

# Encoding Techniques in Genetic Algorithms

**Debasis Samanta**

Indian Institute of Technology Kharagpur

*dsamanta@iitkgp.ac.in*

01.03.2016

# GA Operators

Following are the GA operators in Genetic Algorithms.

- 1 Encoding
- 2 Convergence test
- 3 Mating pool
- 4 Fitness Evaluation
- 5 Crossover
- 6 Mutation
- 7 Inversion

# Encoding Operation

- 1 **Encoding**
- 2 Convergence test
- 3 Mating pool
- 4 Fitness Evaluation
- 5 Crossover
- 6 Mutation
- 7 Inversion

## • Different GAs

- Simple Genetic Algorithm (SGA)
- Steady State Genetic Algorithm (SSGA)
- Messy Genetic Algorithm (MGA)

## • Encoding Schemes

- Binary encoding
- Real value encoding
- Order encoding
- Tree encoding

# Different Encoding Schemes

Often, GAs are specified according to the encoding scheme it follows.

For example:

- Encoding Scheme
- Binary encoding → Binary Coded GA or simply **Binary GA**
- Real value encoding → Real Coded GA or simply **Real GA**
- Order encoding → **Order GA** (also called as **Permuted GA**)
- Tree encoding

# Encoding Schemes in GA

Genetic Algorithm uses metaphor consisting of two distinct elements :

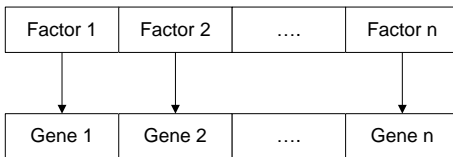
- 1 Individual
- 2 Population

An individual is a single solution while a population is a set of individuals at an instant of searching process.

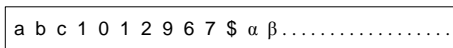
# Individual Representation :Phenotype and Genotype

- An individual is defined by a chromosome. A chromosome stores genetic information (called phenotype) for an individual.
- Here, a chromosome is expressed in terms of factors defining a problem.

Genotype



Phenotype



# Individual Representation :Phenotype and Genotype

## Note :

- A gene is the GA's representation of a single factor (i.e. a design parameter), which has a domain of values (continuous, discontinuous, discrete etc.) symbol, numbering etc.
- In GA, there is a mapping from genotype to phenotype. This eventually decides the performance (namely speed and accuracy) of the problem solving.



**There are many ways of encoding:**

- ➊ **Binary encoding:** Representing a gene in terms of bits (0s and 1s).
- ➋ **Real value encoding:** Representing a gene in terms of values or symbols or string.
- ➌ **Permutation (or Order) encoding:** Representing a sequence of elements)
- ➍ **Tree encoding:** Representing in the form of a tree of objects.

# Binary Encoding

In this encoding scheme, a gene or chromosome is represented by a string (fixed or variable length) of binary bits (0's and 1's)

A :

0 1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 1 0

Individual 1

B :

0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0

Individual 2

# Example: 0-1 Knapsack problem

- There are  $n$  items, each item has its own cost ( $c_i$ ) and weight ( $w_i$ ).
- There is a knapsack of total capacity  $w$ .
- The problem is to take as much items as possible but not exceeding the capacity of the knapsack.

This is an optimization problem and can be better described as follows.

**Maximize :**

$$\sum_i c_i * w_i * x_i$$

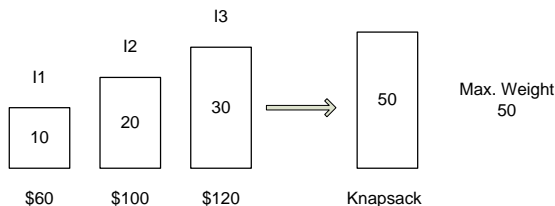
**Subject to**

$$\sum x_i * w_i \leq W$$

where  $x_i \in [0 \dots 1]$

# Example: 0-1 Knapsack problem

Consider the following, an instance of the 0-1 Knapsack problem.



Brute force approach to solve the above can be stated as follows:

- Select at least one item

[10], [20], [30], [10, 20], [10, 30], [20, 30], [10, 20, 30]

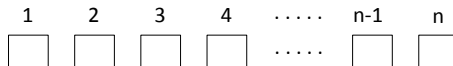
- So, for n-items, there are  $2^n - 1$  trials.
- 0 - means item not included and 1 - means item included

[100], [010], [011], [110], [101], [011], [111]

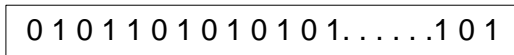
# Example: 0-1 Knapsack problem

The encoding for the 0-1 Knapsack, problem, in general, for  $n$  items set would look as follows.

Genotype :



Phenotype :



A binary string of n-bits

# Few more examples

- **Example 1 :**

Minimize :

$$f(x) = \frac{x^2}{2} + \frac{125}{x}$$

where  $0 \leq x \leq 15$  and  $x$  is any discrete integer value.

Genotype :

**x**

Phenotype :

**0 1 1 0 1**

A binary string of 5-bits

# Few more examples

- **Example 2 :**

Maximize :

$$f(x, y) = x^3 - x^2y + xy^2 + y^3$$

Subject to :

$$x + y \leq 10$$

and

$$\begin{aligned} 1 &\leq x \leq 10 \\ -10 &\leq y \leq 10 \end{aligned}$$

Genotype :

<b>x</b>	<b>y</b>
----------	----------

Phenotype :

<b>0 1 1 0 1</b>	<b>1 1 0 0 1</b>
------------------	------------------

Two binary string of 5-bits each

# Pros and cons of Binary encoding scheme

- **Limitations:**

- ① Needs an effort to convert into binary from
- ② Accuracy depends on the binary representation

- **Advantages:**

- ① Since operations with binary representation is faster, it provides faster implementations of all GA operators and hence the execution of GAs.
- ② Any optimization problem has its binary-coded GA implementation



# Real value encoding

- The real-coded GA is most suitable for optimization in a continuous search space.
- Uses the direct representations of the design parameters.
- Thus, avoids any intermediate encoding and decoding steps.

Genotype :

<b>x</b>	<b>y</b>
----------	----------

Phenotype :

<b>5.28</b>	<b>-475.36</b>
-------------	----------------

Real-value representation

# Real value encoding with binary codes

## Methodology: Step 1 [Deciding the precision]

For any continuous design variable  $x$  such that  $X_L \leq x \leq X_U$ , and if  $\varepsilon$  is the precision required, then string length  $n$  should be equal to

$$n = \log_2 \left( \frac{X_U - X_L}{\varepsilon} \right)$$

where  $X_L \leq x \leq X_U$

Equivalently,

$$\varepsilon = \left( \frac{X_U - X_L}{2^n} \right)$$

In general,  $\varepsilon = [0 \cdots 1]$ . It is also called, *Obtainable accuracy*

**Note:** If  $\varepsilon = 0.5$ , then  $4.05$  or  $4.49 \equiv 4$  and  $4.50$  or  $4.99 \equiv 4.5$  and so on.

# Real value encoding: Illustration 1

## 1 Example 1:

$1 \leq x \leq 16$ ,  $n = 6$ . What is the accuracy?

$$\varepsilon = \frac{16-1}{2^6} = \frac{15}{64} = 0.249 \approx 0.25$$

## 2 Example 2:

What is the obtainable accuracy, for the binary representation for a variable  $X$  in the range  $20.1 \leq X \leq 45.6$  with 8-bits?

## 3 Example 3:

In the above case, what is the binary representation of  $X = 34.35$ ?

# Real value encoding with binary codes

## Methodology: Step 2[Obtaining the binary representation]

Once, we know the length of binary string for an obtainable accuracy (i.e precision), then we can have the following mapping relation from a real value  $X$  to its binary equivalent decoded value  $X_B$ , which is given by

$$X = X_L + \frac{X_U - X_L}{2^n - 1} \times X_B$$

where  $X_B$  = Decoded value of a binary string,  
 $n$  is the number of bits in the representation,

$X_L \rightarrow 0\ 0\ 0\ 0 \dots 0$  and

$X_U \rightarrow 1\ 1\ 1\ 1 \dots 1$

are the decoded values of the binary representation of the lower and upper values of  $X$ .

# Real value encoding: Illustration 2

## Example:

Suppose,  $X_L = 2$  and  $X_U = 17$  are the two extreme decoded values of a variable  $x$ .

$n = 4$  is the number of binary bits in the representation for  $x$ .

$X_B = 10 (= 1\ 0\ 1\ 0)$  is a decoded value for a given  $x$ .

What is the value of  $x = ?$  and its binary representation??

Here,  $x = 2 + \frac{17-2}{2^4-1} \times 10 = 12$

Binary representation of  $x = 1\ 1\ 0\ 0$

# Order Encoding

Let us have a look into the following instance of the Traveling Salesman Problem (TSP).

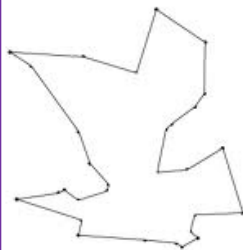
## TSP

- Visit all the cities
- One city once only
- Starting and ending city is the same

All cities are to be visited



A possible tour



**How we can formally define the TSP?**

# Order Encoding for TSP

## Understanding the TSP:

There is a cost of visiting a city from another city and hence the total cost of visiting all the cities but exactly once (except the starting city).

**Objective function:** To find a tour (i.e. a simple cycle covering all the cities) with a minimum cost involved.

## Constraints:

- All cities must be visited.
- There will be only one occurrence of each city (except the starting city).

## Design parameters:

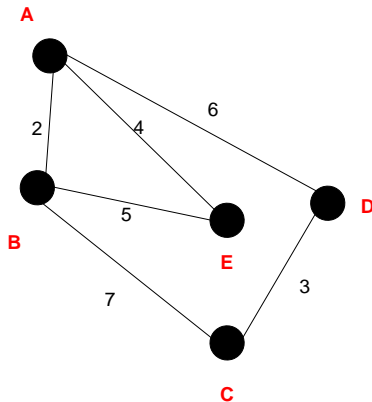
- Euclidean distance may be taken as the measurement of the cost, otherwise, if it is specified explicitly.
- The above stated information are the design variables in this case.

**We are to search for the best path out of  $n!$  possible paths.**

# A small instance of the TSP

d	A	B	C	D	E
A	0	2	$\infty$	6	4
B	2	0	7	$\infty$	5
C	$\infty$	7	0	3	1
D	6	$\infty$	3	0	$\infty$
E	4	5	1	$\infty$	0

d= Distance matrix



Connectivity among cities



# Defining the TSP

## Minimizing

$$\text{cost} = \sum_{i=0}^{n-2} d(c_i, c_{i+1}) + d(c_{n-1}, c_0)$$

## Subject to

$$P = [c_0, c_1, c_2, \dots, c_{n-1}, c_0]$$

where  $c_i \in X$ ;

Here,  $P$  is an ordered collection of cities and  $c_i \neq c_j$  such that  
 $\forall i, j = 0, 1, \dots, n-1$

**Note:**  $P$  represents a possible tour with the starting cities as  $c_0$ .

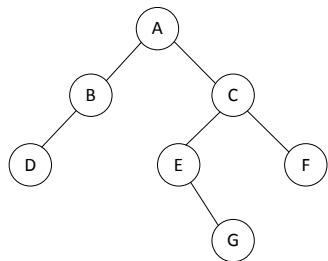
and

$X = x_1, x_2, \dots, x_n$ , set of  $n$  number of cities,

$d(x_i, x_j)$  is the distance between any two cities  $x_i$  and  $x_j$ .

# Tree encoding

In this encoding scheme, a solution is encoded in the form of a binary tree.



A binary tree



D A B E G C F (In-order)

( $T_L$  R  $T_R$ )

A B D C E G F (Pre-order)

(R  $T_L$   $T_R$ )

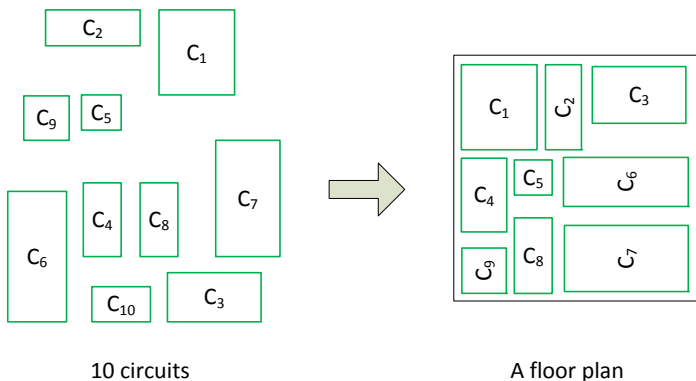
D B G E E C A (Post-order)

( $T_L$   $T_R$  R)

Three compact representation

# Floor Planning : An example of tree encoding

Floor planning is a standard problem in VLSI design. Here, given  $n$  circuits of different area requirements, we are to arrange them into a floor of chip layout, so that all circuits are placed in a minimum layout possible.



# Formulation of floor planning problem

A possible formulation of Floor planning problem of VLSI circuit is as follows.

**Given :**

- 1 A set of  $n$  rectangular blocks  $B = b_1, b_2, \dots, b_i, \dots, b_n$
- 2 For each  $b_i \in B$ , we have the following specification:
  - the width  $w_i$  and height  $h_i$  (which are constant for rigid blocks and variable for flexible blocks)
  - $\rho_i$ , the desirable aspect ratio about where  $\frac{1}{\rho_i} \leq \frac{h_i}{w_i} \leq \rho_i$ , where  $\rho_i = 1$ , if the block  $b_i$  is rigid.
  - $a_i = w_i \times h_i$ , the area of each block  $b_i$ .

# Formulation of floor planning problem

- 3 A set of nets  $N = \{n_1, n_2, \dots, n_k\}$  describing the connectivity information.

$$\text{Wire} = f_1(B, N)$$

- 4 Desirable floor plan aspect ratio  $\rho$  such that  $\frac{1}{\rho} \leq \frac{H}{W} \leq \rho$ , where  $H$  and  $W$  are the height and width of the floor plan, respectively.

$$\text{Area} = f_2(B, N, \rho)$$

- 5 Timing information.

$$\text{Delay} = f_3(B, N, \rho)$$

# Formulation of Floor planning problem

A legal floor plan is a floor plan that satisfies the following constraints.

## Constraints :

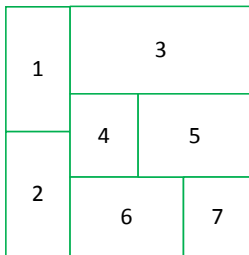
- ③ Each block  $b_i$  is assigned to a location say  $(x_i, y_i)$ .
- ④ No two blocks overlap
- ⑤ For each flexible block say  $b_i$ ,  $a_i = w_i \times h_i$  and should meet aspect ratio constraint.

## Objectives :

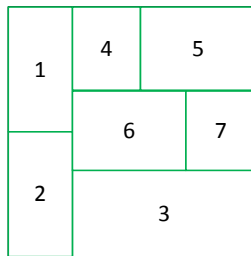
We are to find a floor plan, which would

- ① Minimize floor plan area.
- ② Minimize wire length.
- ③ Minimize circuit delay.

# Tree encoding for Floor planning problem



Floor Plan I



Floor Plan II

- 1 How many floor plans are possible?
- 2 Can we find a binary tree representation of a floor plan??

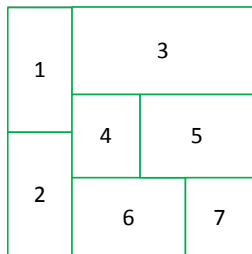
# Binary tree representation for floor planning

A floor plan can be modeled by a binary tree with  $n$  leaves and  $n - 1$  nodes where

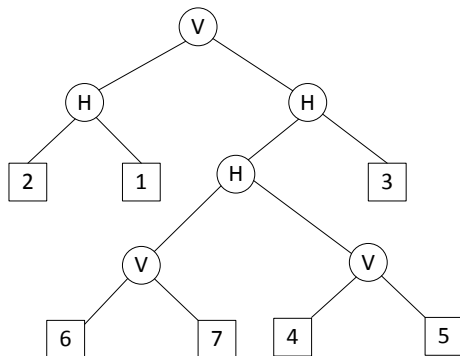
- each node represents a vertical cut-line or horizontal cut-line, and Letter V and H refer to vertical and horizontal cut-operators.
- each leaf node represents a rectangle blocks.



# Example : Floor plane I



Floor Plan I



Binary tree representation of the floor plan I

# Example : Floor plane I

## Note 1:

The operators H and V expressed in polish notation carry the following meanings:

$ijH \rightarrow$  Block  $b_j$  is on top of the block  $b_i$ .

$ijV \rightarrow$  Block  $b_i$  is on the left of block  $b_j$ .

**Note 2: A tree can be represented in a compact form using Polish notation**

**Note 3: Polish notation**

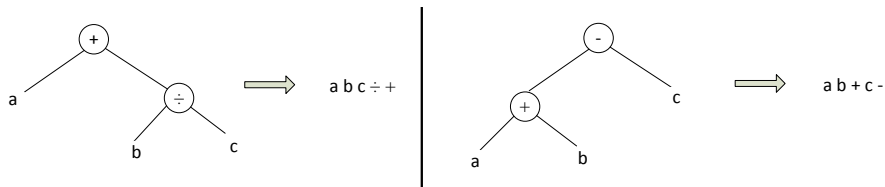
$$a + b \div c = a + (b \div c) = abc \div +$$

$$a + b - c = ab + c -$$

# Example : Floor plane I

## Note 4:

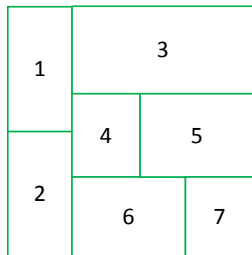
Post order traversal of a binary tree is equivalent to polish notation



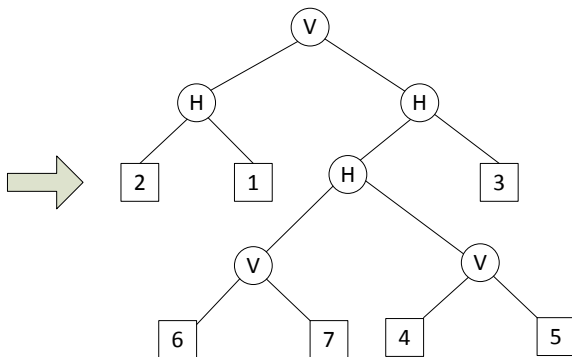
## Note 5:

There is only one way to performing a post order traversal of a binary tree.

# Example : Floor Plane I (with Polish notation)



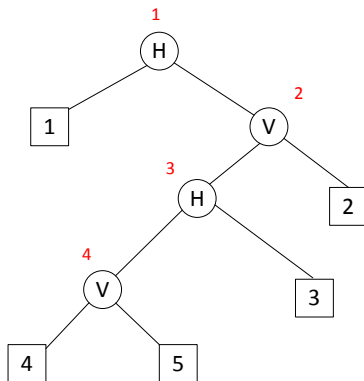
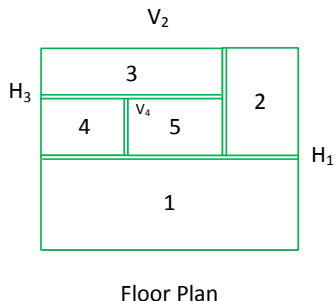
Floor Plan I



Binary tree representation of the floor plan I

Polish notation : 2 1 H 6 7 V 4 5 V H 3 H V

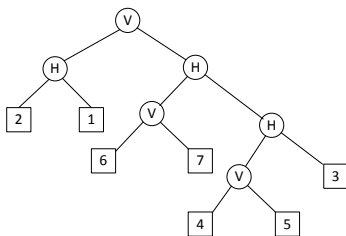
# Example : H and V operators



Binary tree

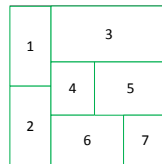
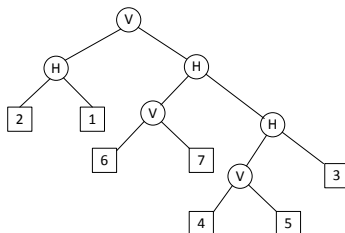
Polish notation : 4 5 V 3 H 2 V 1 H

# Example :Floor plan II



2 1 H 6 7 V 4 5 V 3 H H V

# Example :Floor plan II



2 1 H 6 7 V 4 5 V 3 H H V

## Problem :

How many number of solutions possible with  $n$  blocks in a floor planning problem?



## Problem :

How many number of solutions with  $n$  blocks in a floor planning problem?

$$N = \frac{1}{n+1} \binom{2n}{n}$$

**Any question??**