

# *Genetic Algorithms*



Ujjwal Maulik

Dept. of Comp. Sc. & Engg.

Jadavpur University

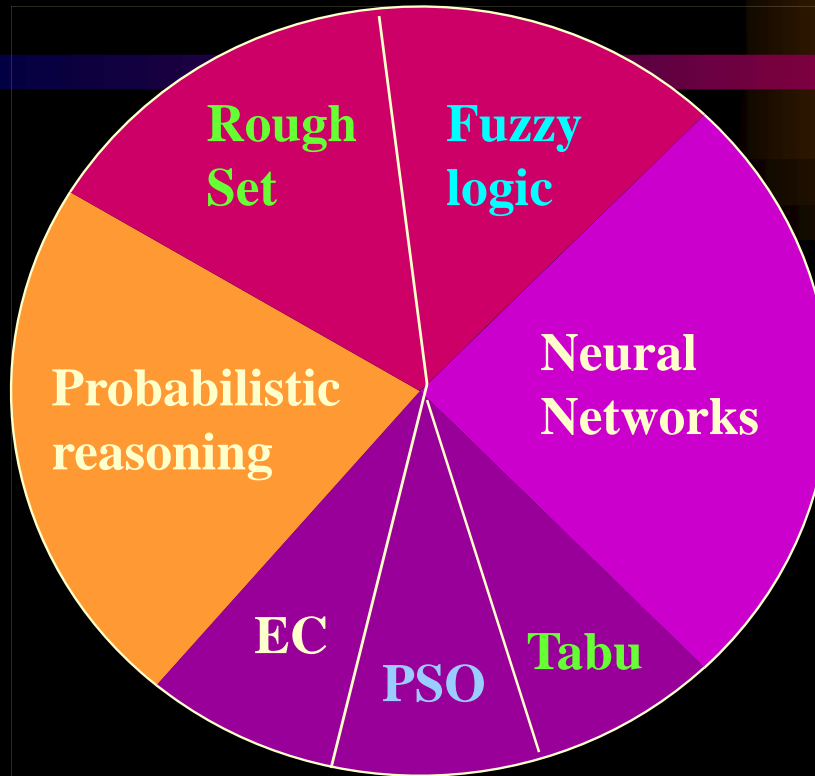
[umaulik@cse.jdvu.ac.in](mailto:umaulik@cse.jdvu.ac.in)

# *Soft Computing*



- Soft computing is a **consortium** of methodologies that works **synergistically** and provides in one form or another **flexible information processing capability** for handling **real life ambiguous situations**.
- As opposed to hard computing
  - Using linguistic/imprecise terms
  - Approximation handling
  - Near optimal solutions
  - Low cost
  - Mimicking natural systems
- Pioneer of Soft Computing → Prof. L. A. Zadeh

# *Components*



Tabu search, ant colony optimization, particle swarm optimization, rough sets, memetic algorithms AND HYBRIDIZATIONS

# *GENETIC ALGORITHMS*



- **DEFINITION**

Randomized search and optimization technique guided by the principles of natural genetic systems.

- **Why Genetic Algorithms (GAs) ?**

1. Evolution produced good individuals  
might work for solving complex problems
2. Most real life search and optimization problems can not be solved in polynomial amount of time using any deterministic algorithm
3. Near optimal solutions requiring less time more desirable than optimal solutions with huge amount of time

# *Example: Traveling Salesman Problem (TSP)*

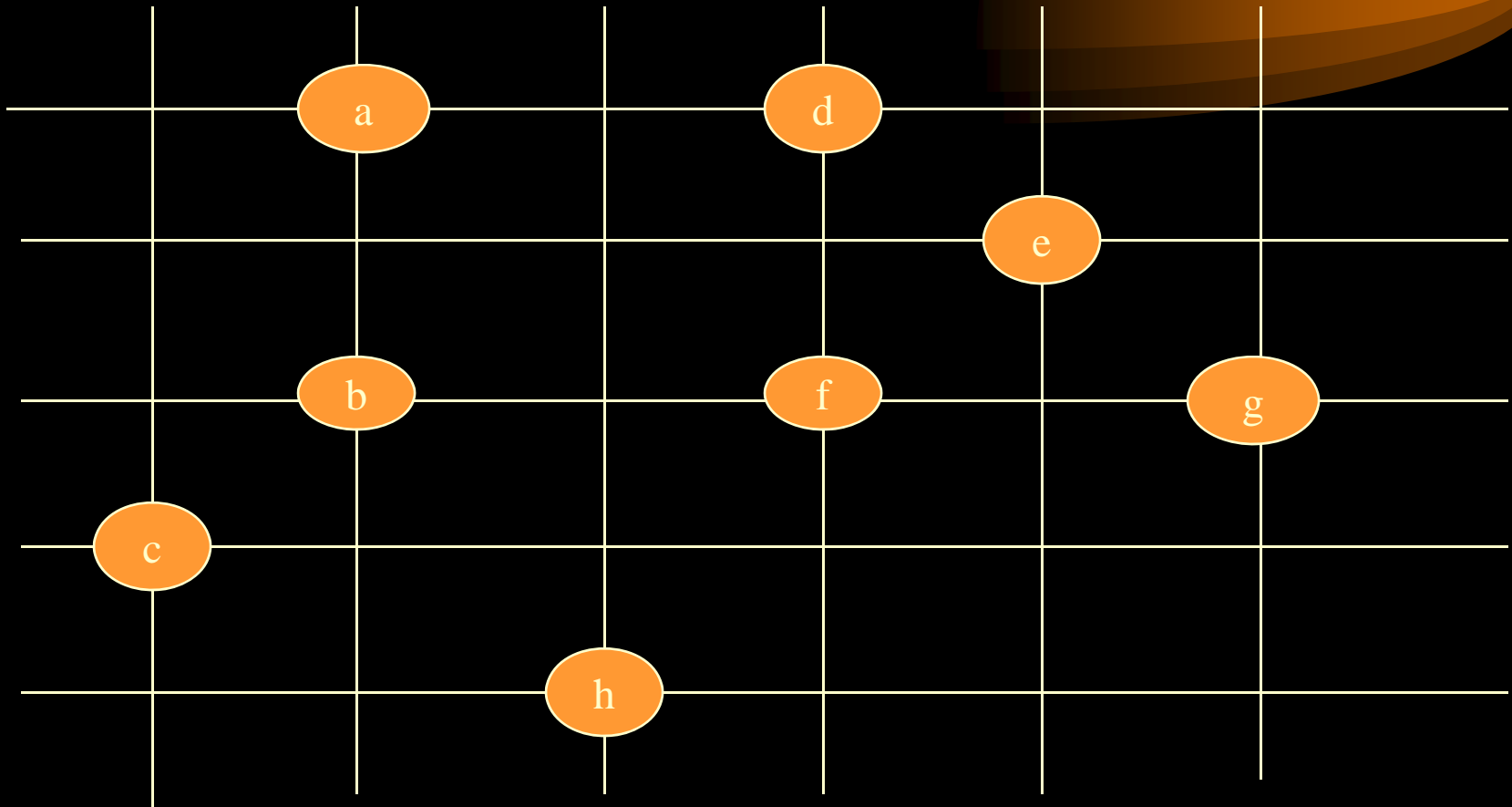
## **Problem Definition :**

Given a complete graph  $G = (V, E)$  that has a nonnegative integer cost  $c(u, v)$  associated with edge  $(u, v) \in E$ , one will have to find a hamiltonian cycle (a tour) of  $G$  with minimum cost.

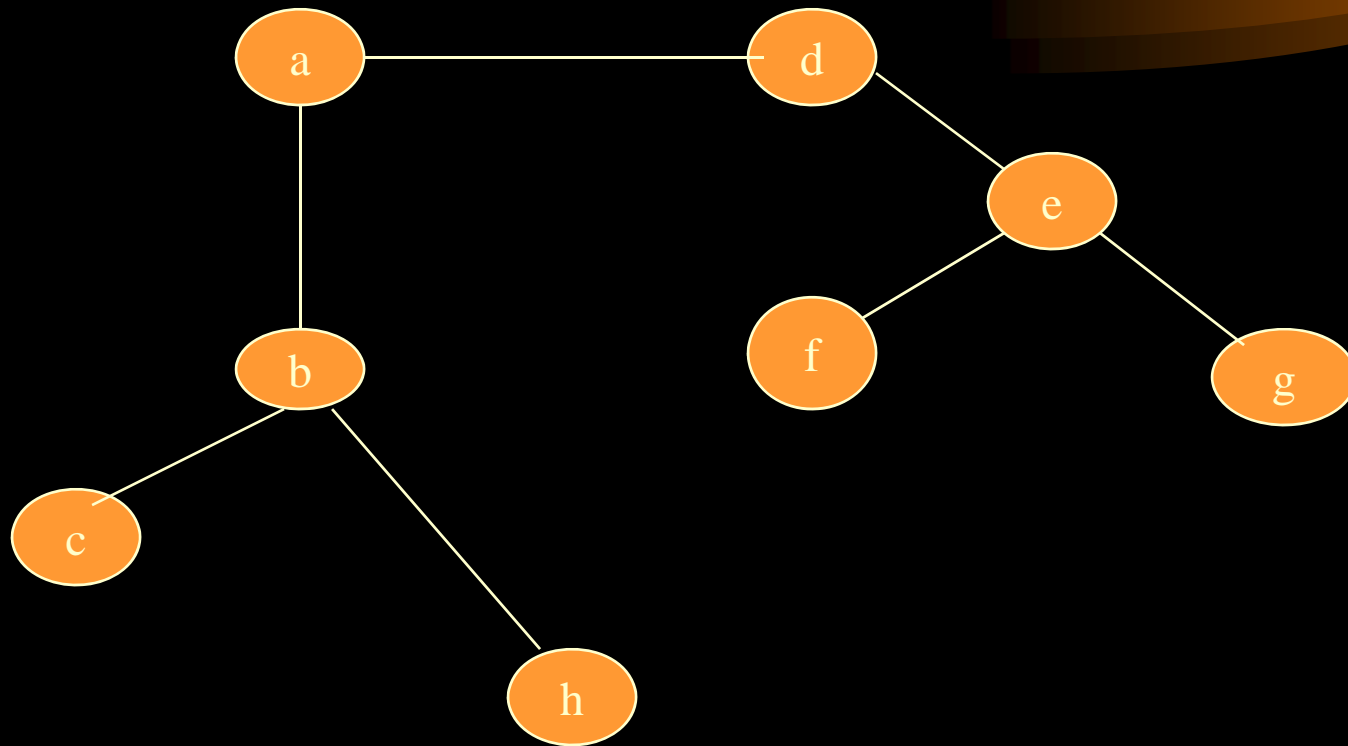
## **Hamiltonian cycle :**

Hamiltonian cycle of a undirected graph  $G = (V, E)$  is a simple cycle that contains each vertex in  $V$ .

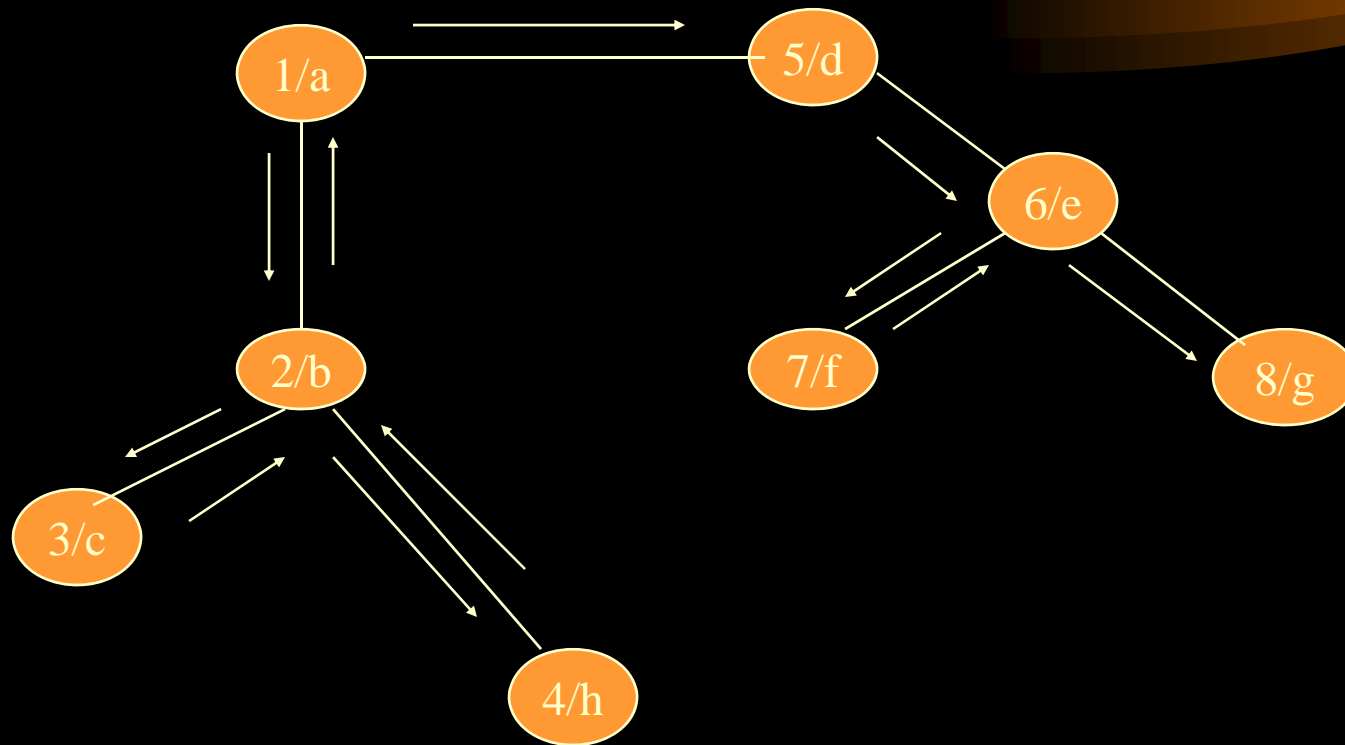
# *Some Heuristic to Solve TS*



# *A minimum spanning tree as computed by Prim's algorithm*

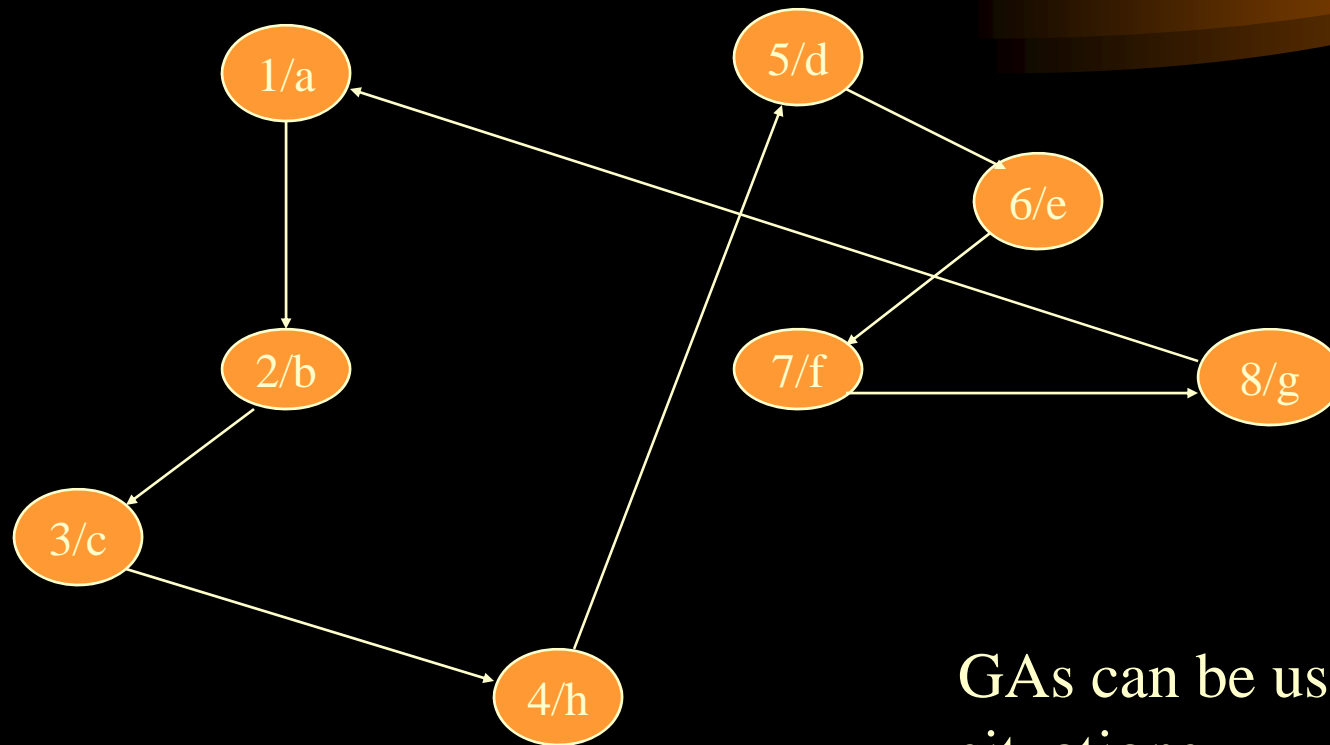


*A preorder traversal of the above minimum spanning tree (abcbhbadefeg)*





*Hamiltonian circuit following preorder traversal  
cost = 19.074, however the optimal cost is =14.715)*

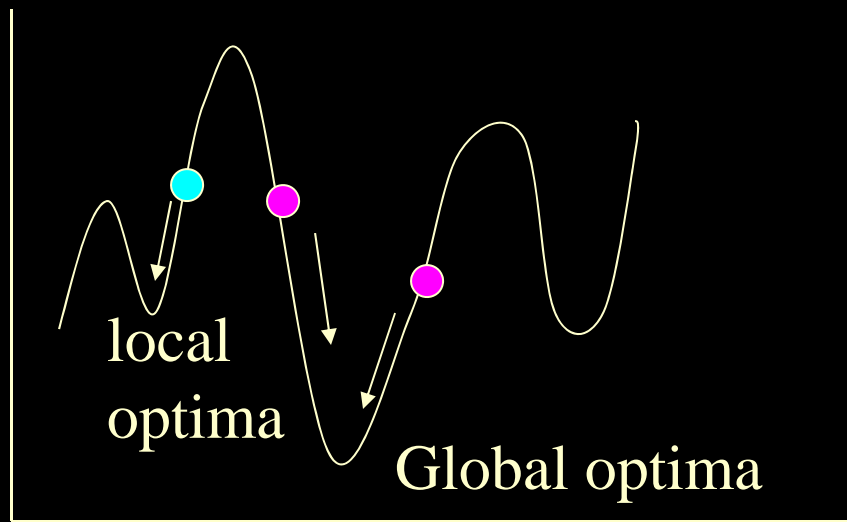


GAs can be used in such situations

# *Search Techniques*

*The traditional vs. the unconventional*

- Calculus based techniques – gradient descent  
(hill climbing)

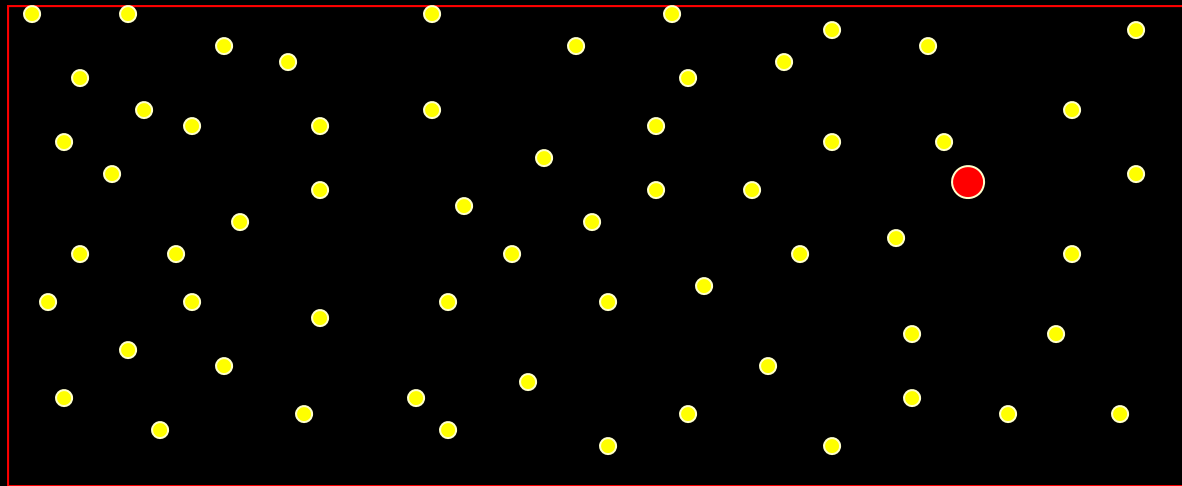


Continuous domain, quadratic optimization – best method

# *Search Techniques*

*The traditional vs. the unconventional*

- Enumerative technique – dynamic programming



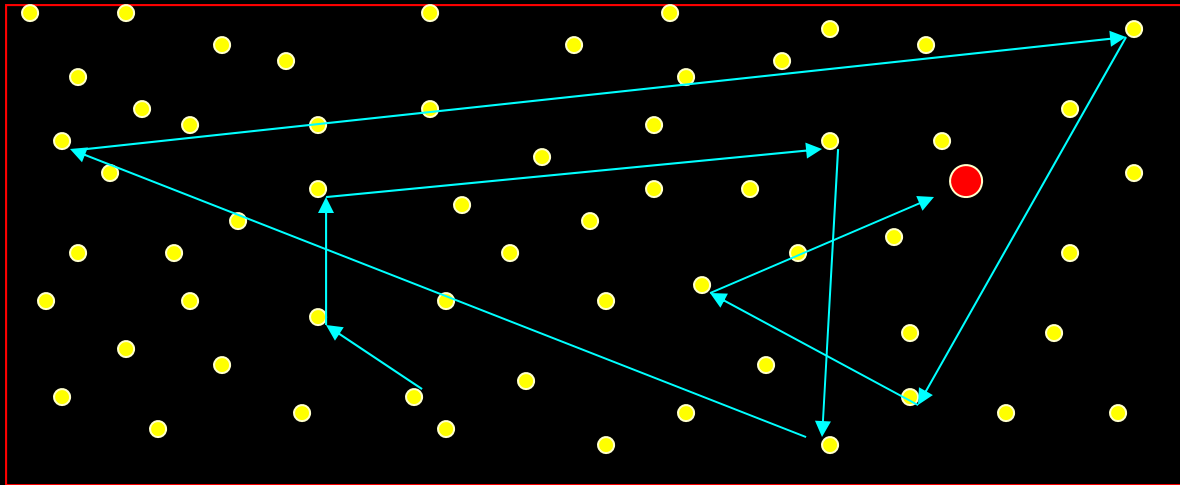
$n$  points

What if  $n$  very very large? Quite likely in practice.

# *Search Techniques*

*The traditional vs. the unconventional*

- Random technique – hoping to find the optimal



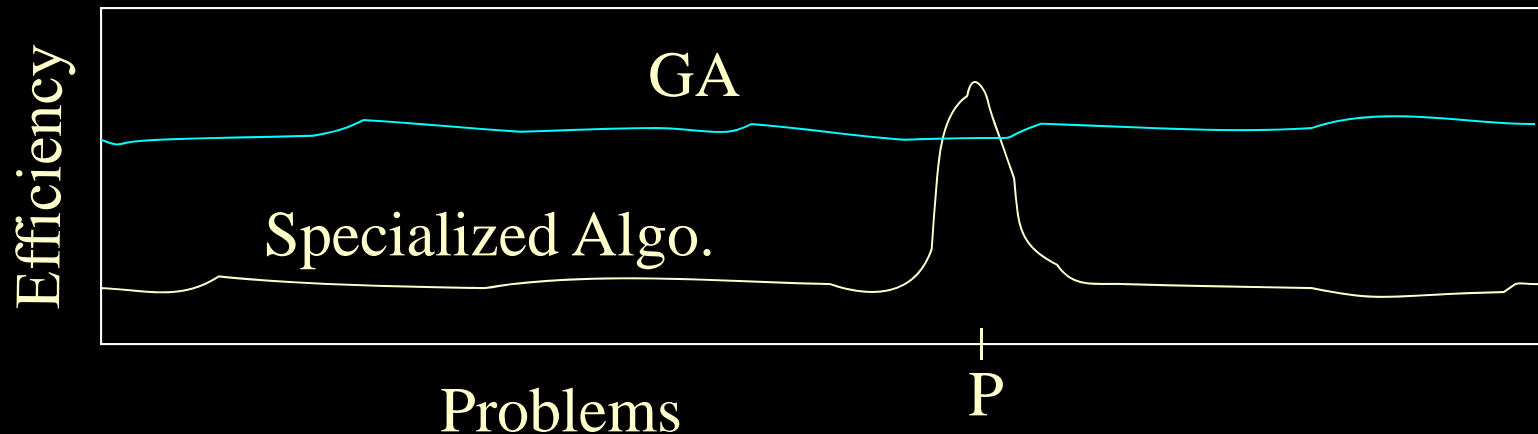
No better than enumerative in the long run

# *Randomized Algorithms*

- Guided random search technique
- Uses the payoff function to guide search



Genetic Algorithms (GAs)



Specialized algorithms – best performance for special problems  
Genetic algorithms – good performance over a wide range of problems

# *Genetic Algorithms - Features*

- Evolutionary Search and Optimization Technique
- Principles of Evolution (*survival of the fittest* and *inheritance*)
- Work with coding of the parameter set
- Searches from a population of points
- Uses probabilistic transition rules

# *Genetic Algorithms* ↔ *Nature*



- A solution (phenotype) Individual
- Representation of a solution (genotype) Chromosome
- Components of the representation Genes
- Set of solutions Population
- Survival of the fittest (selection) Darwins theory
- Search operators Crossover and mutation

# Simple Generational GA

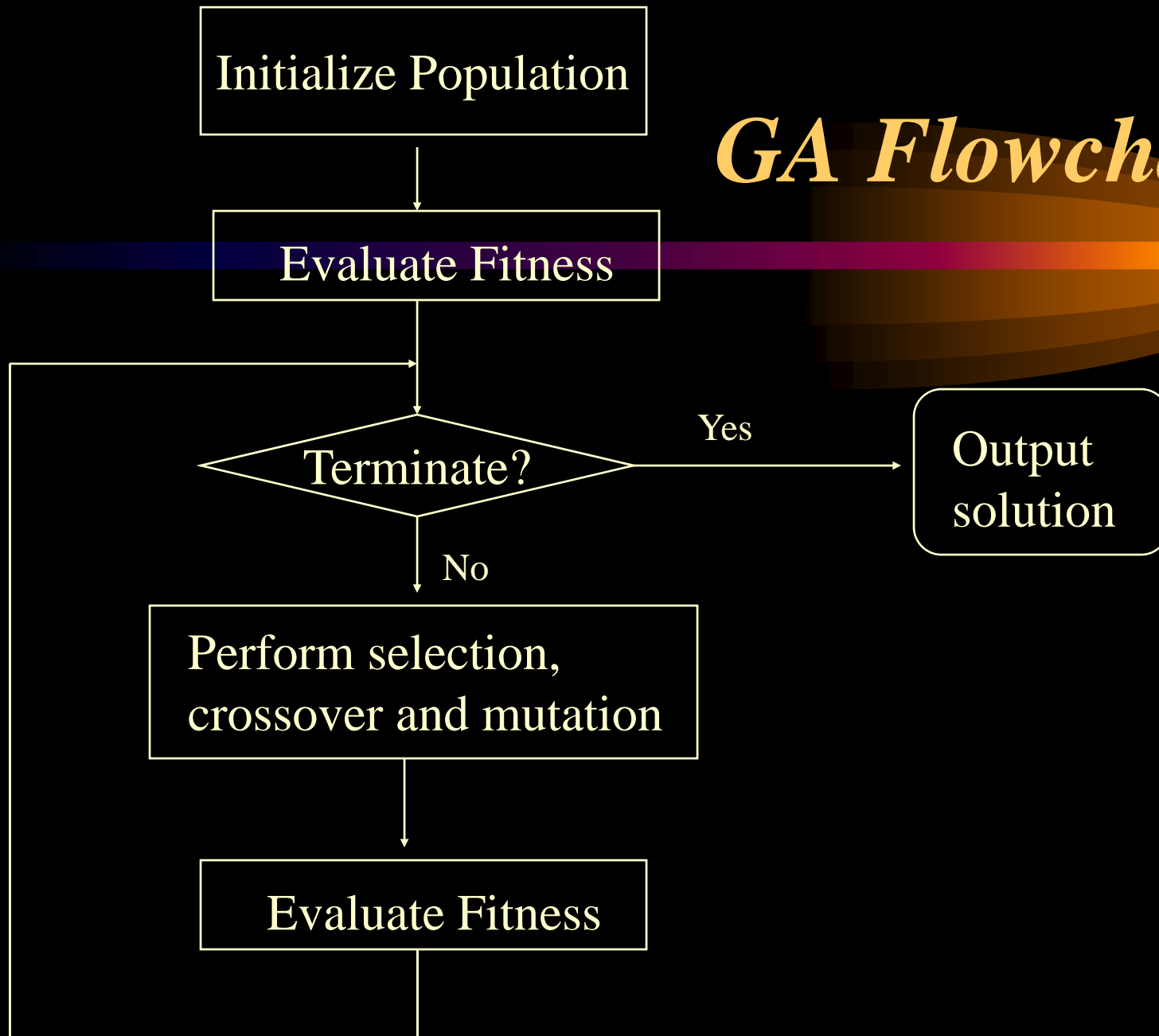


1. Randomly generate a population of chromosomes
2. Decode each chromosome to get an individual
3. Evaluate the fitness of each individual
4. Perform selection, crossover and mutation.
5. Repeat steps 2, 3 and 4 until a stop condition is true.

*Note :* There is no overlapping between generations.



# *GA Flowchart*



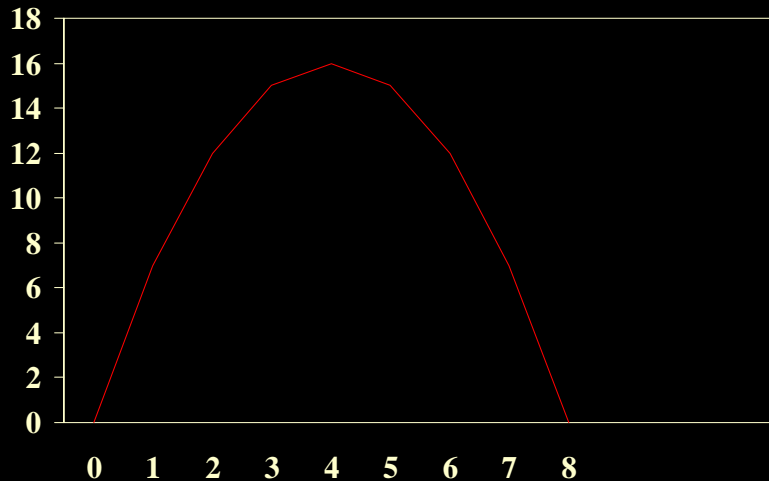
# *Encoding Strategy and Population*

- **Chromosome encodes a solution in the search space**
  - Usually as strings of 0's and 1's
  - If  $l$  is the string length, number of different chromosomes (or strings) is  $2^l$
- **Population**
  - A set of chromosomes in a generation
  - Population size is usually constant
  - Common practice is to choose the initial population randomly.

# Encoding and Population - Example

Optimization Problem :

$$\text{Optimize } f(x) = x(8 - x), \quad x=[0,8]$$

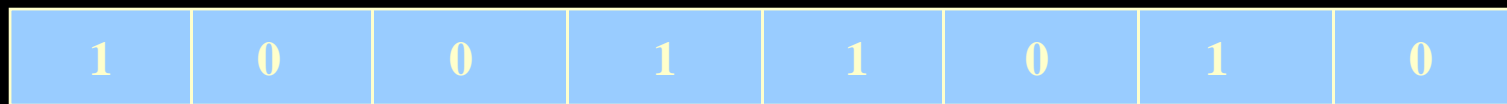


User specified parameters

Binary String of 8 bits

0-255  $\longleftrightarrow$  0-8

Chromosome  
encodes  $x$



Value = 154



$$8/255 * 154 + 0 = 4.8313$$

# *Encoding and Population – Example*

*contd...*

**Population (size = 4)**

**Corresponding x**

1 0 0 1 1 0 1 0

4.8313

0 1 1 0 0 1 1 1

3.2313

0 0 0 1 0 1 0 1

0.6588

1 0 1 1 1 1 0 0

5.8980

# *Fitness Evaluation*

- Fitness/objective function associated with each chromosome
- indicates the degree of goodness of the encoded solution
- only problem specific information (also known as the payoff information) that GAs use
- If minimization problem is to be solved then **fitness  $\propto$  1/objective**.

# *Fitness Evaluation - Example*

**Function**  $f(x) = x(8-x)$

**Population (size = 4)**

**Corresponding x**

**Objective/  
Fitness fn.**

1 0 0 1 1 0 1 0

4.8313

15.3089

0 1 1 0 0 1 1 1

3.2313

15.4091

0 0 0 1 0 1 0 1

0.6588

4.8363

1 0 1 1 1 1 0 0

5.898

12.3975

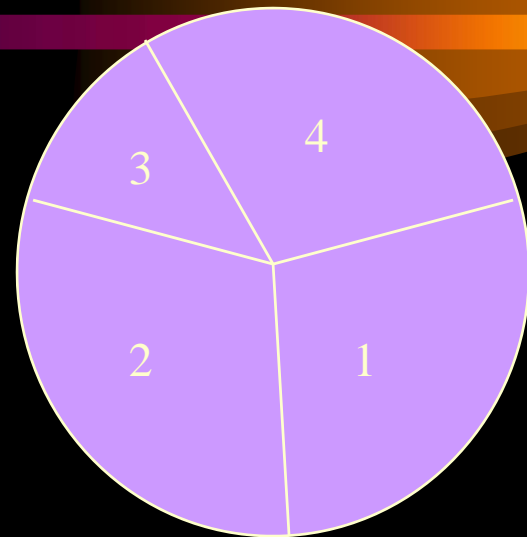
# Selection



- More copies to good strings
- Fewer copies to bad string
- proportional selection scheme
  - Number of copies taken to be directly proportional to its fitness
  - mimicks the natural selection procedure to some extent.
  - *Roulette wheel parent selection and stochastic universal selection* selection are two frequently used selection procedures.

# *Roulette Wheel Selection – Example*

Chromosome #	Fitness
1	15.3089
2	15.4091
3	4.8363
4	12.3975



Spin 1	Chromosome 2 selected
Spin 2	Chromosome 1 selected
Spin 3	Chromosome 2 selected
Spin 4	Chromosome 4 selected

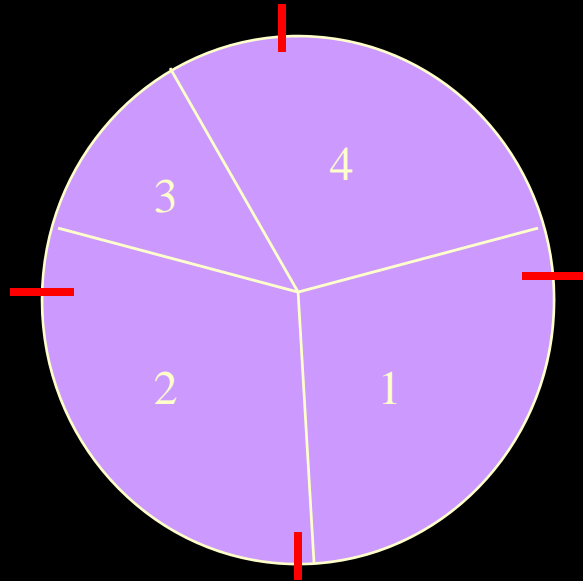
Mating  
→  
Pool

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



# *Stochastic Universal Selection-*

## *Example*



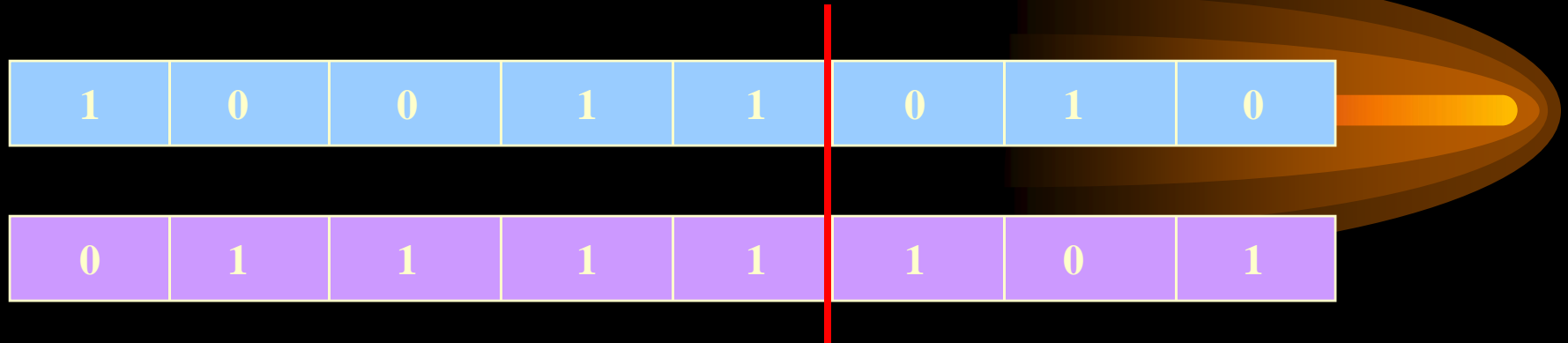
Chromosome 1	1 copy
Chromosome 2	2 copies
Chromosome 3	0 copies
Chromosome 4	1 copy

# *Crossover*



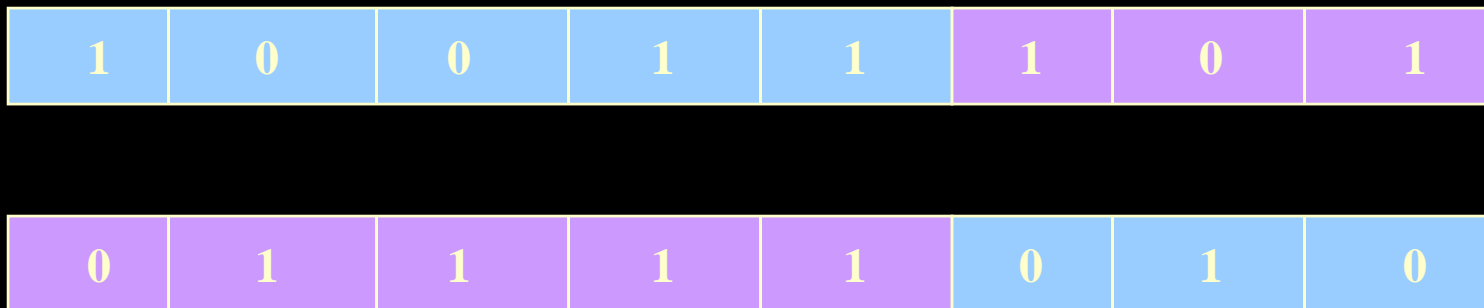
- **exchange information**
  - between randomly selected parent chromosomes
  - Single point crossover is one of the most commonly used schemes.
  - probabilistic operation

# Crossover – Example



Here  $l$  (string length) = 8. Let  $k$  (crossover point) = 5

Offspring formed :




# *Mutation*

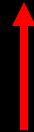
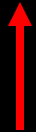


- **random alteration** in the genetic structure
  - introduce **genetic diversity** into the population.
  - Exploration of new search areas
  - Mutating a binary gene involves simple **negation of the bit**,
  - Mutating a real coded gene defined in a variety of ways
  - probabilistic operation

# *Mutation- Example*



1	0	0	1	1	0	1	0
---	---	---	---	---	---	---	---



Mutations at positions 2 and 5

1	1	0	1	0	0	1	0
---	---	---	---	---	---	---	---

# *Parameters ...*

- population size (usually fixed)
- string length (usually fixed)
- probabilities of performing crossover ( $\mu_c$ ) and mutation( $\mu_m$ ),
  - $\mu_c$  is kept high and  $\mu_m$  is kept low
- the termination criteria
  - Generally a maximum number of iterations
- parameters are user determined
- parameters are problem dependent
- no firm guidelines
- parameters can be kept **variable and/or adaptive**.

# *Parameters – Example*

For the example being considered,

$$P = 4, \quad l = 8.$$

But for most realistic cases

$P$  is usually chosen in the range 50-100.

$$\mu_c = [0.6-0.9],$$

$$\mu_m = [0.01-0.1].$$

$l$  usually depends on the required precision

# *Termination Criterion*

The cycle of selection, crossover and mutation is repeated a number of times till:

- the average fitness value of a population becomes more or less constant over a specified number of generations,
- a desired objective function value is attained by at least one string in the population,
- the number of generations (or iterations) is greater than some threshold ----- most commonly used.



# *Elitist Model of GAs*

A decorative graphic consisting of a horizontal bar with a color gradient from dark blue on the left to bright yellow on the right. To the right of the bar is a large, stylized, comet-like shape with a gradient from dark brown to light yellow, pointing towards the right.

The best string (in terms of fitness) seen up to the current generation is preserved in a location either inside or outside the population.

# *Theory of GAs*



- Why bother
  - To look into issues of convergence to the global optima
    - Performance guarantees
  - Might aid better algorithm design
    - Increased understanding can be gained about operator interplay etc.
  - Might be used to explain certain phenomenon to theoretical biologists
- Challenges
  - Vast, complex dynamical systems with many degrees of freedom
    - Probabilistic techniques need to be used
    - Can only define average behaviour, not individual behaviour
  - Problems are hard to model and analytically solve

# *Theoretical Analysis of GAs: Background*

- Schema is a template that may match several chromosomes
  - For 8 bit chromosomes
  - A schema  $S$  - \*10\*\*\*\*\*
  - Matches all chromosomes with a 1 and a 0 in the 2<sup>nd</sup> and 3<sup>rd</sup> positions
- Length of schema
  - Difference between last and first defining position
    - Length of  $S = 1$
- Order of schema
  - Number of defined positions
    - Order of  $S = 2$
- Average fitness of schema
  - Average fitness of all chromosomes matching the schema

# Schema – Example

<i>Schema</i>	<i>Matched Strings</i>	<i>Order</i> $O(h)$	<i>Length</i> $\delta(h)$
#10#	0100 0101 1100 1101	2	$(3-2) = 1$
###	000 001 010 011 100 101 110 111	0	-
1001	1001	4	3
1#1	101 111	2	$(3-1)=2$
##1#	0010 0011 0110 0111 1010 1011 1110 1111	1	$(3-3)=0$

# *Schema – Example contd...*

<i>String</i>	<i>Contained in Schema</i>
0	0 #
100	100 10# 1#0 #00 1## #0# ##0 ###
10	10 1# #0 ##

# Schema Analysis – contd...

String length  $l$ , alphabet size  $k$   $\longrightarrow (k+1)^l$  schemata

**For binary strings of length  $l$  there are  $3^l$  different schemata**

Binary schemata of Order  $O(h)$   $\longrightarrow$  matches  $2^{(l-O(h))}$   
strings of length  $l$

A bit string is contained in  $2^l$  different schemata.

Population of size  $P$  contains between  $2^l$  to  $\min(P * 2^l, 3^l)$  schemata



**implicit parallelism**  $\longrightarrow$

GA works with many

more schemata than the population size.

Example:  $P=300, l=100$   $\longrightarrow 1.267650 * 10^{30} - 3.802951 * 10^{32}$

# *Geometric Interpretation of Schemata*

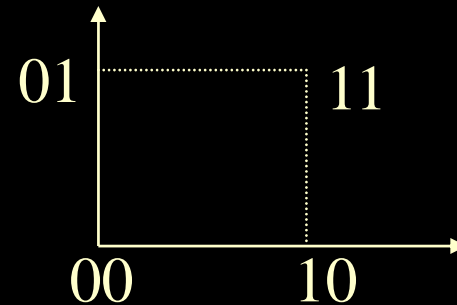
A string of length  $l$  can be considered to be a point in an  $l$  dimensional space.

Example:

Strings of length 1 =  $\{0,1\}$

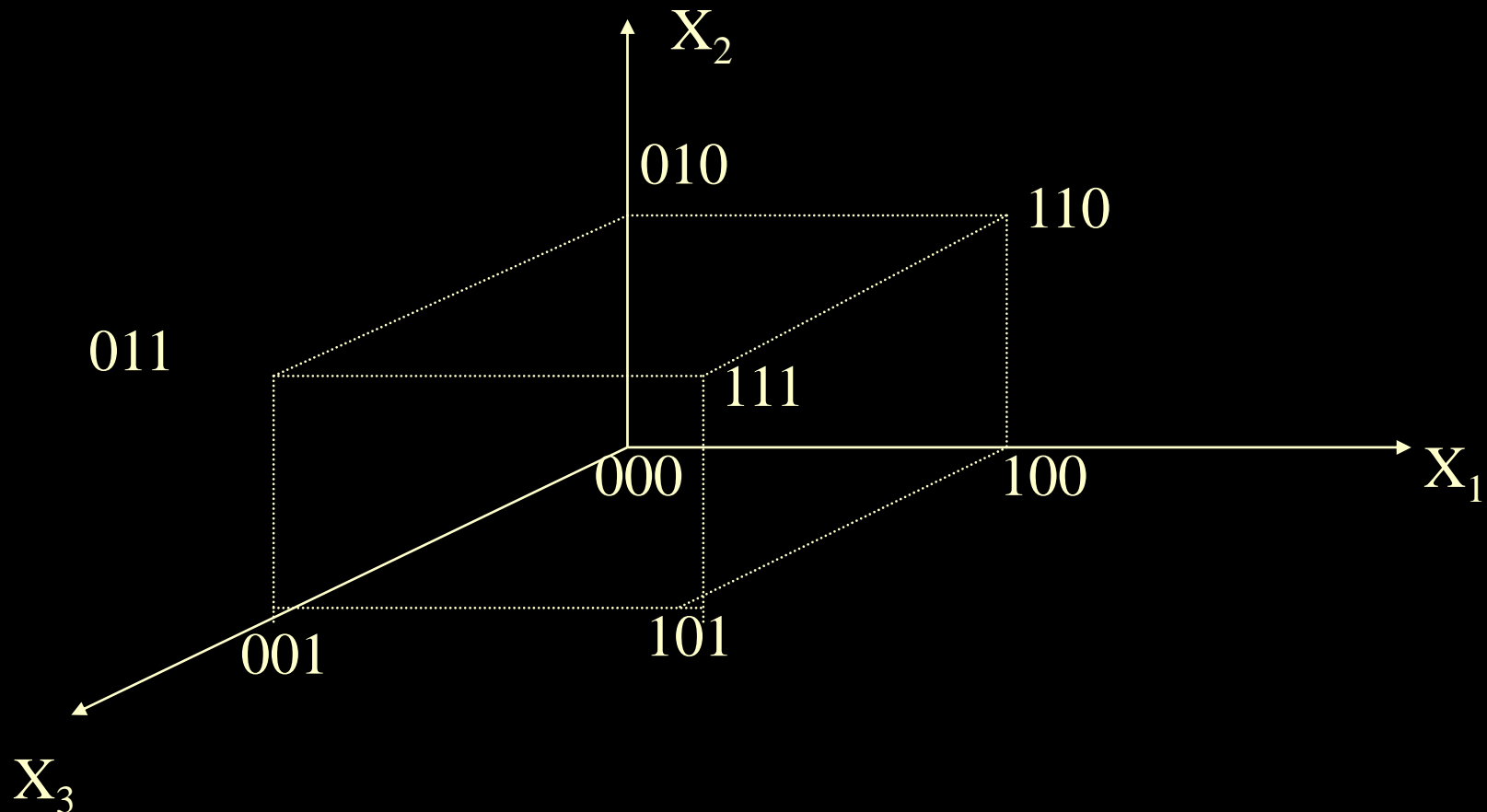


Strings of length 2 =  $\{00, 01, 10, 11\}$



# *Geometric Interpretation of Schemata - contd...*

Strings of length 3 = {000, 001, 010, 011, 100, 101, 110, 111}

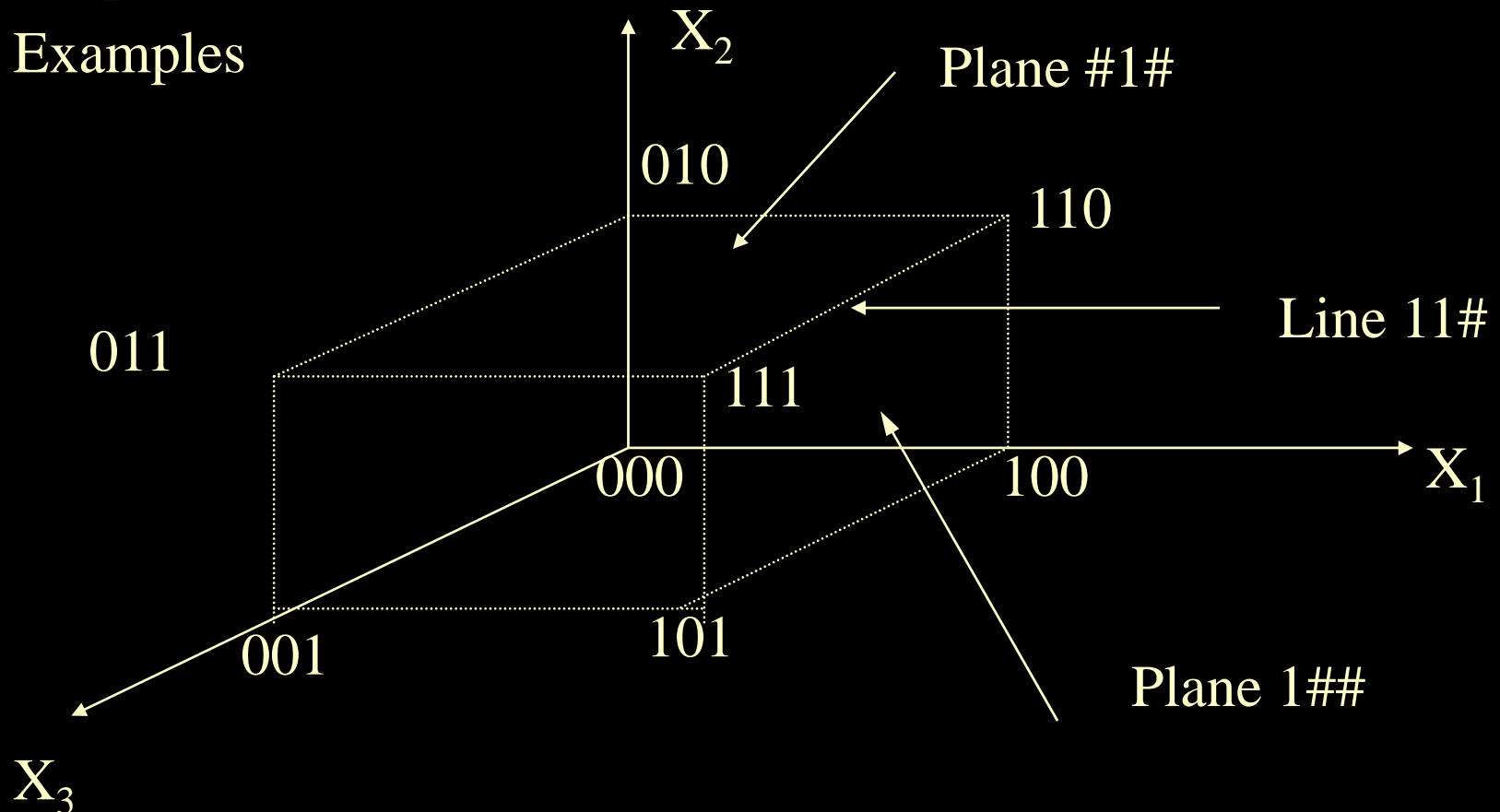




# *Geometric Interpretation of Schemata - contd*

Schemata can be visualized as hyperplanes in such search space.

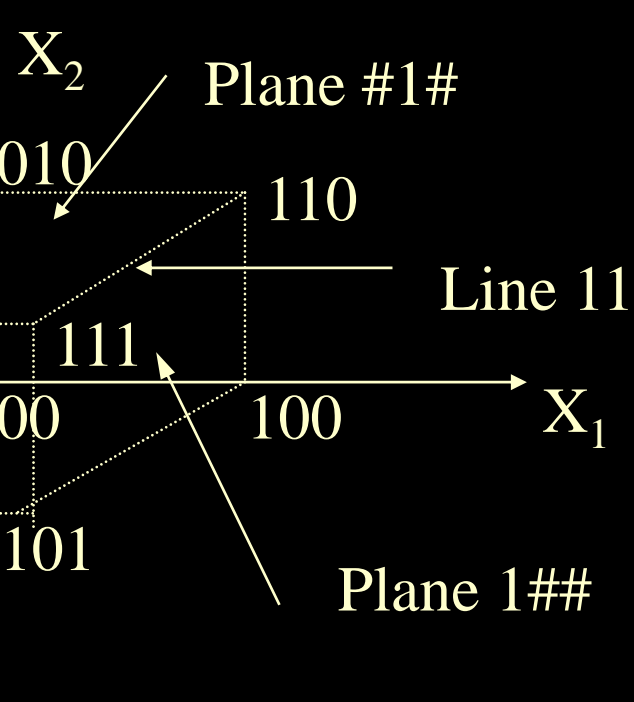
Examples



*nata - contd*



ness of the chromosomes



# *Schema Analysis – contd ...*

Given a population, the fitness of a schema can be estimated by finding the average fitness of strings contained in (or sampled by) the schema.

Example:

Population	fit
1 0 0 1 1 0 1 0	4.8313
0 1 1 0 0 1 1 1	3.2313
0 0 0 1 0 1 0 1	0.6588
1 0 1 1 1 1 0 0	5.898

Schema	fitness
#0#11###	$(4.8313+5.898)/2$
#####1	$(3.2313+.6588)/2$
#####1##	$(3.2313+.6588+5.898)/3$

# Schema Theorem: Selection

In a population, let

- $m(h,t)$ : the number of strings contained in schema  $h$  at generation  $t$ .
- $f(h,t)$ : the average fitness of the above  $m(h,t)$  strings sampling  $h$ .

## Proportional selection :

Each string gets selected with probability  $p_i = f_i / \sum f_j$ .

Since selection is with replacement, so each string can have  $P * f_i / \sum f_j$  copies, where  $P$ =population size.

The number of instances of schema  $h$  at generation  $t+1$  is therefore

$$m(h,t+1) = m(h,t) * P * f(h) / \sum f_j = m(h,t) * f(h) / f_{av}$$

where  $f_{av} = \sum f_j / P$  is the average fitness of the population.

**Schemata with above (below) average fitness are sampled, generation after generation, by an increasing (decreasing) number of chromosomes.**

# *Schema Theorem: Selection – contd...*

Analyzing the equation

$$m(h, t+1) = m(h, t) * f(h) / f_{av}$$

if a schema  $h$  remains above or below the average of a factor  $c$  in all generations, then  $f(h) = f_{av} + c * f_{av}$ . Then

$$m(h, t+1) = m(h, t) * (f_{av} + c * f_{av}) / f_{av}.$$

Starting from  $t=0$ , we get

$$m(h, t+1) = m(h, 0) * (1 + c)^t.$$

**This shows that above (below) average schemata will receive an exponentially increasing (decreasing) number of instances in subsequent generations.**

# *Schema Theorem: Selection - Example*

1	0	0	1	1	0	1	0	15.3089
0	1	1	0	0	1	1	1	15.4091
0	0	0	1	0	1	0	1	4.8363
1	0	1	1	1	1	0	0	12.3975

Mating  
→  
Pool

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

# Schema Theorem: Crossover

$\delta(h)$ : defining length of a schema  $h$  and

$l$ : length of the chromosomes

probability that a random crossover point is between the defining bits of the schema is:

$$\delta(h)/(l-1).$$

All such crossover points may disrupt the schema  $h$ . **Why may?**

Therefore the **max. probability of disruption of schema  $h$**  is

$$p_d = \delta(h)/(l-1).$$

Therefore the **min. probability of survival of schema  $h$**  is

$$p_s = 1 - \delta(h)/(l-1)$$

If crossover is itself performed with probability  $\mu_c$ , then **probability of survival of schema  $h$**  is

$$p_s \geq 1 - \mu_c * \delta(h)/(l-1).$$

# *Schema Theorem: Selection and Crossover*



Therefore the combined effect of selection and crossover is

$$m(h, t+1) \geq m(h, t) * f(h)/f_{av} * [1 - \mu_c * \delta(h)/(l-1)].$$



# *Schema Theorem: Mutation*

Let  $O(h)$  is the order of a schema  $h$ .

Let mutation be applied with probability  $\mu_m$ ,  
then probability that a bit survives mutation is  
 $(1 - \mu_m)$ .

Therefore probability that  $O(h)$  survives mutation is  
 $(1 - \mu_m)^{O(h)}$

This is the probability that the schema  $h$  survives mutation.

When  $O(h)$  is small,

$(1 - \mu_m)^{O(h)}$  can be written as  $1 - O(h)\mu_m$ .

# *Schema Theorem: Selection, Crossover and Mutation*



Therefore the combined effect of selection, crossover and mutation is

$$m(h, t+1) \geq m(h, t) * f(h) / f_{av} * [1 - \mu_c * \delta(h) / (l-1)] * (1 - O(h)\mu_m).$$

Ignoring small cross product terms we get

**Schema Theorem:**

$$m(h, t+1) \geq m(h, t) * f(h) / f_{av} * [1 - \mu_c * \delta(h) / (l-1) - O(h)\mu_m].$$

# *Schema Theorem*

- Short, low order, above average schema receive exponentially increasing number of instances in subsequent generations.
- This has been shown to be the optimal compromise between exploration and exploitation in the search space.
- Short, low order, above average schema are called building blocks.
  - groups of closely interacting genes
- The aim in GAs: building block hypothesis
  - GAs work by first discovering “building blocks”
  - then successively combining these (via crossover) to produce larger building blocks until the problem is solved.

# *Markov Chain Analysis of GAs*

- A system is called a Markov Chain if
  - It can exist only in one of a finite number of states
  - So can be described by a variable  $X^t$
  - The probability of being in any state at time  $t+1$  depends only on the state at time  $t$ .
- Has been used to provide convergence proofs
  - The population at a particular time instant is the state of the Markov Chain
- Eiben et al 1989 (almost sure convergence of GAs):
  - IF the space is connected via variation operators AND selection is elitist AND 2 “trivial conditions”
  - THEN

$$P[\text{generation}(n) \text{ contains optimum}] = 1$$

for some  $n$

# Markov Chain Analysis of GAs

*Lemma 1:* Probability of generating any string  $S1$  from a given string  $S2$  is greater than 0, and its value is  $q^{\nu}(1-q)^{l-\nu}$ , where  $\nu$  is the Hamming distance between the two strings and  $q$  is the mutation probability.

*Proof:* Trivial – the distinct positions ( $\nu$ ) must be mutated and the other positions ( $l-\nu$ ) must not be changed.

The states of the chain are denoted by the populations of GAs. A population  $Q$  is a collection of strings of length  $l$  generated over the finite alphabet  $\mathcal{A}$  and defined as follows:

$$Q = \{S_1, S_1, \dots, (\sigma_1 \text{ times}), S_2, S_2, \dots, (\sigma_2 \text{ times}), \dots, S_{\xi}, S_{\xi}, \dots, (\sigma_{\xi} \text{ times});$$
$$S_i \in \mathcal{S}; \sigma_i \geq 1 \text{ for } i=1, 2, \dots, \xi \text{ and } \sum \sigma_i = M\}.$$

Here  $M$  is the population size.

Let  $Q$  denote the set of all populations of size  $M$ .  
 $|Q| = \mathcal{N}$  is the number of states in the Markov chain or, the number of populations. Note that  $\mathcal{N}$  is finite and countable.

*Convergence of GAs:*

$\mathcal{S}$  is the total search space of GAs which contains  $2^l$  strings.

Let the strings have  $s$  distinct fitness function values. Note that  $s \leq |\mathcal{S}| = 2^l$ . Let the  $s$  different fitness values be ordered such that  $F_1 > F_2 > \dots > F_s > 0$ .

Let us classify the strings into a set of  $s$  classes depending on their fitness values. The classes are:

$$\mathcal{S}_i = \{S: \text{fit}(S) = F_i\}$$

where  $F_i$  denotes the  $i$ th highest fitness function value.

Let fitness of a population = maximum fitness in it

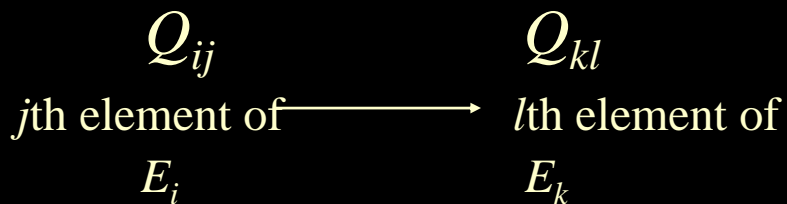
$$fit(Q) = \max_{S \in Q} fit(S).$$

The populations can be partitioned into  $s$  sets,  $E_i$ , such that

$$E_i = \{Q: Q \in \mathcal{Q} \text{ and } fit(Q) = F_i\}$$

i.e., all the populations in a set have the same fitness value. Let  $e_i$  be the number of elements in  $E_i$ .

Let the genetic operators of selection, crossover and mutation generate a population  $Q_{kl}$  from  $Q_{ij}$ . Or,



Transition from  $Q_{ij}$  to  $Q_{kl}$ . Let  $p_{ij.kl}$  denote this transition probability.

Then the probability of transition from  $Q_{ij}$  to any population in  $E_k$ , denoted by  $p_{ij.k}$  can be calculated as:

$$p_{ij.k} = \sum_{l=1}^{e_k} p_{ij.kl} ; \quad j = 1, 2, \dots, e_i ; \quad i, k = 1, 2, \dots, s$$

For all  $j = 1, 2, \dots, e_i$  and  $i = 1, 2, \dots, s$  we get

$$\begin{aligned} p_{ij.k} &> 0 && \text{if } k \leq i \\ &= 0 && \text{if } k > i \quad (\text{by construction and elitist model}) \end{aligned}$$

This indicates that if GA reaches a population  $Q$  in  $E_k$ , it will remain in  $E_k$  for  $k \leq i$ .

Notably, once GA reaches  $E_1$  (the set of best or optimal populations) it will never go out of it.



Let  $p_{ij.kl}^n$  be the probability that GA results in  $Q_{kl}$  in  $n$  steps when  $Q_{ij}$  is the initial state. Let  $p_{ij.k}^n$  denote the probability of reaching one of the populations in  $E_k$  at the  $n$ th step from  $Q_{ij}$ . Then

$$p_{ij.k}^n = \sum_{l=1}^{e_k} p_{ij.kl}^n$$

*Theorem:*

*For an elitist GA with probability of mutation  $q > 0$*

$$\lim_{n \rightarrow \infty} p_{ij.k}^n = 0 \quad \text{for } 2 \leq k \leq s; \quad \forall j=1,2,\dots,e_i, \\ \text{and } i = 1,2,\dots,s.$$

$$\text{Hence } \lim_{n \rightarrow \infty} p_{ij.1} = 1 \quad \forall j=1,2,\dots,e_i, \\ \text{and } i = 1,2,\dots,s.$$

**Proof:** From Lemma 1 we have  $p_{ij.1} > 0$  (since we can go generate any string from any other string with a non zero probability by mutation).

Let 
$$\max_{ij} (1 - p_{ij.1}) = \delta$$

Now

$$\sum_{k \neq 1} p_{ij.k}^{(1)} = \sum_{k=2}^s p_{ij.k} = 1 - p_{ij.1} \leq \delta;$$

$$\sum_{k \neq 1} p_{ij.k}^{(2)} = \sum_{k=2}^s \sum_{i1 \neq 1} \sum_{j1=1}^{e_{i1}} p_{ij.i1j1} p_{i1j1.k} \quad \text{since } p_{lj1.k} = 0 \text{ for } k > 1$$

$$= \sum_{i1 \neq 1} \sum_{j1=1}^{e_{i1}} p_{ij.i1j1} \sum_{k=2}^s p_{i1j1.k}$$

$$= \sum_{i1 \neq 1} \sum_{j1=1}^{e_{i1}} p_{ij.i1j1} (1 - p_{i1j1.1}) \leq \delta \sum_{i1 \neq 1} \sum_{j1=1}^{e_{i1}} p_{ij.i1j1}$$

$$\delta \sum_{i \neq 1} p_{ij.i1} = \delta (1 - p_{ij.1}) \leq \delta^2;$$

Similarly by mathematical induction it can be shown that  $\sum_{k \neq 1} p_{ij.k}^{(n)} \leq \delta^n$  for all  $i, j$ .

Note that since  $0 \leq \delta < 1$ ,  $\delta^n \rightarrow 0$  as  $n \rightarrow \infty$ .

Hence

$\sum_{k \neq 1} p_{ij.k}^{(n)} \rightarrow 0$  as  $n \rightarrow \infty$ . This immediately implies that

$\lim_{n \rightarrow \infty} p_{ij.k}^{(n)} = 0$  for  $2 \leq k \leq s$  for all  $i$  and  $j$ .

Therefore

$$\lim_{n \rightarrow \infty} p_{ij.1}^{(n)} = \lim_{n \rightarrow \infty} (1 - \sum_{k \neq 1} p_{ij.k}^{(n)}) = 1.$$

# *Current Trends in GAs*



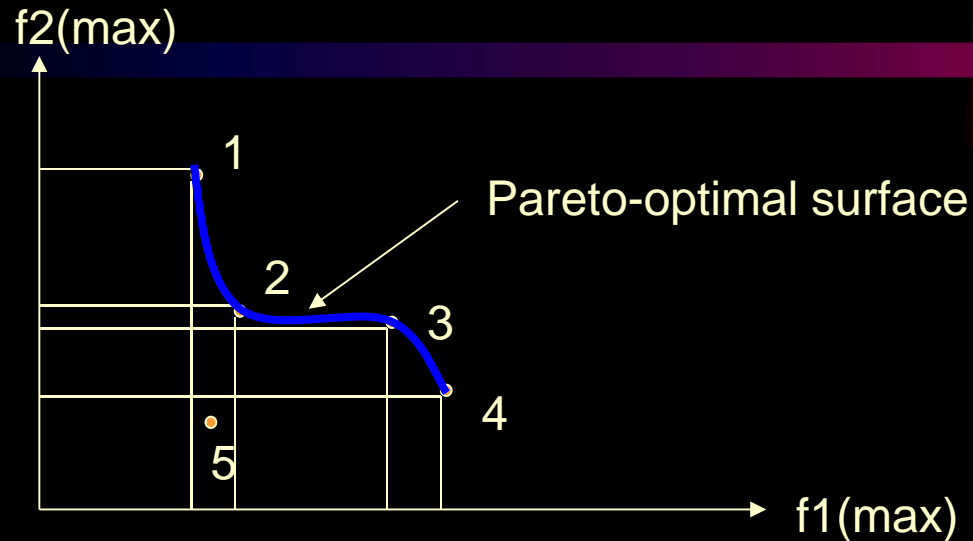
- Encoding strategy
  - Integer encoding
  - Real encoding
  - Encoding of other structures
  - Variable length representation
- Operators
  - New domain specific operators
  - Variable and adaptive probabilities of the operations
  - Chromosome differentiation
- Incorporation of local search
- Handling constraints
  - Reject infeasible strings
  - Penalty based method

# *Current Trends in GAs contd...*



- Multiobjective optimization
  - Multiple conflicting objectives to be simultaneously optimized
  - NSGA-II, PAES, SPEA, AMOSA
- Hybridization with other soft computing tools like
  - Neural networks
  - Fuzzy sets
  - Rough sets

# *Multiobjective SA (AMOSA)*



- **Non dominated solns.**
  - 1, 2, 3 and 4 are
- **Dominated soln**
  - by 2, 3 and 4

Key features:

- Concept of archive in SA
- Amount of domination between two solutions  $\Delta dom$
- Situation specific acceptance probability

■ S. Bandyopadhyay, S. Saha, U. Maulik and K. Deb, "A Simulated Annealing Based Multi-objective Optimization Algorithm: AMOSA", *IEEE Trans. Evolutionary Computation*, 2008.

# Applications

- Clustering

- S. Bandyopadhyay, U. Maulik and A. Mukhopadhyay, “Multiobjective Genetic Clustering for Pixel Classification in Remote Sensing Imagery”, *IEEE Trans. Geoscience & Remote Sensing*, vol. 45, no. 5, pp. 1506-1511, 2007.
- S. Bandyopadhyay and U. Maulik, “Non-parametric Genetic Clustering : Comparison of Validity Indices”, *IEEE Trans. on Systems, Man and Cybernetics Part-C*, vol. 31, no. 1, pp. 120-125, 2001.
- U. Maulik and S. Bandyopadhyay, “Performance Evaluation of Some Clustering Algorithms and Validity Indices”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650-1654, 2002.

- Classification

- S. Bandyopadhyay, S. K Pal and B. Aruna, “Multi-objective GAs, Quantitative Indices and Pattern Classification”, *IEEE Trans. on Systems, Man and Cybernetics - B*, vol. 34, pp. 2088-2099, 2004.

# *Applications*

*contd...*

- **Computational Biology**

- S. Bandyopadhyay, A. Bagchi and U. Maulik, “Active Site Driven Ligand Design: An Evolutionary Approach”, *J. of Bioinformatics and Computational Biology*, vol. 3, No. 5, pp. 1053-1070, 2005.
- S. Bandyopadhyay, “An Efficient Technique for Superfamily Classification of Amino Acid Sequences: Feature Extraction, Fuzzy Clustering and Prototype Selection”, *Fuzzy Sets & Systems*, vol. 152, pp. 5-16, 2005]
- S. S. Ray, S. Bandyopadhyay, and S. K. Pal, “Genetic Operators for Combinatorial Optimization in TSP and Microarray Gene Ordering”, *Applied Intelligence* (accepted).
- S. Bandyopadhyay, A. Mukhopadhyay and U. Maulik, “An Improved Algorithm for Clustering Gene Expression Data”, *Bioinformatics*, Oxford University Press, vol. 23, no. 21, pp. 2859-2865, 2007.



# General References

- S. Bandyopadhyay and S. K. Pal, Classification and Learning Using Genetic Algorithms, Applications in Bioinformatics and Web Intelligence, Springer-Verlag, 2007
- D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York, 1989.
- Michalewicz, Z. 1996. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag. Third edition.
- Mitchell, Melanie. 1996. *An Introduction to Genetic Algorithms*. Cambridge, MA
- Beyer, Hans-Georg. 2001. *The Theory of Evolution Strategies*. Heidelberg: Springer-Verlag.
- Schwefel, Hans-Paul. 1995. *Evolution and Optimum Seeking*. New York, NY: John Wiley.
- Fogel, David B. 1991. *System Identification through Simulated Evolution*. Needham Heights, MA: Ginn Press.
- Kalyanmoy Deb. Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, UK, 2001, ISBN 0-471-87339-X.
- Carlos A. Coello Coello, David A. Van Veldhuizen and Gary B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, New York, March 2002, ISBN 0-3064-6762-3.

# References

- S. Bandyopadhyay, S. K Pal, and B. Aruna, "Multi-objective GAs, quantitative Indices and Pattern Classification", *IEEE Transactions on Systems, Man and Cybernetics - B*, vol. 34, no. 5, pp. 2088-2099, 2004.
- U. Maulik and S. Bandyopadhyay, "Performance Evaluation of Some Clustering Algorithms and Validity Indices", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650-1654, 2002.
- U. Maulik and S. Bandyopadhyay, "Fuzzy Partitioning Using Real Coded Variable Length Genetic Algorithm for Pixel Classification", *IEEE Transactions on Geosciences and Remote Sensing*, vol. 41, no. 5, pp. 1075-1081, 2003.
- S. Bandyopadhyay, "Simulated Annealing Using Reversible Jump Markov Chain Monte Carlo Algorithm for Fuzzy Clustering", *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 479-490, 2005.
- S. Bandyopadhyay, "An Efficient Technique for Superfamily Classification of Amino Acid Sequences: Feature Extraction, Fuzzy Clustering and Prototype Selection", *Fuzzy Sets and Systems*, vol. 152, pp. 5-16, 2005.
- S. Bandyopadhyay, Ac. Bagchi and U. Maulik, "Active Site Driven Ligand Design: An Evolutionary Approach", *Journal of Bioinformatics and Computational Biology*, vol. 3, No. 5, pp.1053-1070, 2005.

# References

- M. K. Pakhira, S. Bandyopadhyay and U. Maulik, ``A Study of Some Fuzzy Cluster Validity Indices, Genetic Clustering and Application to Pixel Classification", *Fuzzy Sets and Systems*, vol. 155, pp. 191-214, 2005.
- S. Bandyopadhyay, U. Maulik and A. Mukhopadhyay, ``Multiobjective Genetic Clustering for Pixel Classification in Remote Sensing Imagery", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1506-1511, 2007.
- S. S. Ray, S. Bandyopadhyay, and S. K. Pal, ``Genetic Operators for Combinatorial Optimization in TSP and Microarray Gene Ordering", *Applied Intelligence*, vol. 26, no. 3, pp. 183-195, 2007.
- S. Bandyopadhyay and S. Saha, ``GAPS: A New Symmetry Based Genetic Clustering Technique", *Pattern Recognition*, vol. 10, no. 12, pp. 3430-3451, 2007.
- S. Bandyopadhyay, S. Saha, U. Maulik and K. Deb, ``A Simulated Annealing Based Multi-objective Optimization Algorithm: AMOSA", *IEEE Transaction on Evolutionary Computation* (accepted).
- S. Bandyopadhyay and S. Santra, "A Genetic Approach for Efficient Outlier Detection in Projected Space", *Pattern Recognition* (accepted).
- S. Bandyopadhyay, A. Mukhopadhyay and U. Maulik, ``An Improved Algorithm for Clustering Gene Expression Data", *Bioinformatics*, Oxford University Press, vol. 23, no. 21, pp. 2859-2865, 2007.



**Thank you!!**