MACHINE INTELLIGENCE

UNIT - 4

Genetic Algorithms

feedback/corrections: vibha@pesu.pes.edu

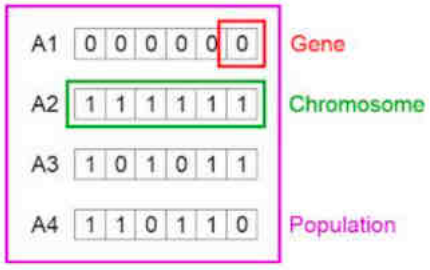
VIBHA MASTI

Genetic Algorithms

- Metaheuristic search-based optimisation technique
- optimal and near-optimal solutions
- Hypothesis fitness: numerical measure for a problem; to be optimised
  - eg: accuracy of hyp over train data

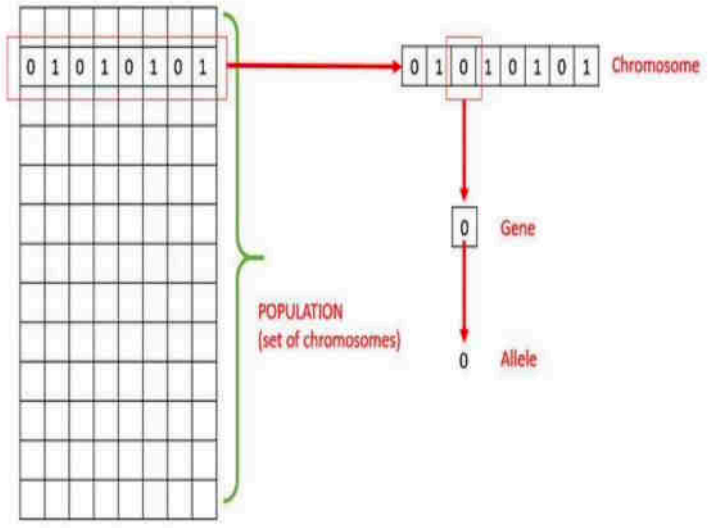
Basic Terminologies

1) Population
   - Subset of all possible encoded solutions
2) Chromosome
   - One such solution to problem
3) Gene
   - One element position of a chromosome



4) Allele

- Value a gene takes for a particular chromosome



5) Genetic operators

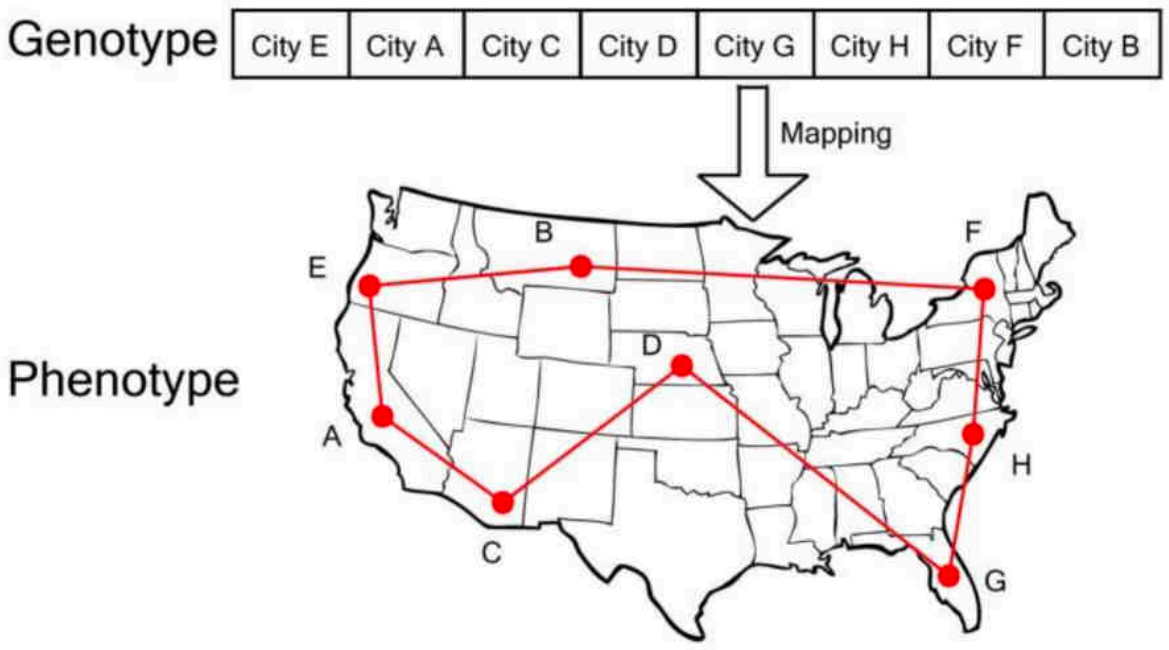
- Alter genetic composition of offspring
- crossover, mutation, selection etc

6) Genotype

- Population in the computation space
- In bio, genotype is possible set of alleles (bb, bB, BB, Bb)

7) Phenotype

- Population in the real-world solution space
- In bio, phenotype is the result of a genotype (blue eyes, brown eyes)



Genotype Representation

- Parameters of the solution (genes) are concatenated to form a string (chromosome)
- Encoding: data structure for representing candidate solutions
- Genotype representation

1. Binary string
2. Value encoding - real / int
3. Permutation representation
4. Random key representation
5. Genes in arrays, trees, lists

- Improper encoding: poor performance

1. Binary encoding

- 0/1 knapsack problem: binary string
- If-then-else representations

0 1 0 0 1 0 1 1

2. Value encoding

- string of any value

Chromosome 1: 3.3245, 5.2132, 1.333, 121.5623 (real numbers)
Chromosome 2: word, asdfasd, 0dfsd, cents, dollar (words)
Chromosome 3: SSNSW, NWWN, NWNNN, SSSS (directions)

- crossover & mutation methods unique to value type

(a) Integer Representation

- eg: North, South, East, West

– North, South, East and West, we can encode them as **{1,2,3,4}**.

1	2	3	4	3	2	4	1	2	1

(b) Real Valued Representation

- best for optimisation in a continuous search space
- no encoding/decoding steps

5.28	-475.36

Real-value representation

3. Permutation Representation

- Path / order representation
- Eg: TSP with 8 cities

3-5-8-1-4-2-6-7

4. Random keys representation

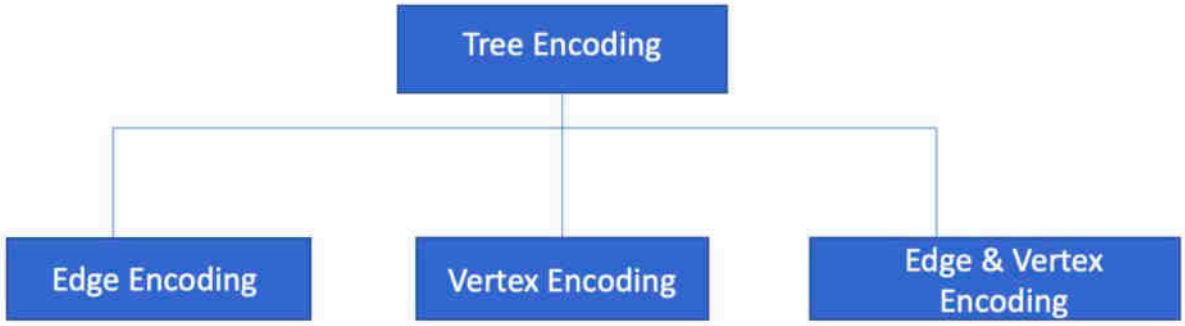
- Solution encoded with random keys
- 8 city TSP can be

[0.45 0.68 0.91 0.11 0.62 0.34 0.74 0.89]

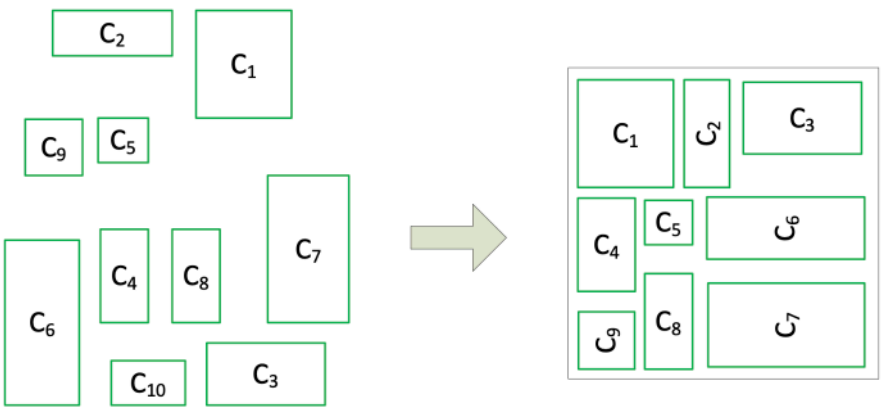
decoded to 3-5-8-1-4-2-6-7

- Eg: machine scheduling
  resource allocation
  vehicle routing
  quadratic assignment

5. Tree representation



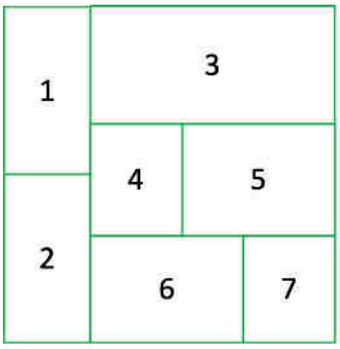
- eg: Prufer's code
- Problem: floor planning - given n circuits of different area requirements, arrange in minimum layout



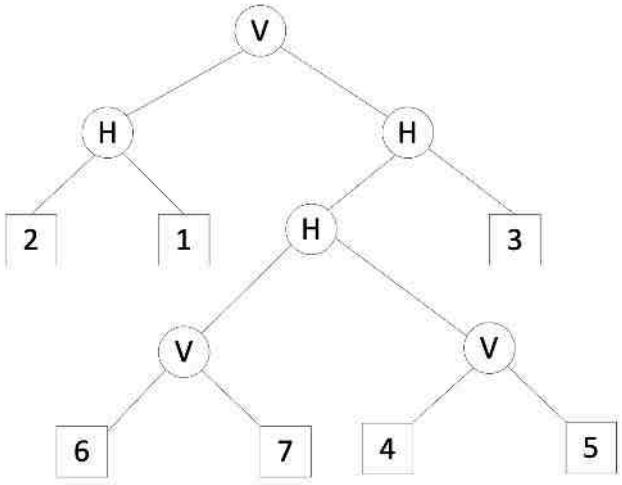
10 circuits

A floor plan

- H operator: horizontal cut; ij H ⇒ j on top of i
- V operator: vertical cut; ij V ⇒ j right of i



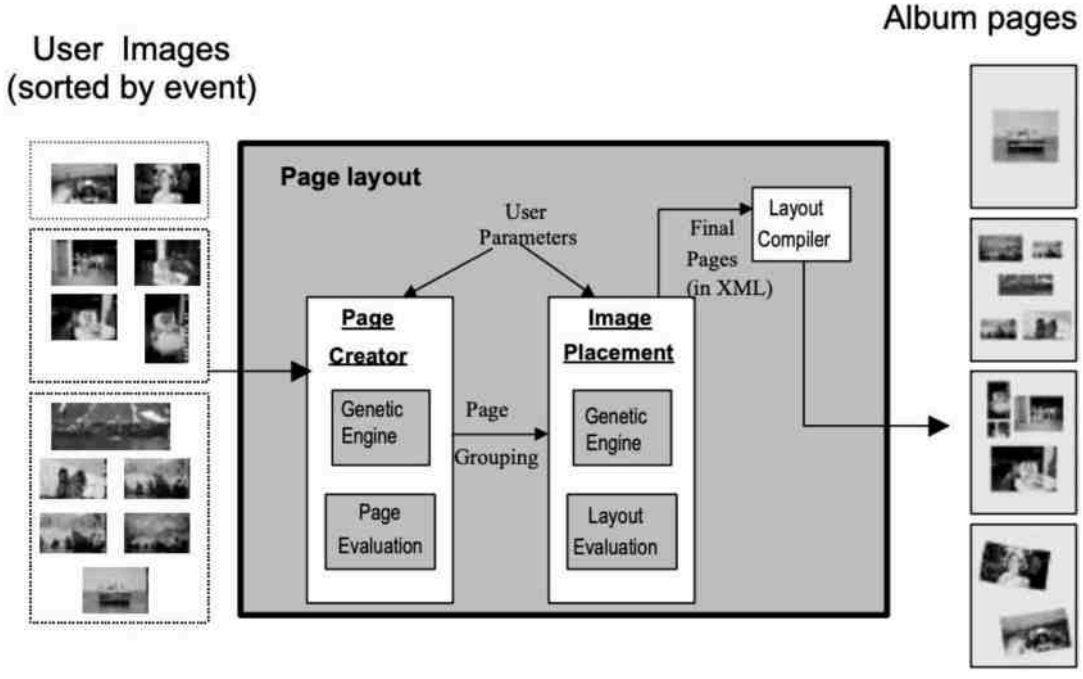
Floor Plan I



Binary tree representation of the floor plan I

Polish notation: 21H 67V 45V H3HV

- Problem: genetic page layout engine



Population Initialisation & Models

1) Random initialisation

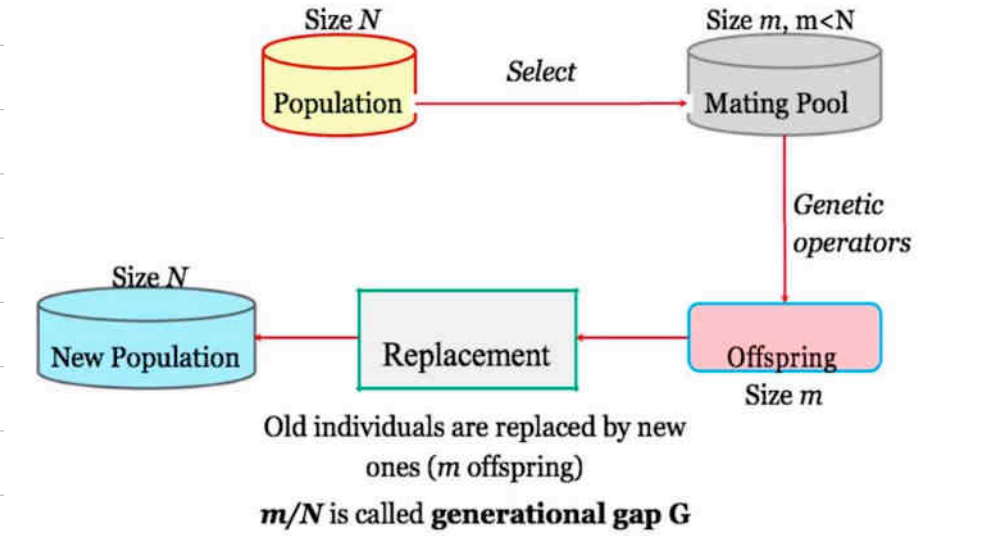
2) Heuristic initialisation

Selection of New population

- Choosing which chromosomes reproduce offspring (pre-selection / parent selection)
- Choose which chromosomes survive to the next generation (post-selection)
- At the end of each generation, new population of candidate sols chosen
- New population selected only from offspring (generational models)
- New population selected from both parents and offspring (steady-state models)

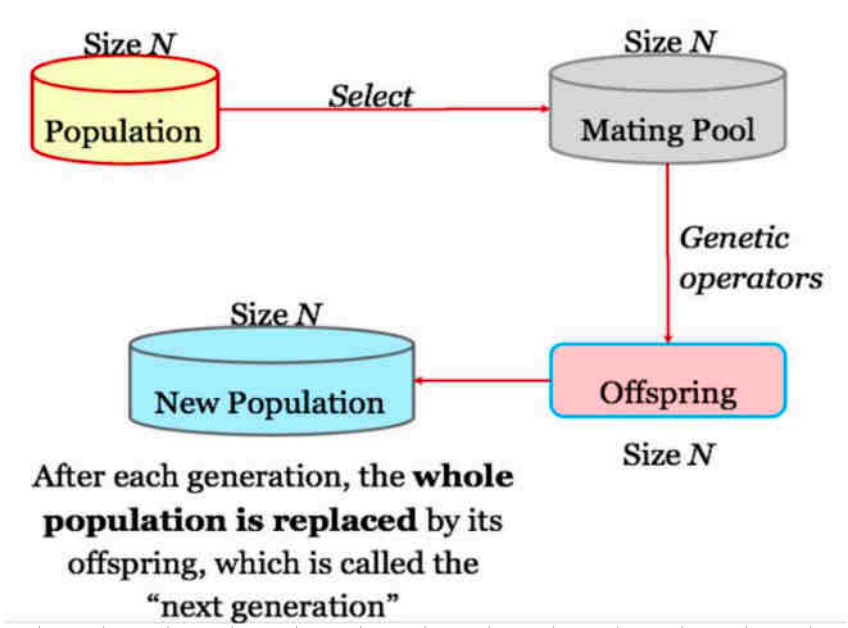
(i) Steady state

- Generate 1 or 2 offsprings in each iter
- Replace 1 or 2 individuals from the pop
- Also called incremental GA
- Not used much



(ii) Generational

- Generate n offsprings (n = pop size)
- Entire population replaced by new one



Generation Gap

- $0 \leq G \leq 1$
- ratio of children of individuals in the prev gen to pop size

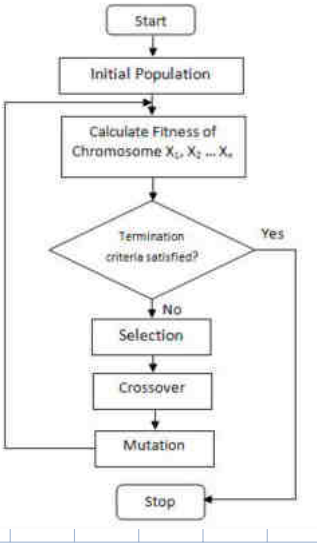
- If $G = 1$, all individuals are children of prev gen
- When G increases, global search capability increases while local search capability decreases

Selective Pressure

- Takeover time
- Probability of best individual being selected
- High selective pressure operators: decrease diversity more rapidly
  - could lead to premature convergence

Basic structure of GAs

- Init pop
- Select parents for crossover
- Apply crossover and mutation operators on parents – generate offspring
- Offsprings replace existing individuals



Basic GA

1. Start: Randomly generate a population of N chromosomes.
2. Fitness: Calculate the fitness of all chromosomes.
3. Create a new population:
    a. Selection: According to the selection method select 2 chromosomes from the population.
    b. Crossover: Perform crossover on the 2 chromosomes selected.
    c. Mutation: Perform mutation on the chromosomes obtained.
4. Replace: Replace the current population with the new population.
5. Test: Test whether the end Termination condition is satisfied. If so, stop. If not, return the best solution in current population and go to Step 2.

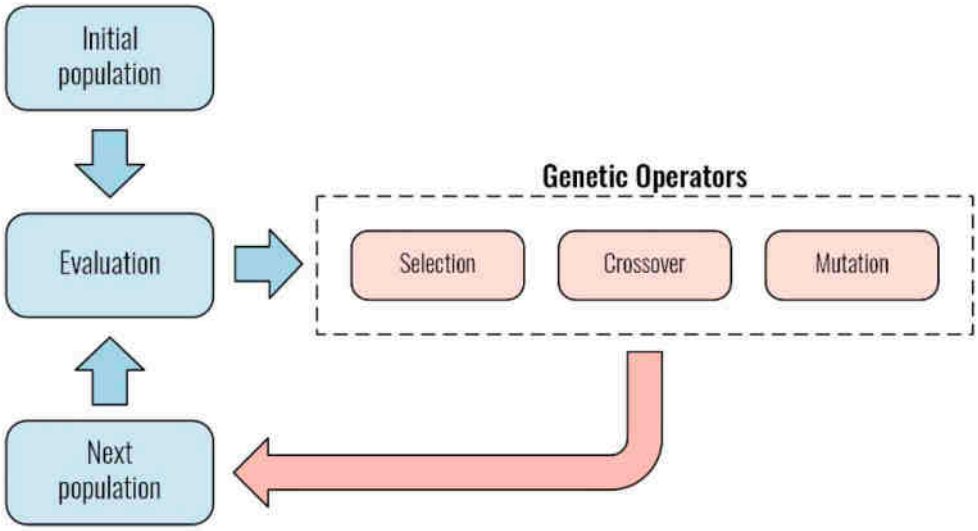
Each iteration of this process is called generation.

Fitness Function

- How good a candidate solution is
- Criteria for ranking potential hypotheses
- Characteristics
  - fast to compute
  - quantitative measure of fit

GENETIC OPERATORS

- Selection, crossover, mutation



(i) Selection Methods

(i) Parent Selection

- select from current gen to take part in reproducing
- eg: fitness proportion
  roulette wheel
  tournament
  rank
  random

(ii) Survivor Selection

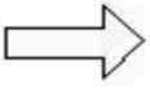
- select from parents + offspring to go into next gen
- eg: age based
  fitness based

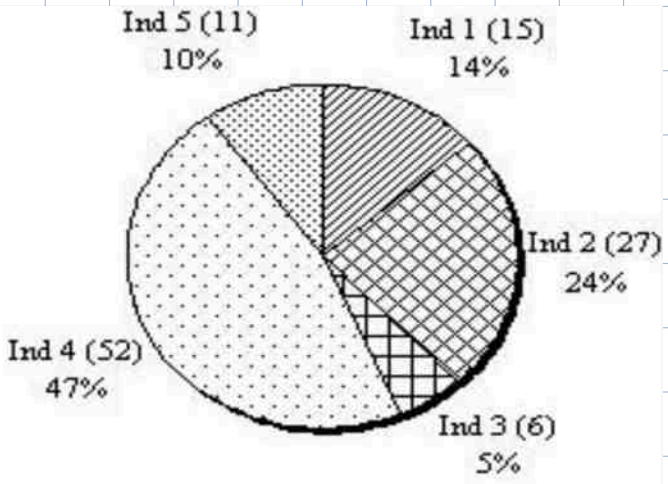
Parent Selection

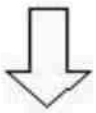
(i) Fitness proportion selection

- Think of pop as circular wheel with n sections
- Area of section $\propto$ fitness of individual

Population	Fitness
Individual 1	15
Individual 2	27
Individual 3	6
Individual 4	52
Individual 5	11

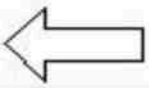






(cumulative limits)

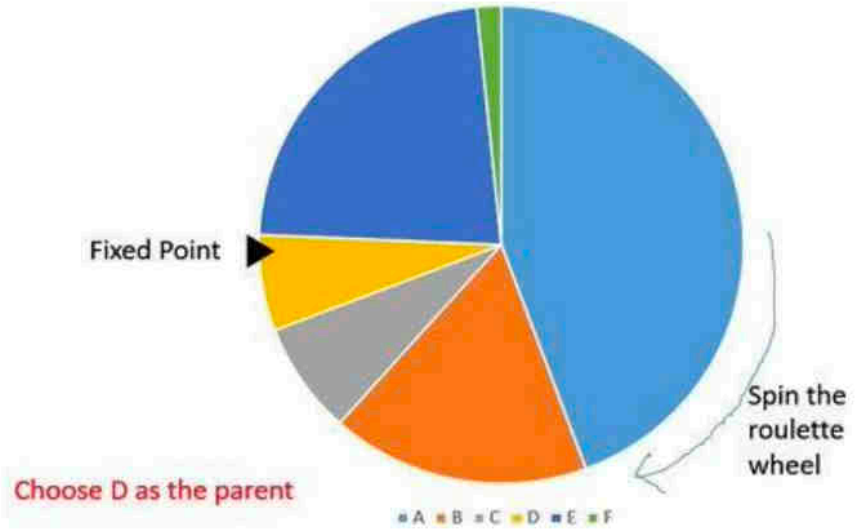
Randomly generated number = 21



Individual 2 is selected

(ii) Roulette Wheel

- Same circular pie as before
- Fixed point on wheel circumference chosen and wheel is spun
- Region that lands in front of fixed point is chosen



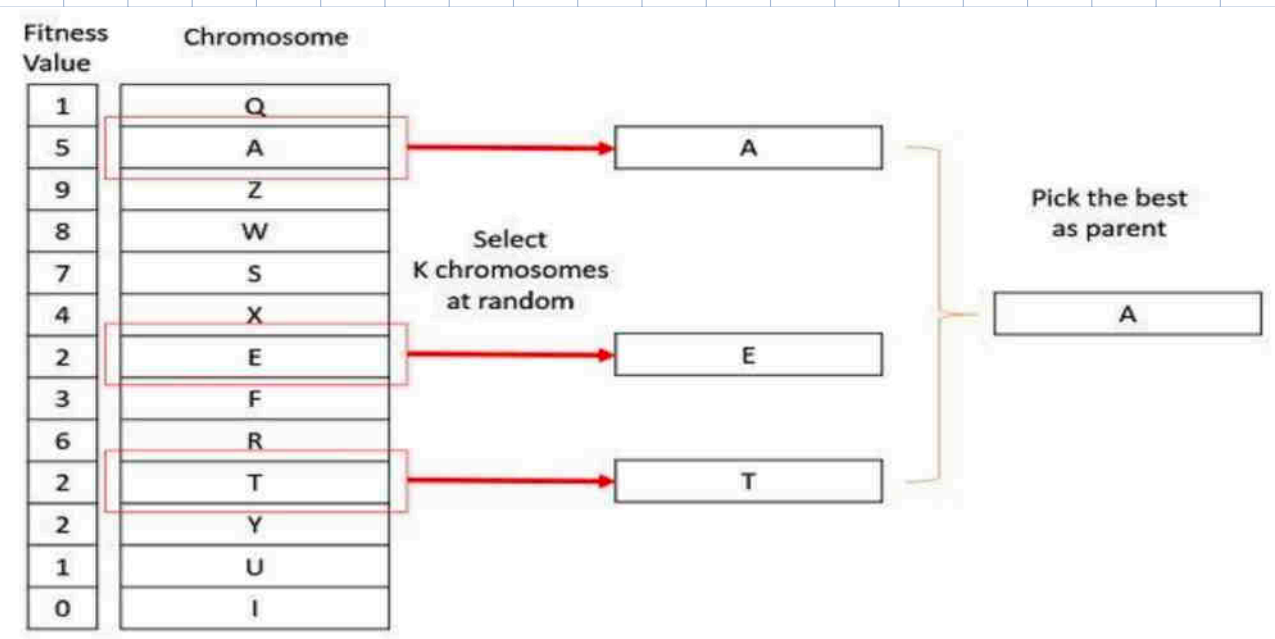
Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

- Implementing

  - select S = sum of fitnesses
  - generate random number b/w 0 and S
  - starting at the top of pop, keep adding fitnesses to partial sum P, till P < S (cumulative fitness)
  - when P > S, it is chosen

(iii) Tournament Selection

- Select k individuals from pop at random
- Select best out of these to become one parent
- Repeat for next parent
- Can work with negative fitness values



(iv) Rank Selection

- Selection based on rank and not fitness values
- Works on negative fitness

(v) Random selection

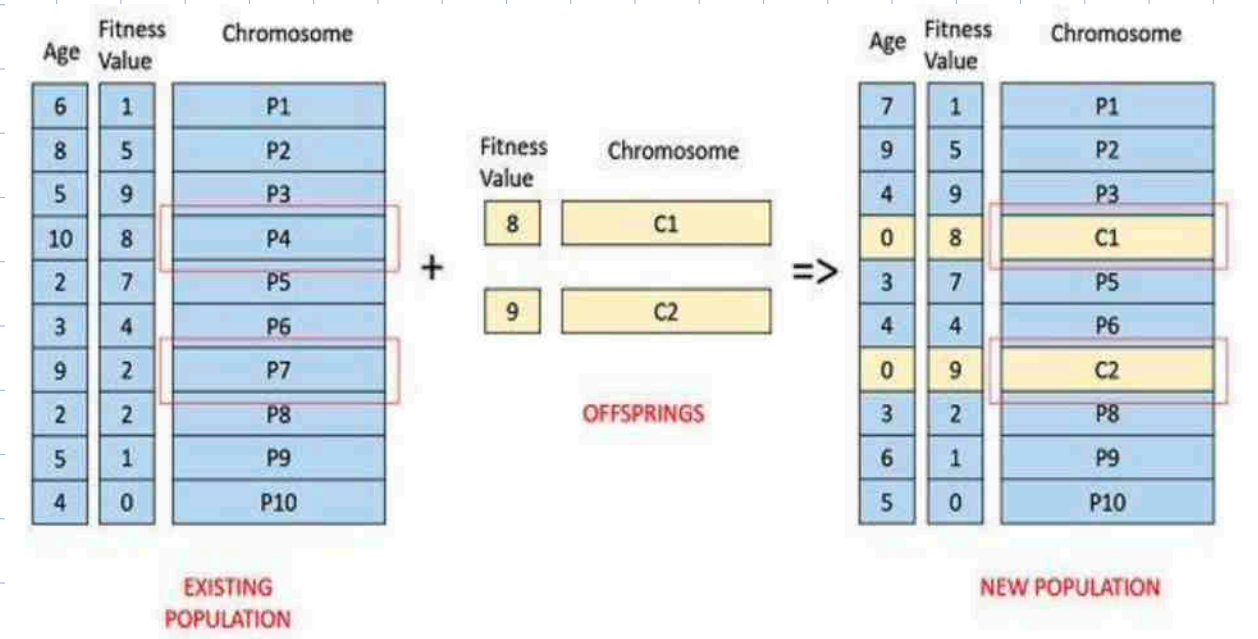
- Randomly selected (no fitness consideration)

Survivor Selection

- Which individuals kicked out
- Elitism: current fittest member of pop always propagated to next gen

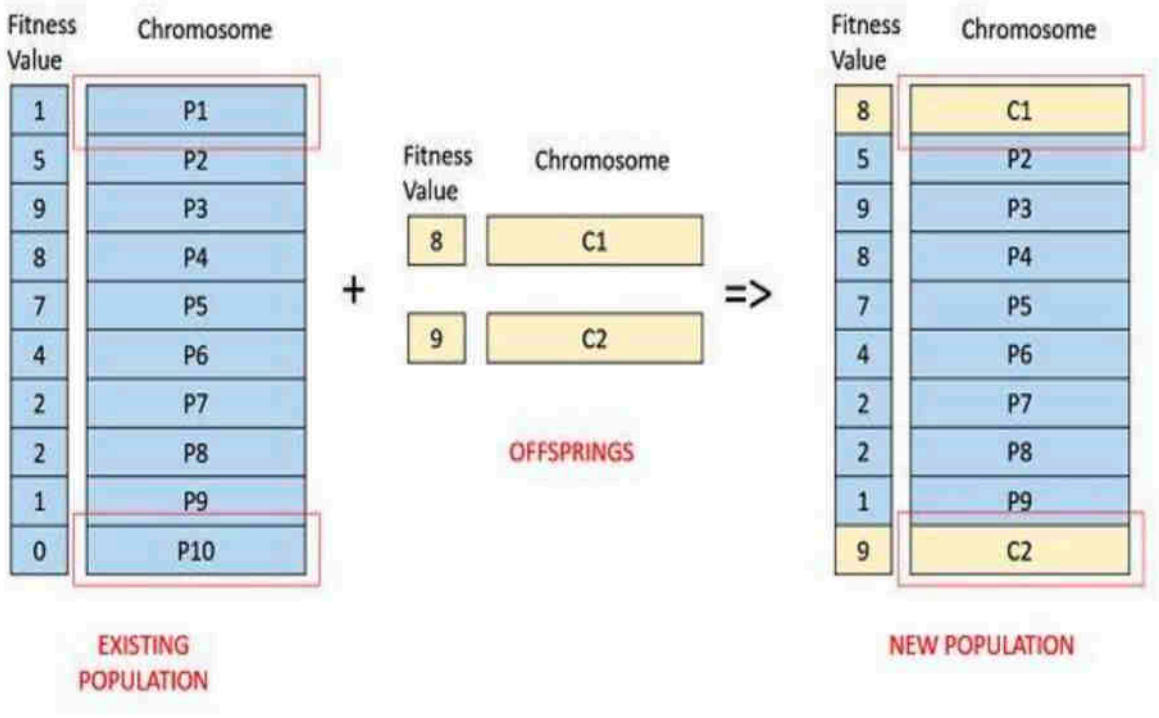
(i) Age-based Selection

- No fitness
- Each individual allowed in pop for a finite generation (when it is allowed to reproduce)
- After that, kicked out



(ii) Fitness-Based Selection

- Children tend to replace least fit
- Select least fit using any policy



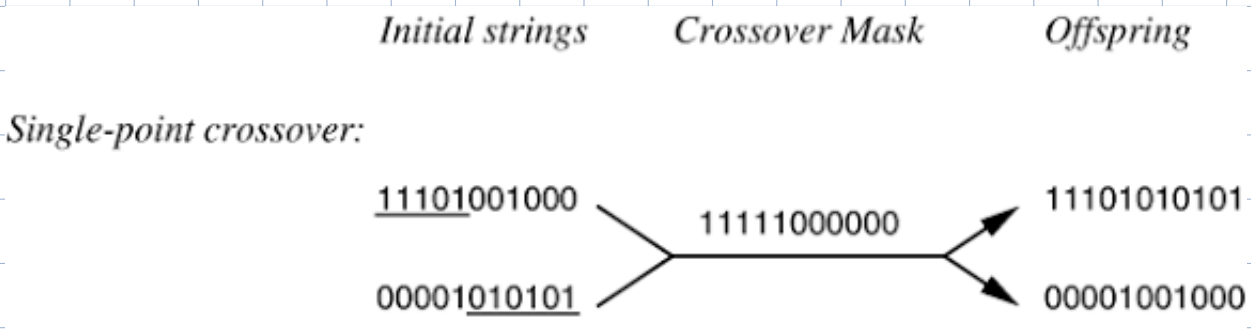
(2) Crossover

(i) One-point
(ii) Two-point
(iii) Multi-point
(iv) Uniform
(v) Arithmatic / Logical
(vi) Whole arithmatic recombination
(vii) Matrix

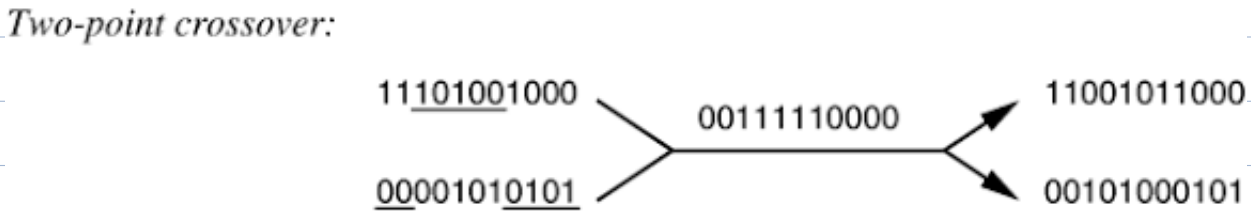
- Produces 2 new offspring from 2 parent strings
- Copy selected bits from parents
- Crossover mask: determines which parent contributes bit in pos i

(i) Single-Point Crossover

- random crossover point chosen
- tails of parents swapped

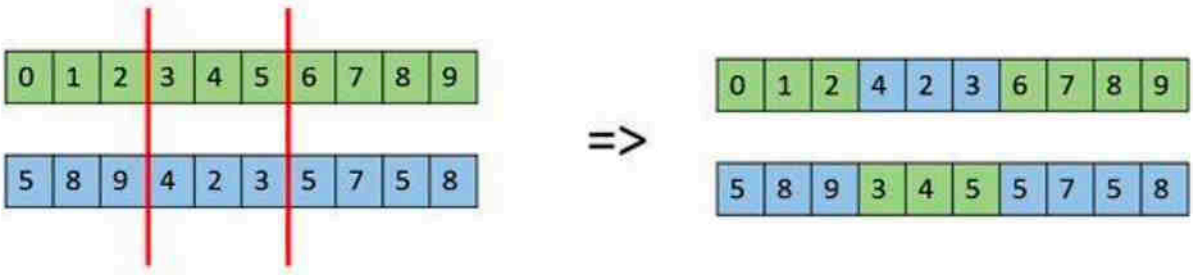


(ii) Two-Point Crossover



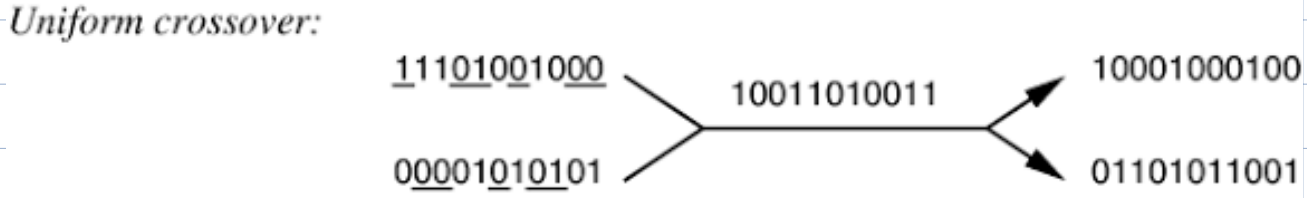
(iii) Multi-Point Crossover

- alternating segments swapped



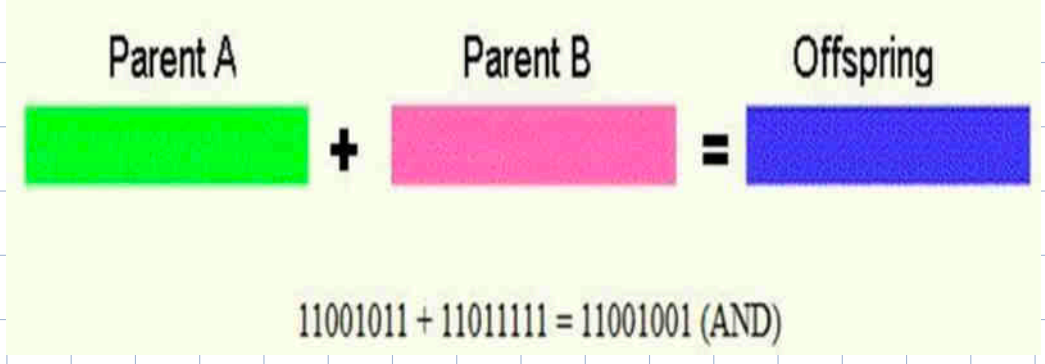
(iv) Uniform Crossover

- treat each gene separately
- 50% chance of going to either offspring



(v) Arithmatic / Logical crossover

- AND, OR, +

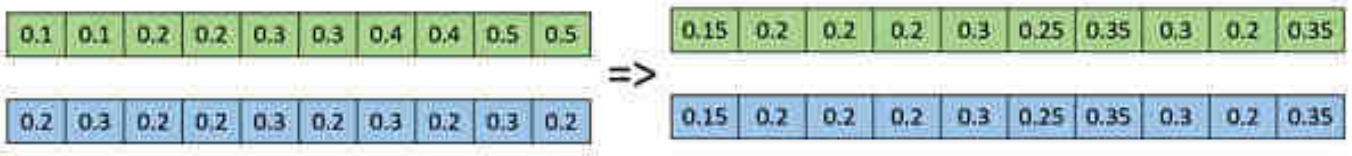


(vi) Whole arithmatic recombination

- weighted avg of 2 parents
- numeric representations

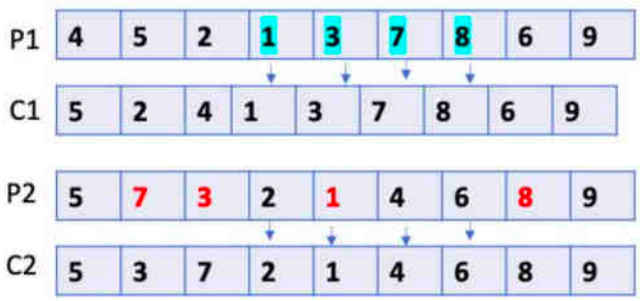
$$child_1 = \alpha\ Parent_1 + (1-\alpha)\ Parent_2$$
$$child_2 = \alpha\ Parent_2 + (1-\alpha)\ Parent_1$$

- gene-wise
- if $\alpha = 0.5$



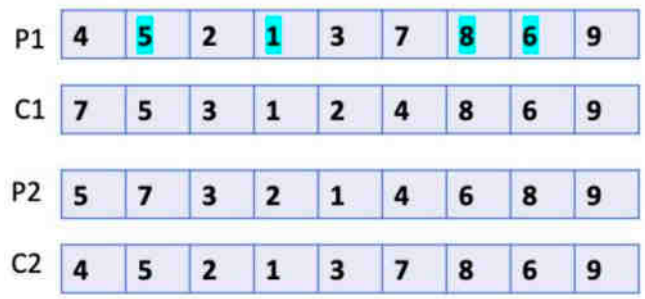
(vii) Order crossover

- Select random set of genes from P1
- Transfer to same pos of C1
- Take remaining genes from P2
- Transfer to same poc in C2
- Fill remaining gaps in C1 with the ones selected in P2 and vice versa

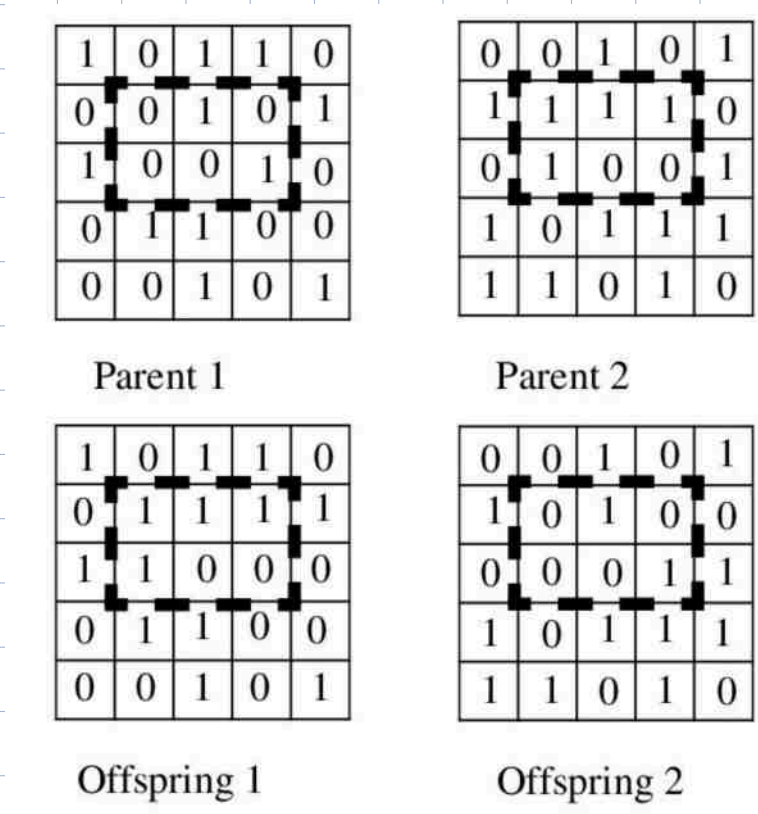


(viii) Position-based Crossover

- Select random positions of genes from P1
- Do same as order crossover

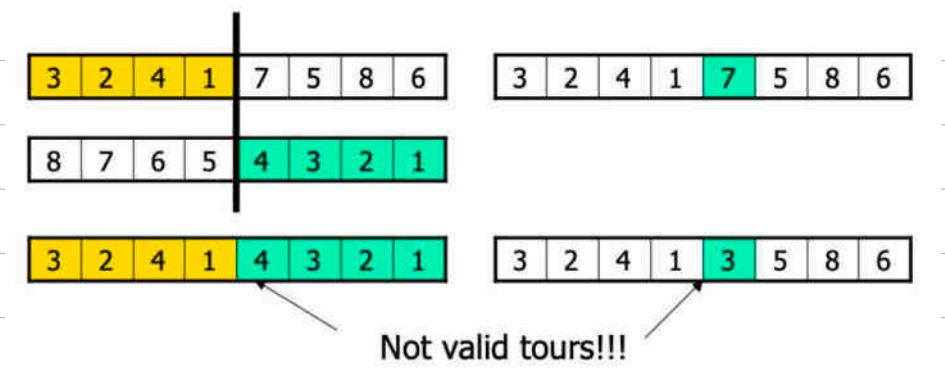


(ix) Matrix Crossover

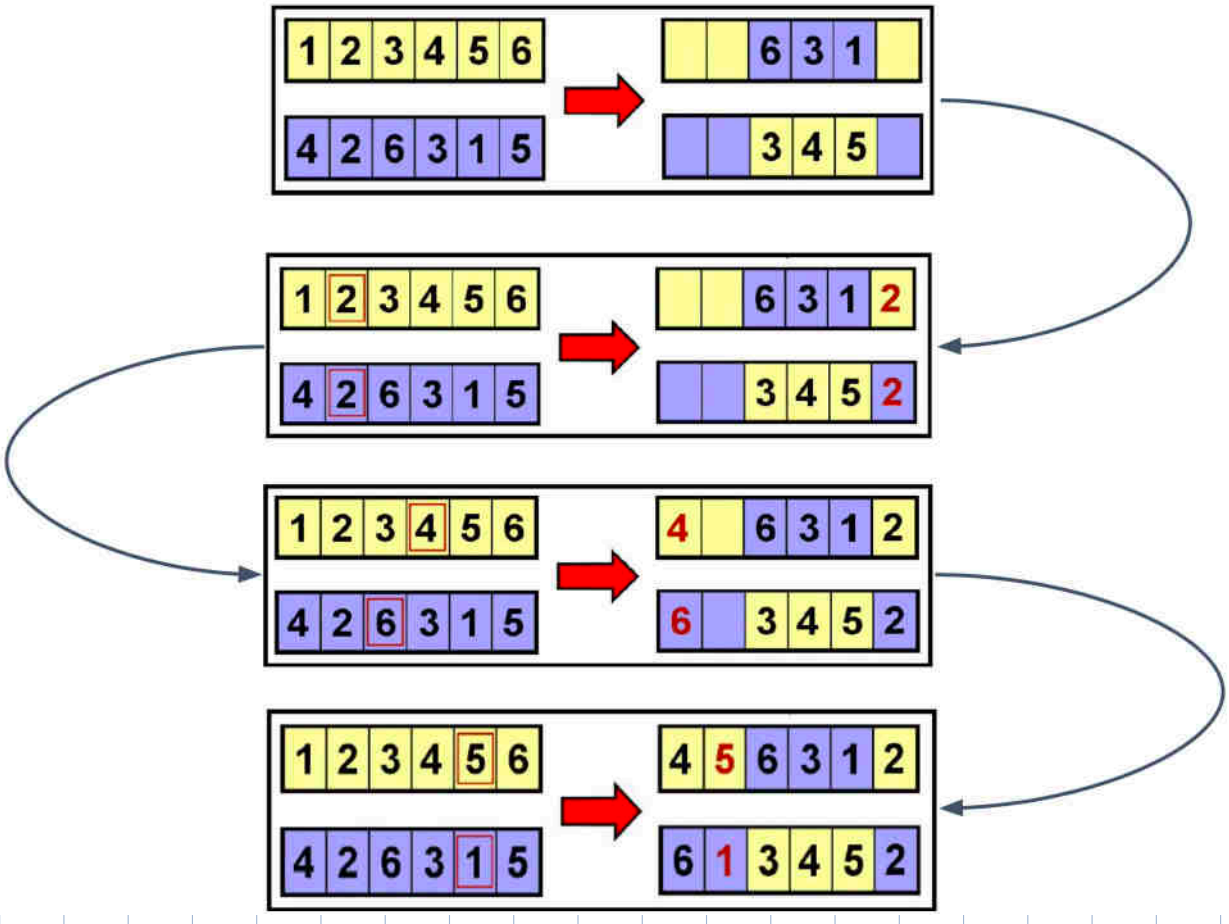


Problem with crossover and TSP

- Permutation as tour



- Solution: ordered crossover
  - 2 point crossover with wraparound



- Alternate solution: random key representation

  - simpler
  - generate n uniform random no.s b/w 0 and 1 to form chromosome
  - any crossover technique
  - visit cities in increasing order of genes

Random Keys:	0.64	0.92	0.14	0.35	0.76	0.86
Decoded As:	3	6	1	2	4	5

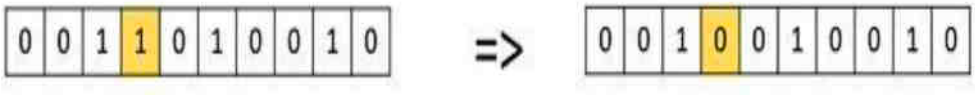
(3) Mutation

- Small random tweak in chromosome
- Used to maintain & introduce diversity in pop (very low probability)
- Mutation is essential for convergence while crossover is not
- Approaches

(i) Bit Flip
(ii) 1's complement
(iii) Random resetting/ uniform mutation
(iv) Boundary
(v) Swap
(vi) Scramble (shuffle)
(vii) Inversion
(viii) Insertion
(ix) Heuristic

(i) Bit Flip

- binary
- random bits chosen, flipped



(ii) 1's complement

- C = [01101]
- C' = [10010]

(iii) Random Resetting

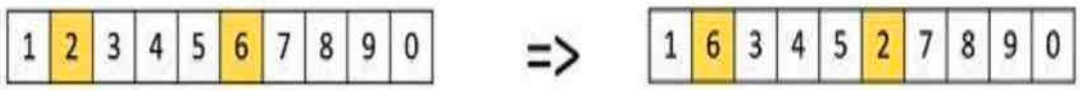
- bit flip for numeric
- random permissible value assigned

(iv) Boundary

- select random gene x_i
- assign upper/lower bound to it
- select random $r \in (0,1)$
- if $r \geq 0.5$, upper bound
  else, lower

(v) Swap

- select 2 positions and swap (permutation)



(vi) Scramble

- subset shuffled (permutation)

(vii) Inversion

- subset inverted

(viii) Insertion

- node from parent randomly selected
- inserted into random position

P_X=[2 5 4 9 6 8 1 3 7]

C_x =[2 6 5 4 9 8 1 3 7]

(ix) Heuristic

- random points chosen
- neighbours formed with lexicographic permutations
- best neighbour chosen

P_X=[1 2 3 4 5 6 7 8 9]

Neighbors are formed with these nodes.

N_X=[1 2 3 4 5 8 7 6 9]

N_X=[1 2 8 4 5 3 7 6 9]

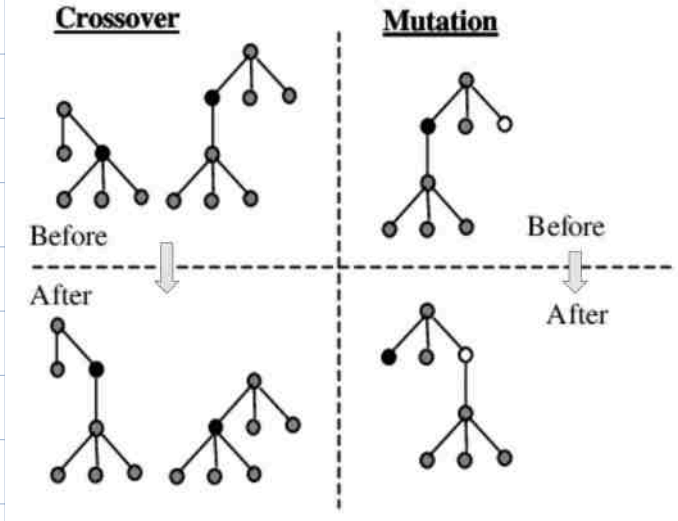
N_X=[1 2 8 4 5 6 7 3 9]

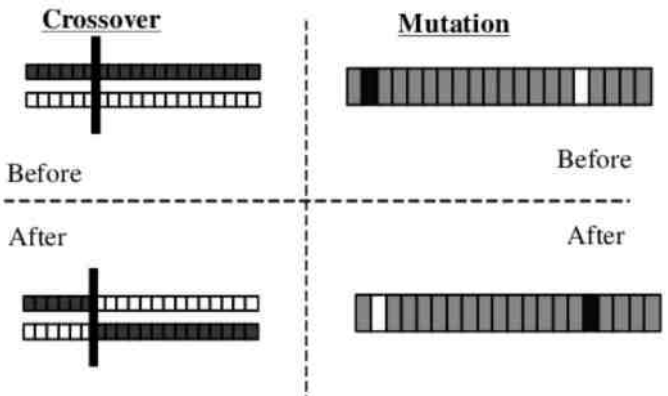
N_X=[1 2 6 4 5 8 7 3 9]

N_X=[1 2 6 4 5 3 7 8 9]

After evaluating all the neighbors the best neighbor is selected.

(x) Trees and Arrays





Termination Condition

- No improvement in fitness for X iters
- After a no. of generations
- Objective function reached pre-defined value

Q: Optimisation problem:

$$\min \; f(x) = x \sin(10\pi x) + 1.0$$

$$\text{s.t.} \quad -1 \le x \le 2$$

Let crossover rate be 0.5
Mutation rate be 0.1

One generation with pop = 10 chromosomes

- GAs can deal with non-smooth, discontinuous functions

Step 1

Initialisation and evaluation of population

- Compute cost (f(x)) for every initial value of x

- sort values of f(x) is ascending order

x	y=f(x)
-0.4691	1.3872
0.1572	0.8468
-0.5914	0.8422
0.6454	1.6387
0.7796	0.5339
1.0580	2.0248
0.0991	1.0028
0.2772	1.1820
0.5605	0.4697
1.6327	2.3962

→

x	y=f(x)
0.5605	0.4697
0.7796	0.5339
-0.5914	0.8422
0.1572	0.8468
0.0991	1.0028
0.2772	1.1820
-0.4691	1.3872
0.6454	1.6387
1.0580	2.0248
1.6327	2.3962

min f(x)
for x = 0.5605

step 2

Selection (Roulette wheel)

- Normalise values of f(x) between 0.1 and 0.9

$$f_N(x) = \frac{(\text{max range} - \text{min range})(f(x) - \text{min } f(x))}{\text{max } f(x) - \text{min } f(x)} + \text{min range}$$

min range = 0.1
max range = 0.9

min f(x) = 0.4697
max f(x) = 2.3962

x	y=f(x)	Normalized
0.5605	0.4697	0.1000
0.7796	0.5339	0.1269
-0.5914	0.8422	0.2584
0.1572	0.8468	0.2566
0.0991	1.0028	0.3213
0.2772	1.1820	0.3958
-0.4691	1.3872	0.4808
0.6454	1.6387	0.5855
1.0580	2.0248	0.7458
1.6327	2.3962	0.9000

- Normalise the normalised f(x) so they all add to 1
  - divide by Σ normalised

- Flip probabilities to get actual probability and compute cumulative prob

x	Normalized f(x)	Prob.	Flipped Prob.	Cumulative Probability
0.5605	0.1000	0.02339	0.2159	**0.2160**
0.7796	0.1269	0.03045	0.1789	**0.3949**
-0.5914	0.2584	0.06114	0.1405	**0.5354**
0.1572	0.2566	0.06157	0.1154	**0.6508**
0.0991	0.3213	0.0771	0.0950	**0.7458**
0.2772	0.3958	0.0950	0.0771	**0.8229**
-0.4691	0.4808	0.1154	0.06157	**0.8844**
0.6454	0.5855	0.1405	0.06114	**0.9456**
1.0580	0.7458	0.1789	.03045	**0.9760**
1.6327	0.9000	0.2159	0.02339	**1.000**

- Select using Roulette wheel technique

Chromosome #	x	Cumulative Probability
1	0.5605	0.2160
2	0.7796	0.3949
3	-0.5914	0.5354
4	0.1572	0.6508
5	0.0991	0.7458
6	0.2772	0.8229
7	-0.4691	0.8844
8	0.6454	0.9456
9	1.0580	0.9760
10	1.6327	1.000

Random Number	Chromosome # Selected	Chromosome Value
0.6777	5	0.0991
0.1389	1	0.5605
0.2345	2	0.7796
0.9805	10	1.6327
0.6190	4	0.1572
0.7784	6	0.2772
0.0579	1	0.5605
0.4111	3	-0.5914
0.8430	7	-0.4691
0.2513	2	0.7796

(5 crossovers)

step 3

cross over

- rate = 0.5 ⇒ no. of crossovers = 5
- Operator: directional operator

$$C_1 = P_1 + R(P_2 - P_1)$$
$$C_2 = P_2 + R(P_1 - P_2)$$

} R is random number

Parent 1	Parent 2	Random number for Directional : X	Crossover Operator
5th (0.0991)	1st (0.5605)	0.3451	C_5: 0.0991+0.3451(0.5605-0.0991)=0.2583 C_1: 0.5605+0.3451(0.0991-0.5605)=0.4013 (Better Value)
2nd (0.7796)	10th (1.6327)	0.1004	C_2: 0.7796+0.4539(1.6327-0.7796)=1.1668 (Better Value) C_{10}: 1.6327+0.4539(0.7796-1.6327)=1.2454
4th (0.1572)	6th (0.2772)	0.4539	C_4: 0.1572+0.8674(0.2772-0.1572)=0.2613 C_6: 0.2772+0.8674(0.1572-0.2772)=0.1731
1st (0.5605) (0.4013)	3rd (-0.5914)	0.7801	C_1: 0.4013+0.1004(-0.5914-0.4013)=0.3016 C_3: -0.5914+0.1004(0.4013+0.5914)= -0.4917
7th (-0.4691)	2nd (0.7796) (1.1668)	0.8674	C_7:0.8071 C_2:-0.1094

using new parents not necessary

if values not in range, do not replace parents

step 4

mutation

- select 3rd chromosome randomly and replace with random no in [-1,2]

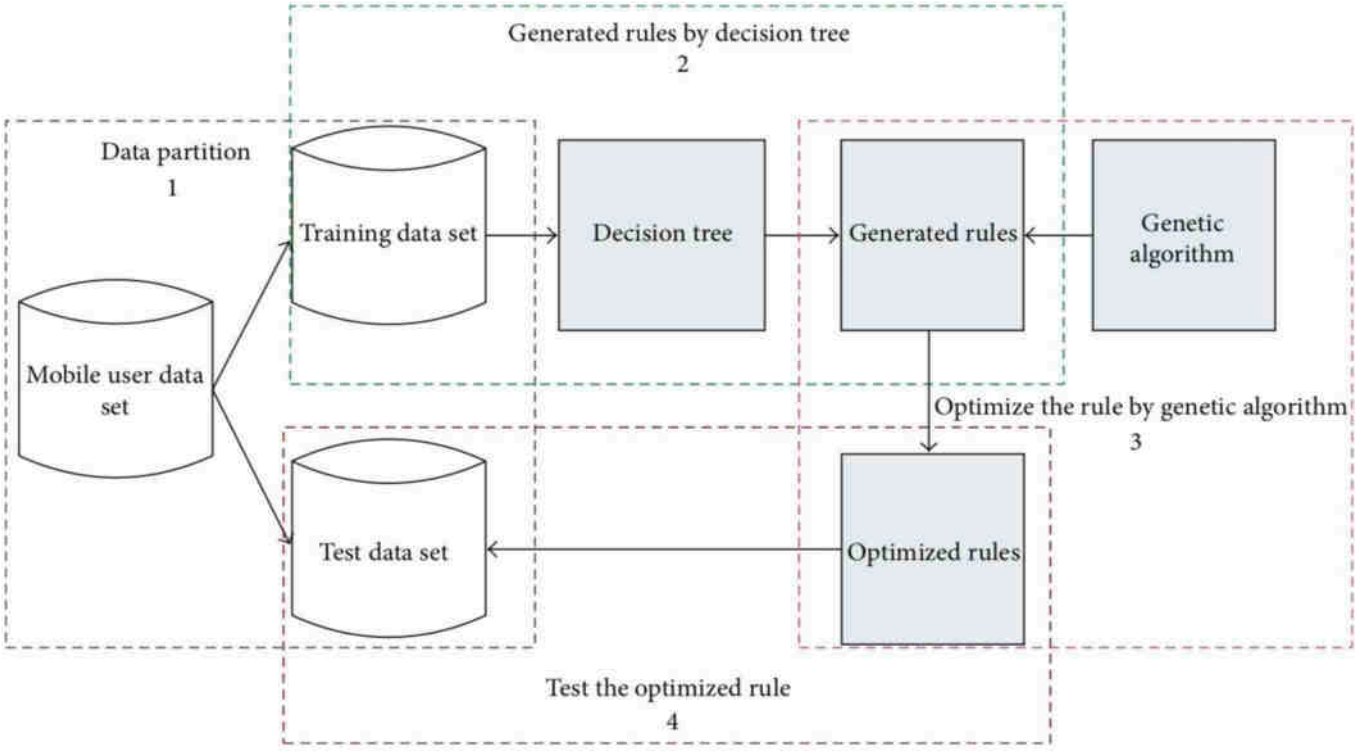
Population after crossover	Population after Mutation
0.3016	0.3016
-0.1094	-0.1094
-0.4917	-0.6283
0.2613	0.2613
0.2583	0.2583
0.1731	0.1731
0.8071	0.8071
0.6454	0.6454
1.0580	1.0580
1.2454	1.2454

step 5

termination

DECISION TREES using GA

1. Preprocessing
2. Rule generation from DT
3. Optimise rules using GA
4. Test optimised Rules



FEATURE SELECTION using GA

- String representing whether feature selected or not
- Run ML models and calculate accuracy

NEURO-GENETIC SYSTEMS (ANN + GA)

1. Weight determination
2. Evolves ANN topologies

Pros & Cons