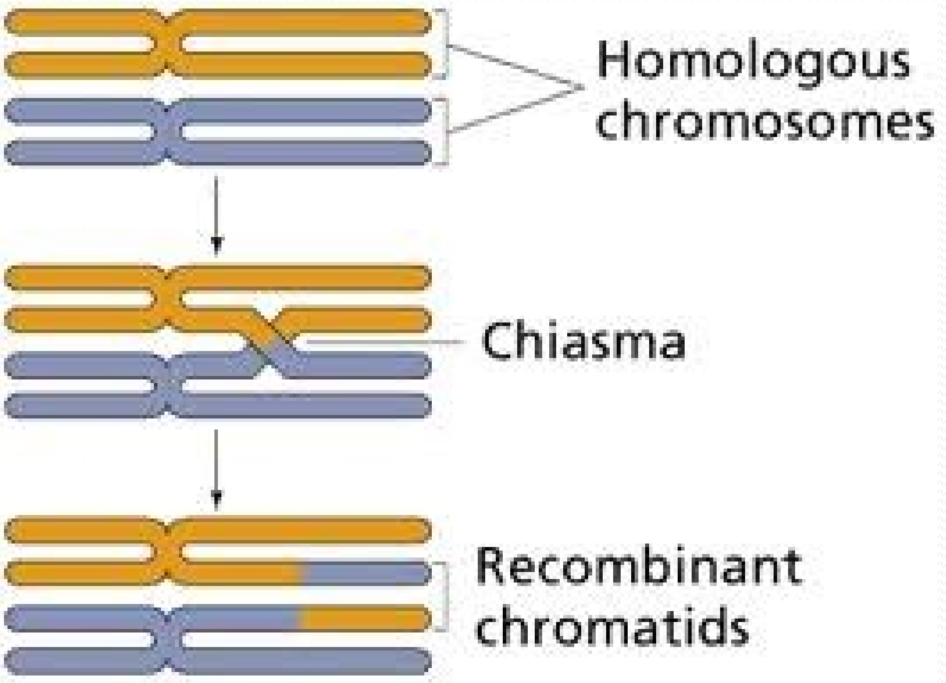




- DNA → Chromosome → Gene
- Genotype, DNA inside is the observable in level of gene or chromosome inside individual.
- Phenotype is the observable characteristics of an individual.

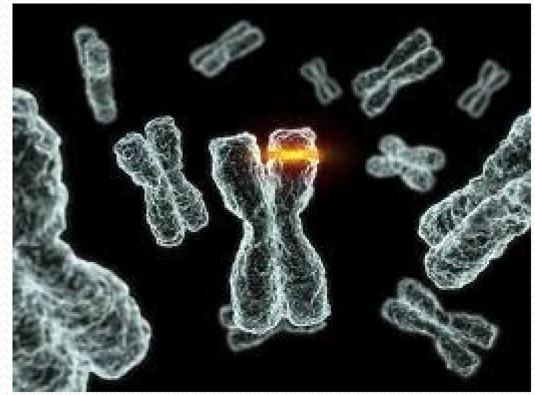
Sexual Recombination

- GA called Crossover
- Crossover is the primary operator to introduce diversity in population



Asexual

- GA called Mutation
- Crossover is the second operator to introduce diversity in population



Genetic Selection techniques

- Uniform selection
- Roulette wheel selection
- Tournament selection
- Random selection

Genetic Selection techniques

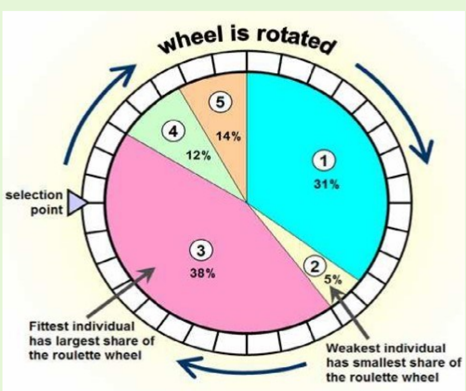
- Uniform selection
- Roulette [illegible]el selection
- Tourna[illegible]tion

Uniform selection: is the selection by using same pattern.
Example:
select: 1 5 8 9 10 23 45

Genetic Selection techniques

- Uniform selection
- Roulette wheel selection
- Tournam … lection

Roulette selection: is random selection applying concept of roulette board.

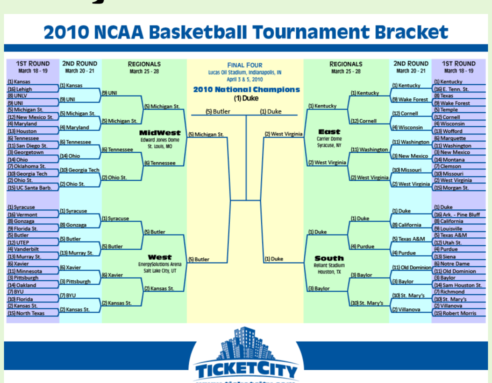


American style double zero roulette wheel.
Photo courtesy www.abbiati.it

Genetic Selection techniques

- Uniform selection
- Roulette wheel selection
- Tournament selection
- Rando

Tournament selection: is the selection by comparing each pair of population. In each pair, the selection may select a population by randomness.



Genetic Selection techniques

- Uniform selection
- Roulette wheel selection
- Tournament selection
- Random selection

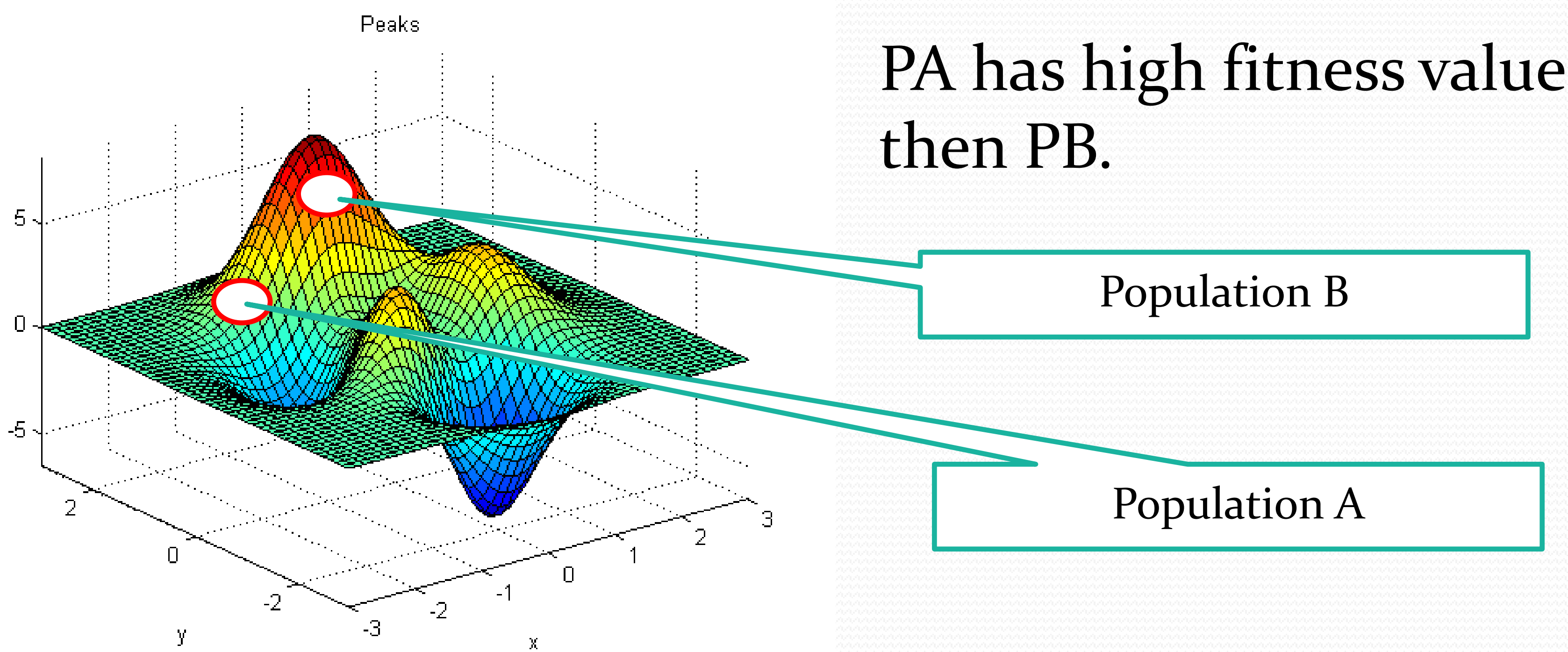
Random selection is the randomness selecting.

Example:	C-code	srand(), rand()
	Excel	random

Fitness value

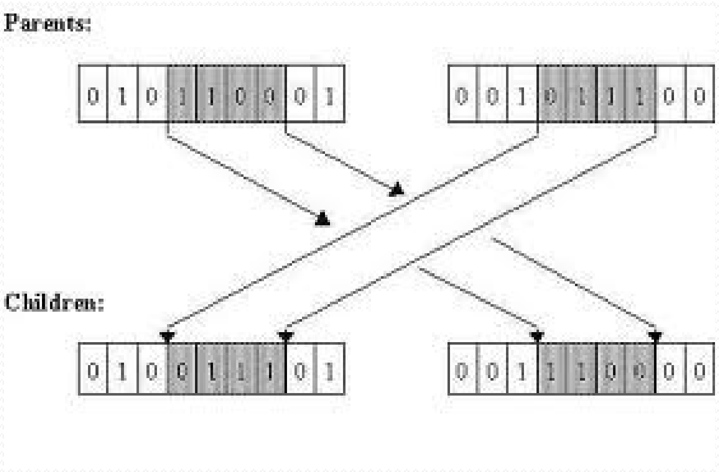
- Fitness value is a value of quality to measure result of solving a problem.

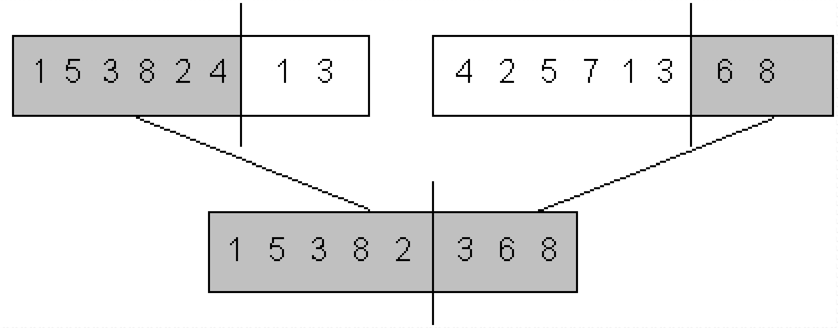
PA has high fitness value then PB.



Chromosome Representation

- Problem representation can be encoding two techniques.
  - Binary chromosome
  - Real number chromosome





Glossary--GA

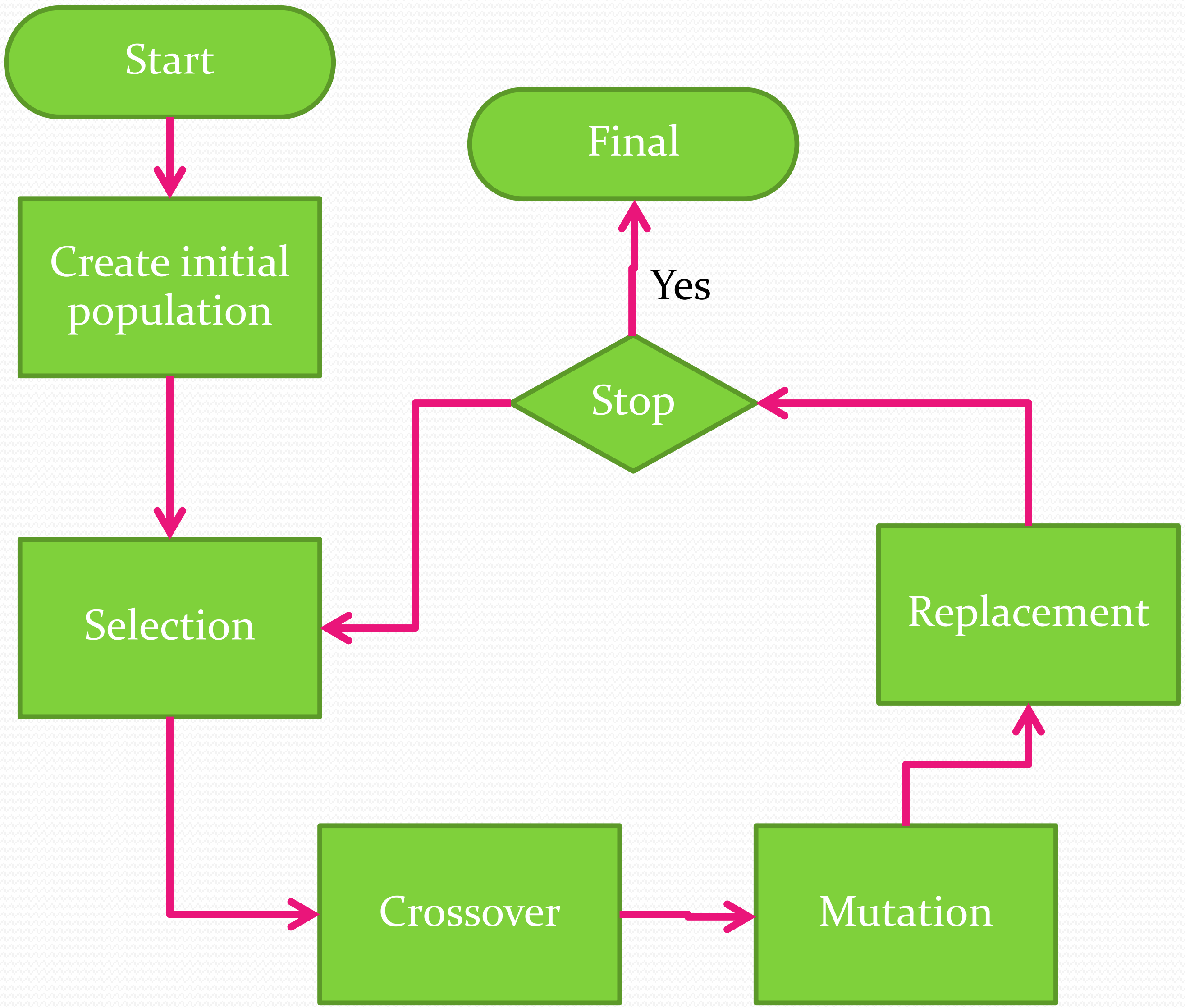
- Evolutionary Computation
  - GA
  - EC
  - EA
  - GP
  - EP
- Selection
  - Roulette wheel
  - Uniform
  - Tournament
  - Random
- Crossover (Recombination)
- Mutation
- Chromosome
  - Gene
- Phenotype ←Genotype
- Fitness
- Populations
  - A population = Individual
  - Old population (Parent)
  - New population (Offspring)
- Encode Chromosome
  - Binary
  - Real
- Generation

4.Start
Genetic Algorithm

Process in Flow chart

Code

GA process


Start
Final
Create initial population
Yes
Stop
Selection
Replacement
Crossover
Mutation

Represent problem

- Problem
  - $z = x^2+y^2$
- What is the value x, y at z =0?
- Step1
  - Encode problem (Example use Binary Chromosome)

Example

X = 9 = 1001

Y = 3 = 0011

Then, a population uses two gene 1001|0011

Create population

- Parents

P1: 1001|0011 x=? y=? z=?

P2: 0101|0110 x=? y=? z=?

P3: 0011|0010 x=? y=? z=?

P4: 1001|0011 x=? y=? z=?

P5: 1101|0010 x=? y=? z=?

Create population

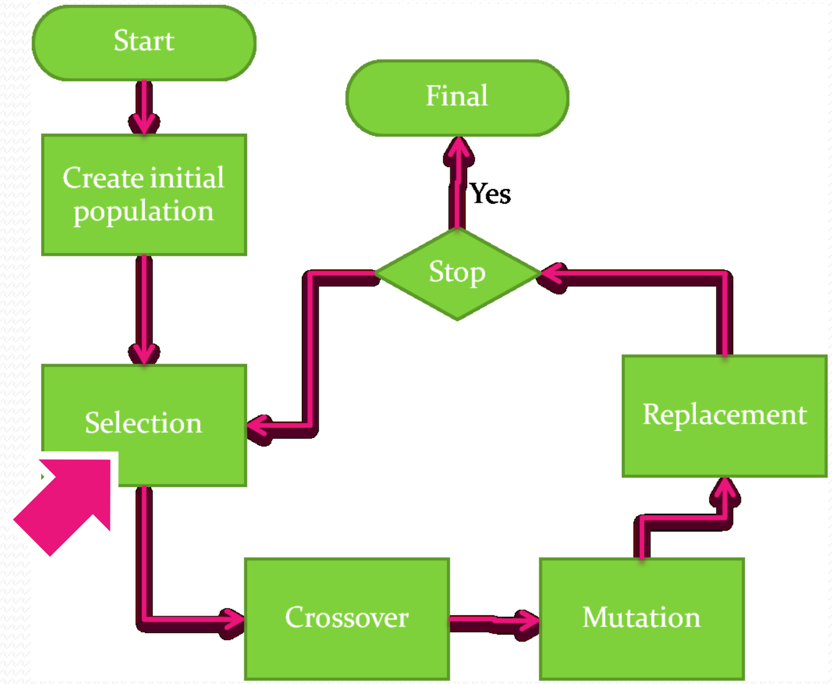
- Parents

P1: 1001|0011	x=9 y=3	z=90
P2: 0101|0110	x=5 y=6	z=61
P3: 0011|0010	x=3 y=2	z=13
P4: 1001|0011	x=9 y=3	z=90
P5: 1101|0010	x=13 y=2	z=139

Which population has best fitness?

Selection

- Use roulette wheel generating 5 offspring



Selection

- Parents

P1: 1001|0011	x=9 y=3	z=90
P2: 0101|0110	x=5 y=6	z=61
P3: 0011|0010	x=3 y=2	z=13
P4: 1001|0011	x=9 y=3	z=90
P5: 1101|0010	x=13 y=2	z=139

Sum z = 393

Which population has best fitness?

Selection

- Parents

P1: 1001|0011	x=9 y=3	z=90/**393**
P2: 0101|0110	x=5 y=6	z=61/**393**
P3: 0011|0010	x=3 y=2	z=13/**393**
P4: 1001|0011	x=9 y=3	z=90/**393**
P5: 1101|0010	x=13 y=2	z=139/**393**

Sum z = 393

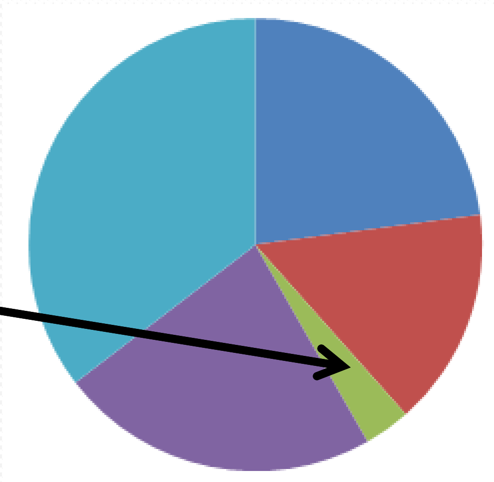
Which population has best fitness?

Selection

- Parents

P1: 1001|0011	x=9 y=3	z=0.229
P2: 0101|0110	x=5 y=6	z=0.1552
P3: 0011|0010	x=3 y=2	z=0.033
P4: 1001|0011	x=9 y=3	z=0.229
P5: 1101|0010	x=13 y=2	z=0.353

Sum z = 393

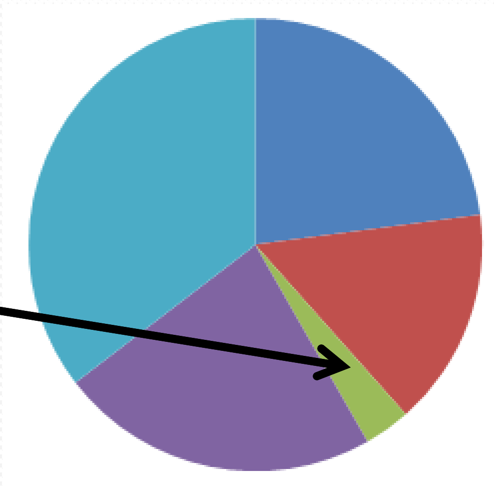


Selection

- Parents

P1: 1001|0011	x=9 y=3	z=0.229
P2: 0101|0110	x=5 y=6	z=0.1552
P3: 0011|0010	x=3 y=2	z=0.033
P4: 1001|0011	x=9 y=3	z=0.229
P5: 1101|0010	x=13 y=2	z=0.353

Sum z = 393

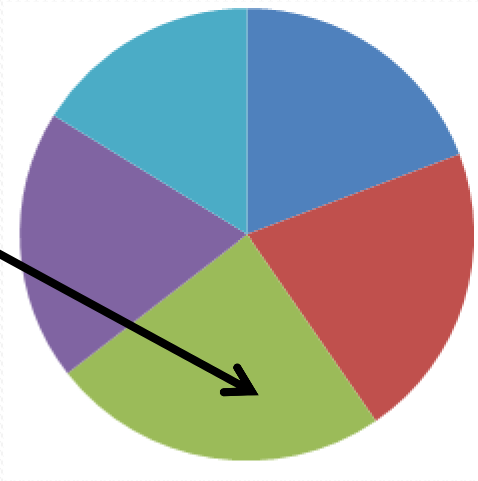


Selection

- Parents

P1: 1001|0011	x=9 y=3	z=1-0.229
P2: 0101|0110	x=5 y=6	z=1-0.1552
P3: 0011|0010	x=3 y=2	z=1-0.033
P4: 1001|0011	x=9 y=3	z=1-0.229
P5: 1101|0010	x=13 y=2	z=1-0.353

Sum z = 393



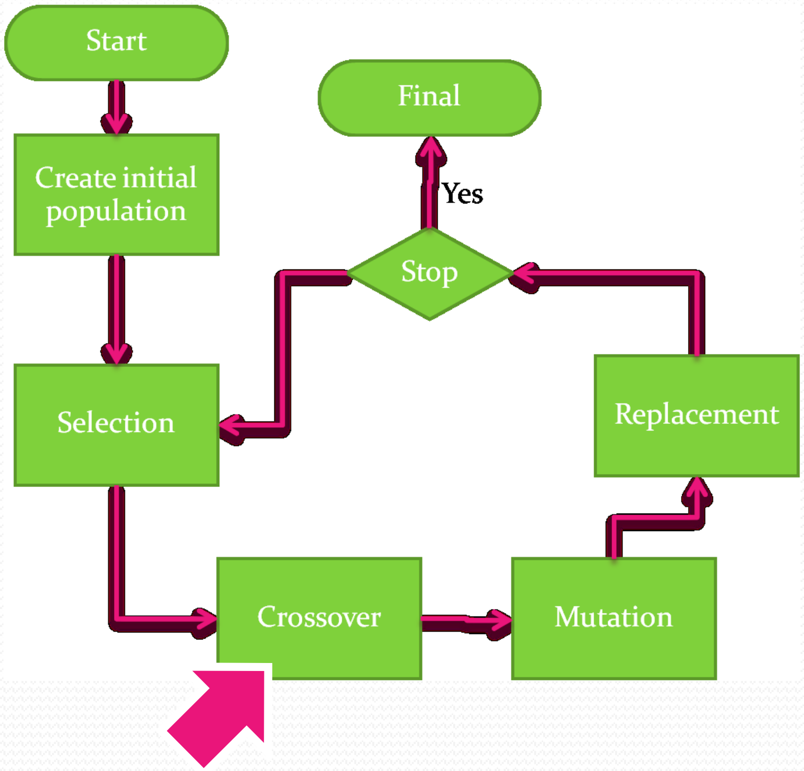
Selection

- Offspring

P6: 0011|0010	x=3 y=2	z=1-0.033
P7: 0101|0110	x=5 y=6	z=1-0.1552
P8: 0011|0010	x=3 y=2	z=1-0.033
P9: 1001|0011	x=9 y=3	z=1-0.229
P10: 0101|0110	x=5 y=6	z=1-0.1552

Crossover

- Crossover techniques
  - One point
  - Two point
  - N-point



Crossover

- Select P6 and P10 by random control with Crossover Probability

P6: 0011|0010 x=3 y=2

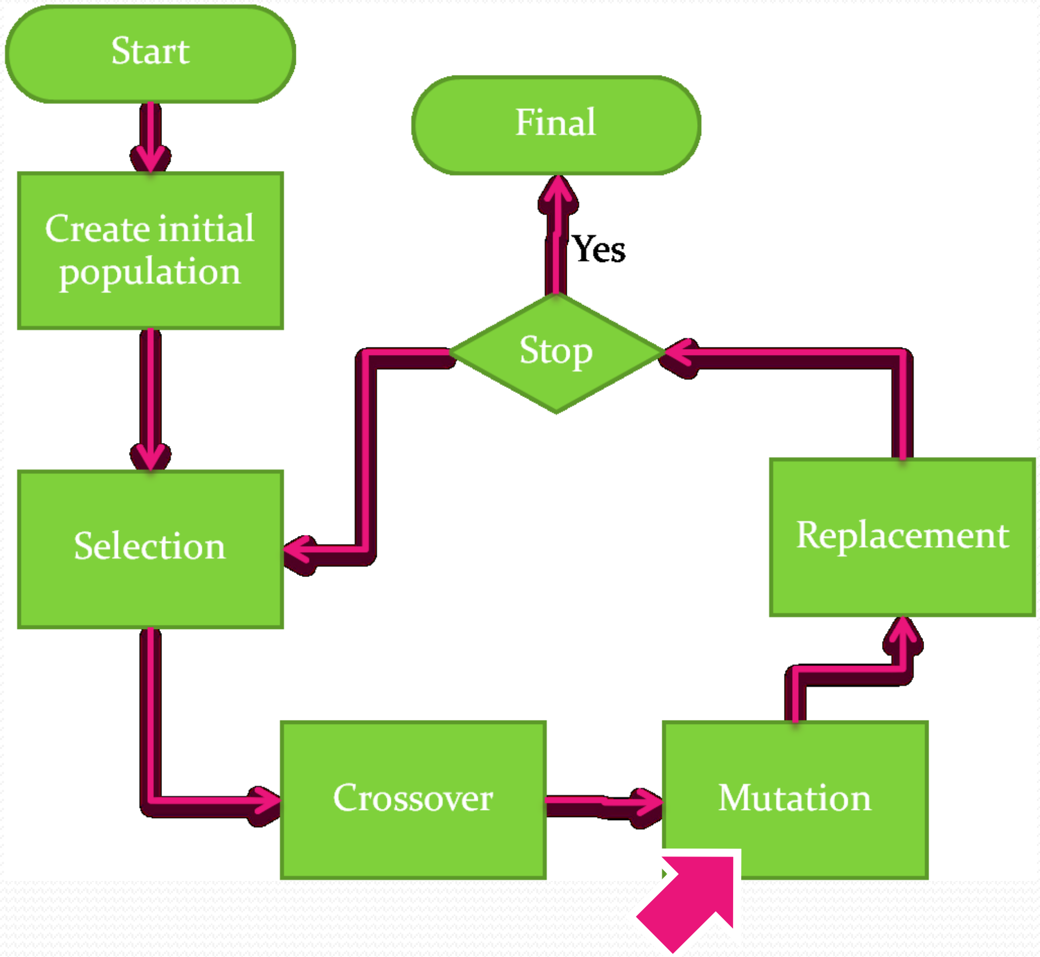
P10: 0101|0110 x=5 y=6

- Random select gene to swap

P6: 0111|0110 x=? y=? z=?

P10: 0001|0010 x=? y=? z=?

Mutation



Start
Create initial population
Final
Yes
Stop
Selection
Replacement
Crossover
Mutation

Crossover

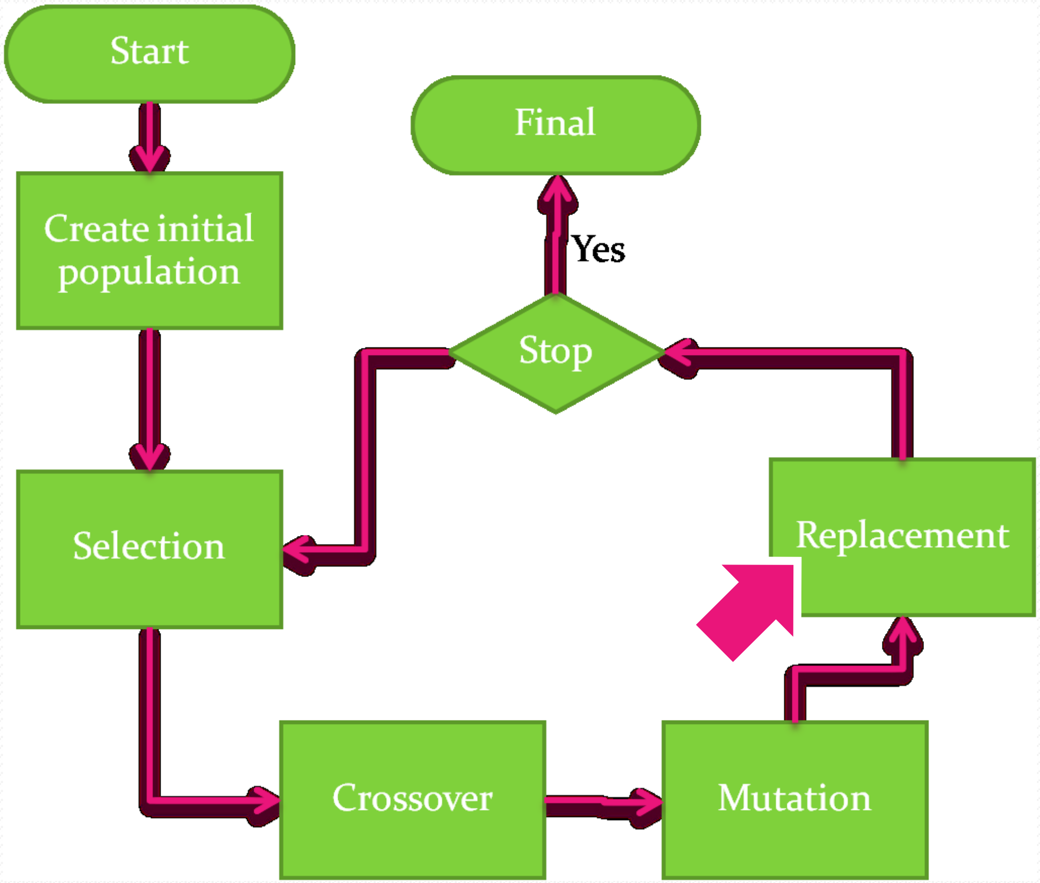
- Select P6 by random control with mutation probability (pm)

P6: 0011|0010 x=3 y=2

- Random select gene to give new value

P6: 0001|1010 x=? y=?

Replacement



Selection

- Offspring

P6: 0011|0010 x=3 y=2 z=?

P7: 0111|0100 x=5 y=6 z=?

P8: 1011|0010 x=3 y=2 z=?

P9: 1001|0011 x=9 y=3 z=?

P10: 0111|0110 x=5 y=6 z=?

- Parents

P1: 1001|0011 x=9 y=3

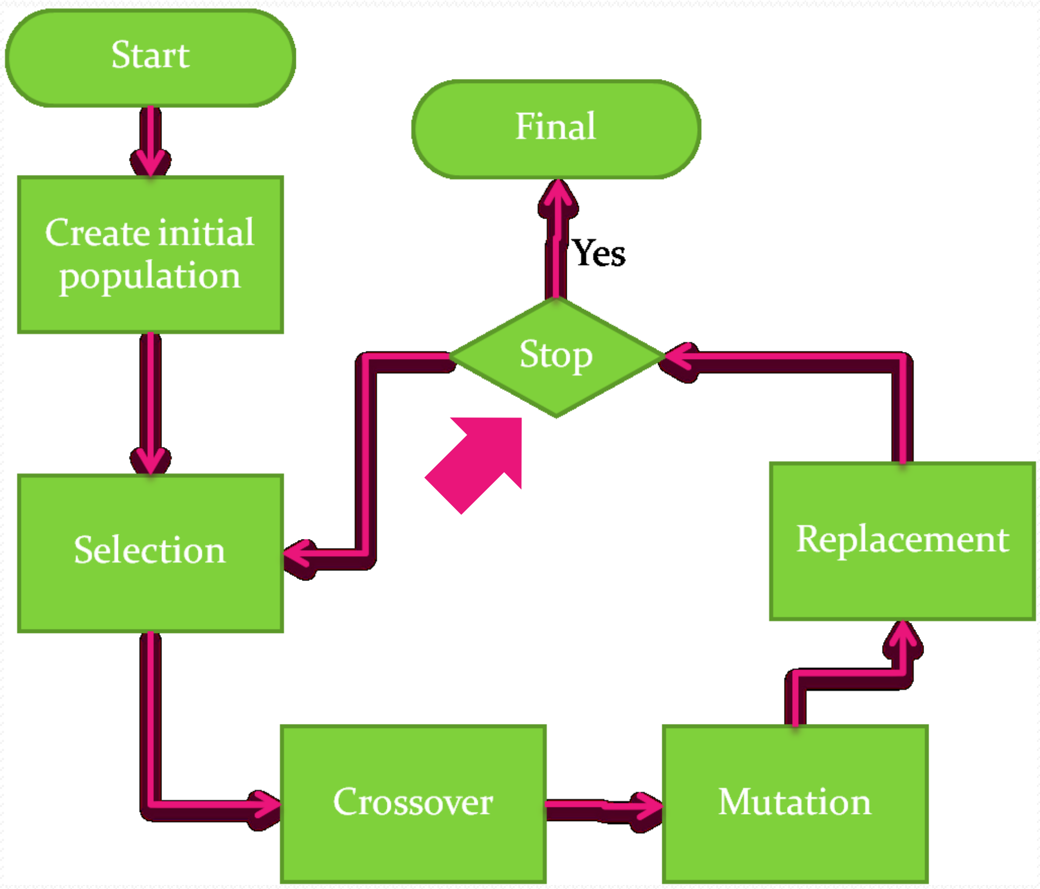
P2: 0101|0110 x=5 y=6

P3: 0011|0010 x=3 y=2

P4: 1001|0011 x=9 y=3

P5: 1101|0010 x=13 y=2

Termination?



Example GA

```
//Example Simple-GA
//Programmer: Dr.Supakit Nootyaskool
//Email: supakitnootyaskool@gmail.com
//Work: IT-KMITL
//Date: 2012 July 02
//Code status: development
//(c)2012


#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

typedef struct population
{
    double a,b,c;
    double fit;
};

void mathfunc(population &p)
{
    double fitness;
    fitness = p.a*p.a + p.b*p.b + p.c;
    p.fit = fitness;
}

```

```
void showdata(char name[10],population p)
{
    printf("%s [%f,%f,%f] fit= %f\n",name,p.a,p.b,p.c,p.fit);
}

double random()
{
    return (double) rand()/1000;
}

double random_0to1()
{
    return (double) rand()/INT_MAX;
}

int main(void)
{
    //parameters
    const int maxpop = 10;
    double pc = 0.1;
    double pm = 0.8;
    int maxgen = 3000;
    int gen = 1;

```

```
    //create initial population
    population oldp[maxpop];
    population newp[maxpop];
    for(int i=0;i<maxpop;i++)
    {
        oldp[i].a = random();
        oldp[i].b = random();
        oldp[i].c = random();
        mathfunc(oldp[i]);
        showdata("oldpop",oldp[i]);
    }

    //generation cycle
    while(1)
    {
        //Selection uses uniform random
        printf("\n--SELECTION--\n");
        for(int i=0;i<maxpop;i++)
        {
            newp[i] = oldp[rand()%maxpop];
            showdata("newpop",newp[i]);
        }

```

```
        //crossover
        printf("\n--CROSSOVER--\n");
        for(int i=0;i<maxpop;i++)
        {
            int z1,z2;
            z1 = rand()%maxpop;
            z2 = rand()%maxpop;

            if(random_0to1()<=pc)
            {
                double t;
                t = newp[z1].a;
                newp[z1].a = newp[z2].a;
                newp[z2].a = t;
            }

            if(random_0to1()<=pc)
            {
                double t;
                t = newp[z1].b;
                newp[z1].b = newp[z2].b;
                newp[z2].b = t;
            }
```

```
        if(random_0to1()<=pc)
        {
            double t;
            t = newp[z1].c;
            newp[z1].c = newp[z2].c;
            newp[z2].c = t;
        }
        mathfunc(newp[i]);
        showdata("newpop",newp[i]);
    }

    //Mutation
    printf("\n--MUTATION--\n");
    for(int i=0;i<maxpop;i++)
    {
        if(random_0to1()<=pm)
            newp[i].a = random();
        if(random_0to1()<=pm)
            newp[i].b = random();
        if(random_0to1()<=pm)
            newp[i].c = random();
        mathfunc(newp[i]);
        showdata("newpop",newp[i]);
    }

```

```
            //Replacement
            population tmp[maxpop*2];
            for(int j=0;j<maxpop;j++)
                tmp[j] = oldp[j];
            for(int j=0;j<maxpop;j++)
                tmp[j+maxpop] = newp[j];


            for(int j=0;j<maxpop*2;j++)
            for(int i=0;i<maxpop*2-1;i++)
            {
                if(tmp[i].fit > tmp[i+1].fit)
                {
                    population t;
                    t = tmp[i];
                    tmp[i] = tmp[i+1];
                    tmp[i+1] = t;
                }
            }
            printf("\n--REPLACEMENT--\n");
            for(int i=0;i<maxpop-1;i++)
            {
                showdata("tmp",tmp[i]);
                oldp[i] = tmp[i];       //copy
            }

            //pause
            printf("\n--PAUSE Gen[%d]--",gen++);
            getch();
        }
        getch();
        return 0;
    }
```

d:\Temp\FirstGA\Debug\FirstGA.exe

```
oldpop [0.041000,18.467000,6.334000] fit= 347.365770
oldpop [26.500000,19.169000,15.724000] fit= 1085.424561
oldpop [11.478000,29.358000,26.962000] fit= 1020.598648
oldpop [24.464000,5.705000,28.145000] fit= 659.179321
oldpop [23.281000,16.827000,9.961000] fit= 835.113890
oldpop [0.491000,2.995000,11.942000] fit= 21.153106
oldpop [4.827000,5.436000,32.391000] fit= 85.241025
oldpop [14.604000,3.902000,0.153000] fit= 228.655420
oldpop [0.292000,12.382000,17.421000] fit= 170.820188
oldpop [18.716000,19.718000,19.895000] fit= 758.983180

--SELECTION--
newpop [14.604000,3.902000,0.153000] fit= 228.655420
newpop [4.827000,5.436000,32.391000] fit= 85.241025
newpop [26.500000,19.169000,15.724000] fit= 1085.424561
newpop [0.292000,12.382000,17.421000] fit= 170.820188
newpop [18.716000,19.718000,19.895000] fit= 758.983180
newpop [11.478000,29.358000,26.962000] fit= 1020.598648
newpop [14.604000,3.902000,0.153000] fit= 228.655420
newpop [18.716000,19.718000,19.895000] fit= 758.983180
newpop [0.491000,2.995000,11.942000] fit= 21.153106
newpop [23.281000,16.827000,9.961000] fit= 835.113890

--CROSSOVER--
newpop [14.604000,3.902000,0.153000] fit= 228.655420
newpop [18.716000,19.718000,19.895000] fit= 758.983180
newpop [26.500000,19.169000,15.724000] fit= 1085.424561
newpop [23.281000,16.827000,9.961000] fit= 835.113890
newpop [18.716000,19.718000,19.895000] fit= 758.983180
newpop [14.604000,3.902000,0.153000] fit= 228.655420
newpop [14.604000,3.902000,0.153000] fit= 228.655420
newpop [0.292000,12.382000,17.421000] fit= 170.820188
newpop [0.491000,2.995000,11.942000] fit= 21.153106
newpop [23.281000,16.827000,9.961000] fit= 835.113890

--MUTATION--
newpop [32.439000,11.323000,21.538000] fit= 1202.037050
newpop [2.082000,16.541000,31.115000] fit= 309.054405
newpop [29.658000,9.930000,2.306000] fit= 980.507864
newpop [22.386000,28.745000,19.072000] fit= 1346.480021
newpop [5.829000,15.573000,16.512000] fit= 293.007570
newpop [13.290000,18.636000,24.767000] fit= 548.691596
newpop [15.574000,12.052000,1.150000] fit= 388.950180
newpop [21.724000,3.430000,30.191000] fit= 513.888076
newpop [11.337000,12.287000,10.383000] fit= 289.880938
newpop [8.909000,9.758000,18.588000] fit= 193.176845

--REPLACEMENT--
tmp [0.491000,2.995000,11.942000] fit= 21.153106
tmp [4.827000,5.436000,32.391000] fit= 85.241025
tmp [0.292000,12.382000,17.421000] fit= 170.820188
tmp [8.909000,9.758000,18.588000] fit= 193.176845
tmp [14.604000,3.902000,0.153000] fit= 228.655420
tmp [11.337000,12.287000,10.383000] fit= 289.880938
tmp [5.829000,15.573000,16.512000] fit= 293.007570
tmp [2.082000,16.541000,31.115000] fit= 309.054405
tmp [0.041000,18.467000,6.334000] fit= 347.365770

--PAUSE Gen[1]--
--SELECTION--
```

d:\Temp\FirstGA\Debug\FirstGA.exe

```
--REPLACEMENT--
tmp [1.054000,0.213000,0.475000] fit= 1.631285
tmp [1.369000,0.641000,4.820000] fit= 7.105042
tmp [1.888000,1.887000,0.648000] fit= 7.773313
tmp [1.453000,2.141000,1.807000] fit= 8.502090
tmp [1.831000,1.668000,5.099000] fit= 11.233785
tmp [0.821000,2.175000,6.360000] fit= 11.764666
tmp [0.632000,1.950000,7.763000] fit= 11.964924
tmp [0.533000,2.767000,4.415000] fit= 12.355378
tmp [2.890000,1.869000,0.948000] fit= 12.793261

--PAUSE Gen[599]--
--SELECTION--
newpop [1.054000,0.213000,0.475000] fit= 1.631285
newpop [1.369000,0.641000,4.820000] fit= 7.105042
newpop [1.831000,1.668000,5.099000] fit= 11.233785
newpop [1.369000,0.641000,4.820000] fit= 7.105042
newpop [1.888000,1.887000,0.648000] fit= 7.773313
newpop [0.533000,2.767000,4.415000] fit= 12.355378
newpop [1.888000,1.887000,0.648000] fit= 7.773313
newpop [0.533000,2.767000,4.415000] fit= 12.355378
newpop [1.453000,2.141000,1.807000] fit= 8.502090
newpop [2.890000,1.869000,0.948000] fit= 12.793261

--CROSSOVER--
newpop [1.054000,0.213000,0.475000] fit= 1.631285
newpop [1.369000,0.641000,4.820000] fit= 7.105042
newpop [1.888000,1.887000,0.648000] fit= 7.773313
newpop [1.369000,0.641000,4.820000] fit= 7.105042
newpop [1.888000,1.887000,0.648000] fit= 7.773313
newpop [1.054000,0.213000,0.475000] fit= 1.631285
newpop [1.831000,1.668000,5.099000] fit= 11.233785
newpop [0.533000,2.767000,4.415000] fit= 12.355378
newpop [0.533000,2.767000,4.415000] fit= 12.355378
newpop [1.369000,0.641000,4.820000] fit= 7.105042

--MUTATION--
newpop [21.360000,25.005000,16.756000] fit= 1098.255625
newpop [6.344000,28.462000,25.972000] fit= 876.303780
newpop [2.175000,32.700000,7.145000] fit= 1081.165625
newpop [12.622000,16.545000,19.500000] fit= 452.551909
newpop [25.318000,31.141000,10.898000] fit= 1621.661005
newpop [30.084000,12.271000,2.439000] fit= 1058.063497
newpop [29.387000,23.826000,18.383000] fit= 1449.657045
newpop [27.941000,12.300000,16.524000] fit= 948.513481
newpop [20.183000,11.349000,29.143000] fit= 565.296290
newpop [15.495000,24.412000,5.393000] fit= 841.433769

--REPLACEMENT--
tmp [1.054000,0.213000,0.475000] fit= 1.631285
tmp [1.369000,0.641000,4.820000] fit= 7.105042
tmp [1.888000,1.887000,0.648000] fit= 7.773313
tmp [1.453000,2.141000,1.807000] fit= 8.502090
tmp [1.831000,1.668000,5.099000] fit= 11.233785
tmp [0.821000,2.175000,6.360000] fit= 11.764666
tmp [0.632000,1.950000,7.763000] fit= 11.964924
tmp [0.533000,2.767000,4.415000] fit= 12.355378
tmp [2.890000,1.869000,0.948000] fit= 12.793261

--PAUSE Gen[600]--
```