CS 235: Artificial Intelligence

Local Search Algorithm
Part-II

Dr. Moumita Roy
CSE Dept., IIITG

Local Beam Search

- Keep track of k states rather than just one, as in hill climbing
- K is the beam size.

Local Beam Search

- Begins with k randomly generated states
- At each step, all successors of all k states are generated
- If any one is a goal, algorithm halts
- Otherwise, selects best k successors from the complete list, and repeats

Local Beam Search

- Successors can become concentrated in a small part of state space if less diversity in k successors (like hill climbing with high cost)

- Stochastic beam search: choose k successors at random with a probability based on their goodness

- Like natural selection: successors (offspring) of a state (organism) populate the next generation according to its value (fitness)

Complete state-formulation (Clustering)

Formulate the following problem for intelligent agent:

Divide the N students in k groups based on their marks in AI such as diversity in each group is minimized.

For demonstration, you may use:

N=5, k=2, set of marks={50, 20,10, 5, 15, 45} (out of 60)

- State representation:
- Initial state:
- Goal state:
- Successor function:
- Objective function:

Complete state-formulation (Version 1) (Clustering)

Set of students/marks $A=\{a_1, a_2, a_3,\ldots\ldots,a_N\}= =\{50, 20,10, 5, 15, 45\}$

- State representation:

  $S=[C_1\ C_2\ C_3\ \ldots\ldots\ C_N]$

  if a_j belongs to i^{th} group, then $C_j=i$, where $i \in \{1, 2, 3,\ldots.,k\}$

- Initial state: (example for demonstration)

  $S^0=[1\ 2\ 2\ 2\ 2\ 1]$ $\rightarrow$ better representation is needed

  $F(S^0)=(50-45)^2 + [(20-10)^2+(20-5)^2+(20-15)^2+(10-5)^2+(10-15)^2+(5-15)^2]$

- Goal state/test:

  No binary goal state

  report the state when we have the minimum value of objective function
- Successor function: Successor is generated by changing the group of one student at a time
- Objective function: summation of diversity in each cluster

Complete state-formulation (Version 2) (Clustering)

Set of students/marks $A=\{a_1, a_2, a_3,\ldots\ldots,a_N\}= =\{50, 20,10, 5, 15, 45\}$

- State representation:

  $S=[C_1\ C_2\ C_3\ \ldots\ldots\ C_k]$

  if C_j is representative of the jth group, then $C_j \in \{a_1, a_2, a_3,\ldots,a_N\}$

- Initial state: (example for demonstration)

  $S^0=[50\ 20]$ (assigned in a group with nearest representative; then same as [1 2 2 2 2 1])

  $F(S^0)=[(50-50)^2+(50-45)^2]+[(20-20)^2+(20-10)^2+(20-5)^2+(20-15)^2]$

  $$F(S^0)=(50-45)^2 + [(20-10)^2+(20-5)^2+(20-15)^2+(10-5)^2+(10-15)^2+(5-15)^2]$$

- Goal state/test:

  No binary goal state

  report the state when we have the minimum value of objective function

- Successor function: Successor is generated by changing a group representative at a time
- Objective function: summation of diversity in each cluster

Observation

- We need some strategy to generate successors

- State space may be continuous or discrete

Genetic Algorithms

- Variant of stochastic beam search
- Combine two parent states to generate successors

Natural Evolution Process

- GA mimics the natural evolution process.
- **Evolution:** Evolution is the process by which modern organisms have descended from ancient ones
- **Microevolution** is evolution within a single population; (a population is a group of organisms that share the same gene pool).
- Components:

➢ Heredity

Information needs to be passed on from one generation to the next

➢ Genetic Variation

There has to be differences in the characteristics of individuals in order for change to occur (mutation)

➢ Differential Reproduction

Some individuals need to (get to) reproduce more than others thereby increasing the frequency of their genes in the next generation; often the term fitness is used to describe the relative ability of individuals to pass on their genes

Start with MAXONE problem

- Formulate the problem for Intelligent agent:

  Task is to maximize the number of 1' s in a string of L binary digits.

- Is it a very simple one? We know the answer beforehand.

- However, we can think of it as maximizing the number of correct answers, each encoded by 1, to L yes/no difficult questions.

Complete state-formulation (MAXONE)

- State representation:
  $S=[C_1\ C_2\ C_3\ \ldots\ldots\ C_L]$
  C_j = 0 or 1

- Initial state: (example for demonstration; L=6)
  $S^0=[1\ 0\ 1\ 0\ 1\ 0]$ / Encoding
  $F(S^0)=3$

- Goal state/test:
  report the state when we have the maximum value of objective function

- Successor function: Successor is generated by changing one element at a time

- Objective function: number of 1's in a string (Fitness values)

Basic steps of GA

Step 1: **Encoding** of the problem (state representation)

Step 2: Random generation of Initial population (set of initial states)

Step 3: Compute fitness of each individual of the current population (objective function of each state/a possible solution)

Step 4: Select the pairs of parents from the current population based on fitness value **(Selection)**

Step 5: Generate the offspring (successor) using **Crossover and Mutation**

Step 6: Repeat step 3 to 6 until satisfying solution is reached or convergence is reached

Encoding/Selection/Crossover/ Mutation

Solve the MAXONE problem using GA

Encoding

- State/Solution is represented by Chromosome
- Each element of state is represented by gene
  (here each 0/1)

0	0	1	0	1	1	1	0	0	1

- **Genes** are joined into a string to form Chromosome (solution)
- Representation should easy to apply mutation and crossover

➢ **Binary representation**: same as above (binary string)

➢ **Real valued representation**: For problems where we want to define the genes using continuous rather than discrete variables, the real valued representation is the most natural.

0.5	0.2	0.6	0.8	0.7	0.4	0.3	0.2	0.1	0.9

➢ **Integer Representation**: For discrete valued genes, we cannot always limit the solution space to binary 'yes' or 'no'. For example, if we want to encode the clustering with 4 groups({1,2,3,4}). In such cases, integer representation is desirable.

1	2	3	4	3	2	4	1	2	1

➢ **Character Representation:** also possible

➢ **Permutation Representation:** In many problems, the solution is represented by an order of elements. In such cases permutation representation is the most suited. A classic example of this representation is the travelling salesman problem .

MAXONE Problem using GA

Step 1: **Encoding** of the problem (state representation)

Here, we use binary representation

Step 2: Random generation of Initial population (set of initial states)

Step 3: Compute fitness of each individual of the current population (objective function of each state/a possible solution)

Step 4: Select the pairs of parents from the current population based on fitness value (Selection)

Random initialization (Population size: 4/L=6)

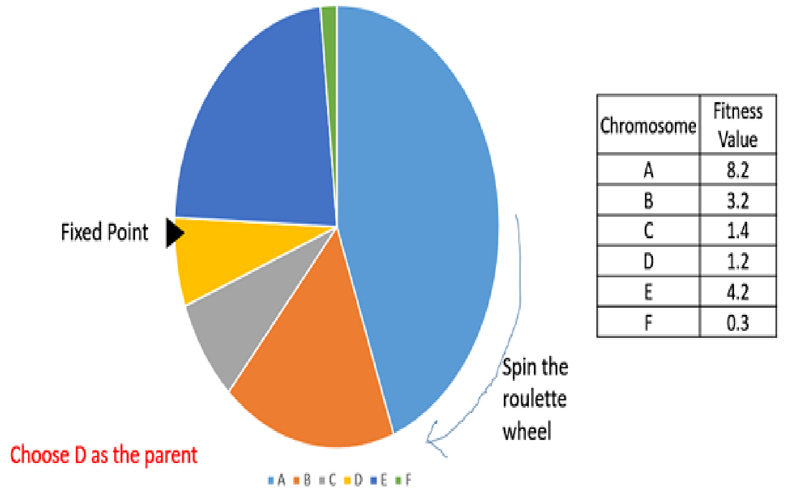
Initial Population	Fitness value
(Chromosome 1) [0 1 0 1 1 1]	4
(Chromosome 2) [1 1 0 1 1 1]	5
(Chromosome 3) [1 0 1 0 1 0]	3
(Chromosome 4) [1 0 0 0 0 1]	2

Selection Operator (survival of the fittest)

- In this case, every individual can become a parent with a probability which is proportional to its fitness.

➢ **Roulette Wheel Selection:**

1. Consider a circular wheel.
2. The wheel is divided into n pies, where n is the number of individuals in the population.
3. Each individual gets a portion of the circle which is proportional to its fitness value.
4. A fixed point is chosen on the wheel circumference as shown and the wheel is rotated.
5. The region of the wheel which comes in front of the fixed point is chosen as the parent.
6. For the second parent, the same process is repeated.



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

Problem: If initial population contains one or two very fit but not best individual and the rest of the members are not so good, then these fit individuals dominate others; prevent to explore the population. (premature/fast convergence)

Selection Operator

- Rank selection:

1. every individual in the population is ranked according to their fitness.
2. The selection of the parents depends on the rank of each individual and not the fitness.
3. The higher ranked individuals are preferred more than the lower ranked ones.

Initial Population	Fitness value	Portion in wheel	Rank	Portion in wheel
A	40	80%	4	40%
B	5	10%	3	30%
C	3	6%	2	20%
D	2	4%	1	10%
Fitness of the population	50		10	

Problem: Slower convergence/ Can't differ best chromosome from others

Selection Operator

- **Tournament selection:**

1. In k-way tournament selection, select k individuals from the population at random and select the best out of these to become a parent.
2. The same process is repeated to select the next parent.

- **Random selection:** We randomly select parents from the existing population. There is no selection pressure towards fitter individuals and therefore this strategy is usually avoided.

- **Elitism:**

1. Copy the best chromosome (solution) to the new population
2. Intuition: creating a new population (set of successors) using crossover and mutation; may lost the best one.
3. Retain some number of best individual at each generation.

MAXONE Problem using GA

Step 1: **Encoding** of the problem (state representation)

Here, we use binary representation

Step 2: Random generation of Initial population (set of initial states)

Step 3: Compute fitness of each individual of the current population (objective function of each state/a possible solution)

Step 4: Select the pairs of parents from the current population based on fitness value (Selection)

Random initialization (Population size: 4/L=6)

Initial Population	Fitness value
(Chromosome 1) [0 1 0 1 1 1]	4
(Chromosome 2) [1 1 0 1 1 1]	5
(Chromosome 3) [1 0 1 0 1 0]	3
(Chromosome 4) [1 0 0 0 0 1]	2

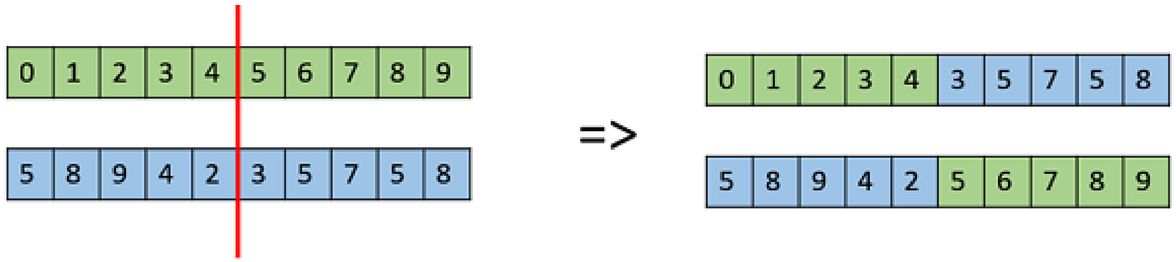
Select Chromosome 1 as parent 1 and Chromosome 3 as parent 2

Step 5: Generate the offspring (successor) using **Crossover and Mutation**

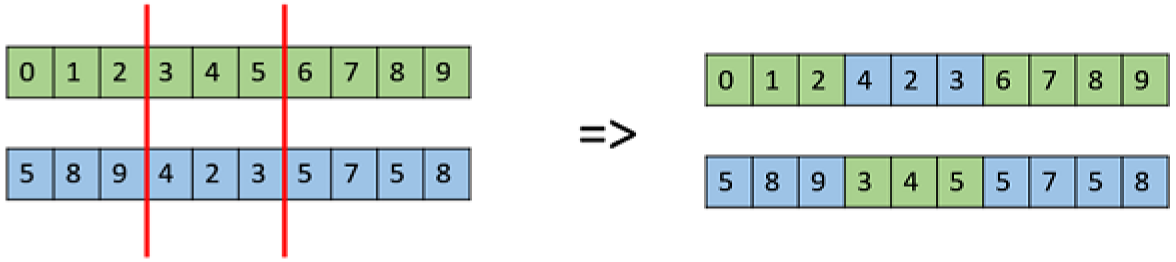
Generation of successors (crossover operators)

- We need to generate the same number of successor as the population size.
- It means we need to repeat the step 4 (selection) until the desired number of successors are generated.

➢ **Single-point crossover**: In this one-point crossover, a random crossover point is selected and the tails of its two parents are swapped to get new off-springs.

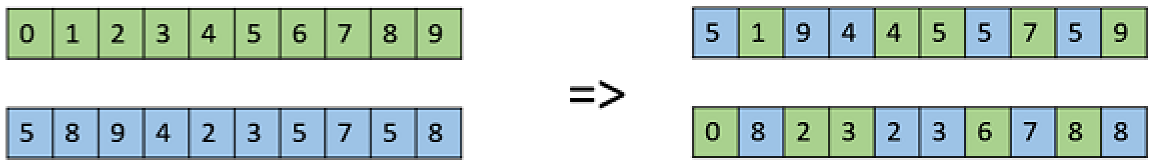


➢ **Multi Point Crossover:** Multi point crossover is a generalization of the one-point crossover wherein alternating segments are swapped to get new off-springs.



Generation of successors (crossover operators)

- Uniform Crossover
- In a uniform crossover, we don't divide the chromosome into segments, rather we treat each gene separately.
- In this scheme, at each bit position (gene) of the parent, we toss a coin to determine whether there will be swap of the bits or not.
- Each gene is chosen from either parent with equal probability

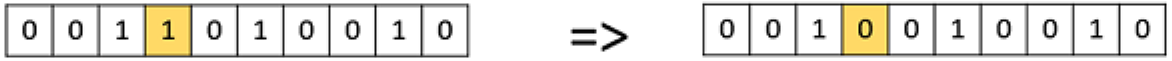


- Crossover is performed on two parents to form two new offspring.
- The GA has a crossover probability (in the range of [0, 1]) that determines if crossover will happen. A randomly generated floating-point value is compared to the crossover probability, and if it is less than the probability, crossover is performed; otherwise, the offspring are identical to the parents.
- If crossover is to occur, one or more crossover points are generated, which determines the position in the chromosomes where parents exchange genes.

Mutation Operators

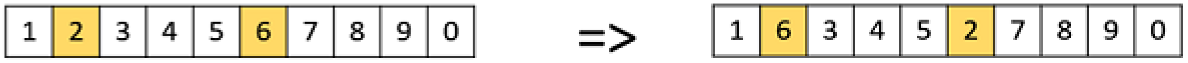
- Mutation is a genetic operator used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next.

➢ **Bit Flip Mutation**: select one or more random bits and flip them. This is used for binary encoded GAs.



➢ **Random Resetting**: the extension of the bit flip for the integer representation. In this, a random value from the set of permissible values is assigned to a randomly chosen gene.

➢ **Swap Mutation**: select two positions on the chromosome at random, and interchange the values.



Mutation Probability

- After the offspring are generated from the selection and crossover, the offspring chromosomes may be mutated.
- Like crossover, there is a mutation probability (in the range of [0, 1]).
- If a randomly selected floating-point value is less than the mutation probability, mutation is performed on the offspring; otherwise, no mutation occurs.

Parameters in GA

- Population size: too few, search space not explored; too large slow down GA
- Crossover Probability/Crossover Points
- Mutation Probability

MAXONE Problem using GA

Step 1: **Encoding** of the problem (state representation)

Here, we use binary representation

Step 2: Random generation of Initial population (set of initial states)

Step 3: Compute fitness of each individual of the current population (objective function of each state/a possible solution)

Step 4: Select the pairs of parents from the current population based on fitness value (Selection)

Random initialization (Population size: 4/L=6)

Initial Population	Fitness value
(Chromosome 1) [0 1 0 1 1 1]	4
(Chromosome 2) [1 1 0 1 1 1]	5
(Chromosome 3) [1 0 1 0 1 0]	3
(Chromosome 4) [1 0 0 0 0 1]	2

Select Chromosome 1 as parent 1 and Chromosome 3 as parent 2

Step 5: Generate the offspring (successor) using **Crossover and Mutation**

MAXONE Problem using GA

Step 5: Generate the offspring (successor) using Crossover and Mutation

Crossover:

Selected Individuals	Successors (Crossover)
(Chromosome 1) [0 1 0 1 1 1]	[0 1 0 0 1 0]
(Chromosome 3) [1 0 1 0 1 0]	[1 0 1 1 1 1]

By this way, generate the individuals in the next generation (single-point crossover/crossover probability):

Current Population	Next Population (after crossover)
(Chromosome 1) [0 1 0 1 1 1]	[0 1 0 0 1 0]
(Chromosome 2) [1 1 0 1 1 1]	[1 0 1 1 1 1]
(Chromosome 3) [1 0 1 0 1 0]	[1 1 0 0 1 0]
(Chromosome 4) [1 0 0 0 0 1]	[1 0 1 1 1 1] (Repetition)

MAXONE Problem using GA

Step 5: Generate the offspring (successor) using Crossover and Mutation

Mutation (mutation probability)

Current Population	Next Population (after crossover)	Next Population (after crossover and mutation)
[0 1 0 1 1 1]	[0 1 0 0 1 0]	[0 1 0 0 1 0]
[1 1 0 1 1 1]	[1 0 1 1 1 1]	[1 0 0 1 1 1]
[1 0 1 0 1 0]	[1 1 0 0 1 0]	[1 1 0 0 0 0]
[1 0 0 0 0 1]	[1 0 1 1 1 1]	[1 0 1 0 1 1]

Step 6: Continue till convergence (number of generations, no change in the fitness between populations, reach the goal state (if knows))

When all the individuals in a generation is identical???

Note: The next states are generated without storing the all the successors of k current states

Observation

- GA discussed so far is applicable for the discrete state space mainly due to crossover operator (single point/multi-point/uniform)

- The suitable encoding policy is binary. In case of Integer representation, it is generally encoded into binary; same for other representation also.

- We need a crossover operator, suitable for Real-valued representation and work comfortably in continuous state-space.

- The continuous genetic algorithm is faster than the binary genetic algorithm because the chromosomes do not need to be decoded before testing their fitness with the fitness function.

- Example:

  Apply GA to maximize the following function:

  $f(x)=2x^2+1$; where $0 <= x =< 6$

Real-valued GA (Arithmetic Crossover)

- $Ch1=[X_m\ Y_m]$ and $Ch2=[X_d\ \ Y_d]$
- we first randomly select one of the parameters to be the point of crossover
- Suppose, *X* is the crossover point for a particular *Ch 1 and Ch2*
- Then we introduce a random value between 0 and 1 represented by *β* and the *X*-values in the offspring are

$$X_{new1} = (1 - \beta)X_m + \beta X_d$$
$$X_{new2} = (1 - \beta)X_d + \beta X_m$$

- The remaining parameter (*y* in this case) is inherited directly from each parent, so the completed offspring are

  offspring1 = $[X_{new1}\ \ Y_m]$
  offspring2 = $[X_{new2}\ Y_d]$

Complete state-formulation (Clustering)

Formulate the following problem for intelligent agent:

Divide the N students in k groups based on their marks in AI such as diversity in each group is minimized.

For demonstration, you may use:

N=5, k=2, set of marks={50, 20,10, 5, 15, 45} (out of 60)

Complete state-formulation (Version 2) (Clustering)

Set of students/marks $A=\{a_1, a_2, a_3,\ldots\ldots,a_N\}=\{50, 20,10, 5, 15, 45\}$

- State representation:

  $S=[C_1\ C_2\ C_3\ \ldots\ldots\ C_k]$

  if C_j is representative of the jth group, then $C_j \in \{a_1, a_2, a_3,\ldots,a_N\}$

- Initial state: (example for demonstration)

  $S^0=[50\ 20]$ (assigned in a group with nearest representative; then same as [1 2 2 2 2 1])

  Group representation is always someone from group

  $F(S^0)=[(50-50)^2+(50-45)^2]+[(20-20)^2+(20-10)^2+(20-5)^2+(20-15)^2]$

- Goal state/test:

  No binary goal state

  report the state when we have the minimum value of objective function

- Successor function: Successor is generated by changing a group representative at a time
- Objective function: summation of diversity in each cluster

Clustering using GA (Design 1)

- Encoding: Binary(encode integer into binary; It may lead to infeasible solution as per our present formulation)/Integer

- Parameters: Population size
  crossover probability
  crossover point (randomly)
  mutation probability
  number of generations

- Strategy for selection: Roulette wheel selection/rank selection/tournament selection/Elitism

- Type of crossover: Single-point/multi-point/Uniform-crossover

- Type of mutation: Bit flip/Random setting/Swap mutation

Clustering using GA (Design 1)

- Using integer encoding
- Restricts our search space (random resetting in mutation can be a partial solution)

Current Population	Selection/crossover (3rd and 4th chromosomes)
[50 20]	
[10 5]	
[50 5]	[50 45]
[20 45]	[20 5]

Observation

- As per the present formulation, the group representative is always someone from the group (from set $A=\{a_1, a_2, a_3,\ldots\ldots,a_N\}= =\{50, 20,10, 5, 15, 45\}$)

- We are searching for a better representative, not compulsorily someone from the set of students; The representative is necessary to divide the students in a group that minimize the diversity.

- We can relax the constraint that the group representative are from the set of students (example, in K-means clustering algorithm; the representative of i^{th} group is updated by the mean value of the students, assigned in the i^{th} group)

S=[Representative of first cluster Representative of second cluster]
=[50 20]

- In this regard, we may re-design the GA (encoding and crossover)

Clustering using GA (Design 2)

- Encoding: Real-valued
- Parameters: Population size
  crossover probability
  crossover point (randomly)
  mutation probability
  number of generations

- Strategy for selection: Roulette wheel selection/rank selection/tournament selection/Elitism

- Type of crossover: Arithmetic crossover

- Type of mutation: Random setting the value from a range of permissible value

Real-valued GA (Arithmetic Crossover)

- Ch1=[X_m Y_m] =[50 5] and Ch2 =[X_d Y_d]=[20 45]
- $X_{new1} = (1 - \beta)X_m + \beta X_d$ =(1-0.2)*50+0.2*20
  $X_{new2} = (1 - \beta)X_d + \beta X_m$ =(1-0.2)*20+0.2*50
- offspring1 = [Xnew1 Ym]
  offspring2 = [Xnew2 Yd]

Current Population	Selection/crossover (3rd and 4th chromosomes)
[50 20]	
[10 5]	
[50 5]	[44 5]
[20 45]	[26 45]

Swarm Intelligence

- **Swarm**: Grouping of similar animals, generally moving in the same direction

- Ants swarm to build colonies
- Birds swarm to find food
- Bees swarm to reproduce

Swarm Intelligence

- Inspired from the nature social behavior and dynamic movements with communications of insects, birds and fish
- **Swarm can achieve things that an individual cannot**

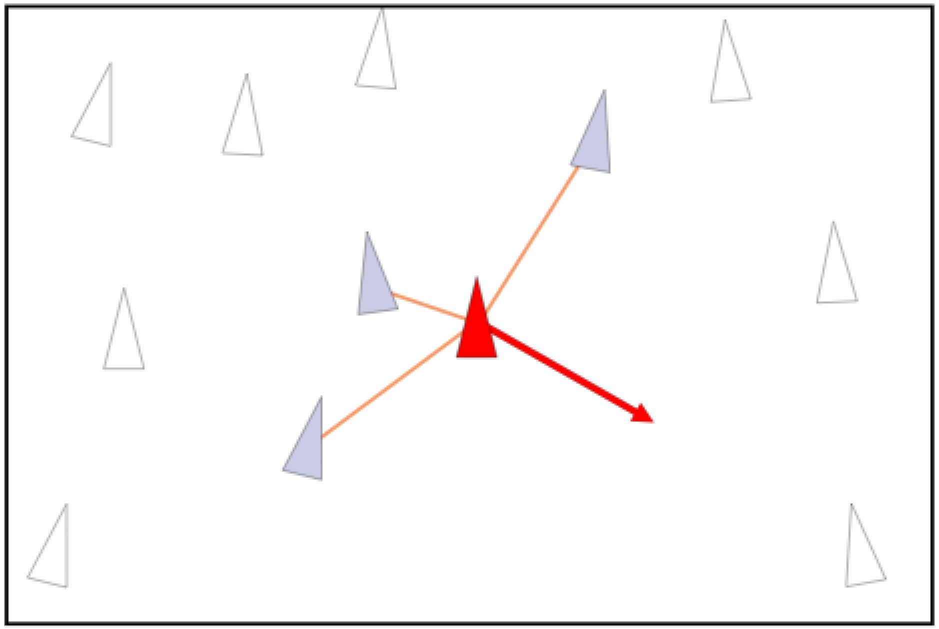
Swarm Intelligence

- Inspiration from swarm intelligence has led to some highly successful optimisation algorithms.
- **Particle Swarm Optimisation (PSO):** a different way to solve optimisation problems, based on the swarming behaviour of several kinds of organisms.
- **Ant Colony (-based) Optimisation:** a way to solve optimisation problems based on the way that ants indirectly communicate directions to each other.

Only using three rules

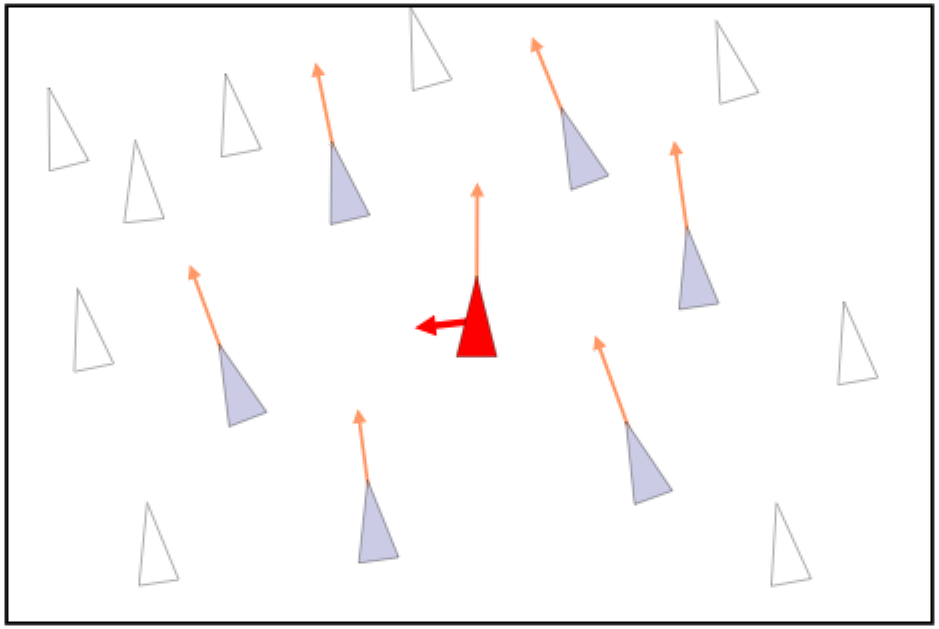
Collision Avoidance

- Rule 1: Avoid Collision with neighboring birds



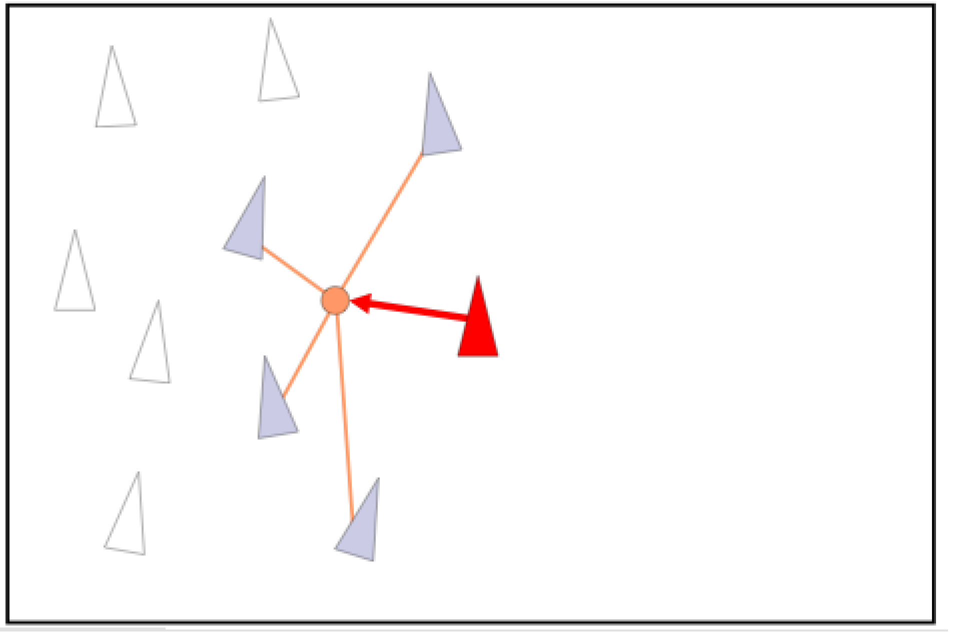
Velocity Matching

- Rule 2: Match the velocity of neighboring birds



Flock Centering

- Rule 3: Stay near neighboring birds

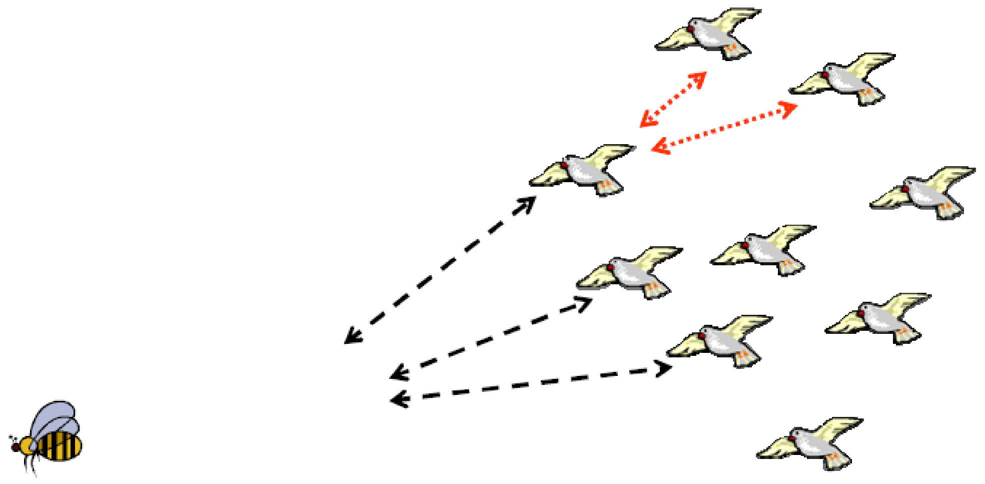


Swarming - Characteristics

- Simple rules for each individual
- No central control
  - Decentralized and hence robust
- Emergent
  - Performs complex functions

Particle Swarm Optimization

- Particle swarm optimization imitates human or insects social behavior.
- Individuals interact with one another while learning from their own experience, and gradually move towards the goal.
- It is easily implemented and has proven both very effective and quick when applied to a diverse set of optimization problems.
- Bird flocking is one of the best example of PSO in nature.



PSO

- Each particle (a possible solution/state) in a swarm evaluates the function to maximize at each point it visits in spaces.

- Each particle has a fitness value/objective value and is searching for the optimum.

- Each particle is moving and has a velocity.

- Each particle has two components: position and velocity.
- In each time-step, a particle has to move to a new position by adjusting its velocity.

- **Local behavior:** Each particle remembers the best value of the function found so far by it (its personal best, pbest) and its position.

- Local behavior is not sufficient; particles in swarm needs to cooperate; They exchange the information.
- **Social behavior:** Each particle knows the globally best position that one member in a swarm had found, and its value (gbest).

Algorithm of PSO

- Each particle modifies its position according to:
  - its current position
  - its current velocity
  - the distance between its current position and position corresponding to pbest
  - the distance between its current position and position corresponding to gbest

To sum up,

- Change position by adjusting the velocity
- Change velocity based on its past experience and feedback received from its neighbors.

Algorithm of PSO

Step1: Randomly initialize particles (position and velocity) in swarm (set of states)

Step2: Compute fitness of each particle along with corresponding pbest and gbest

Step3: Update velocity of each particle using its current velocity (inertia effect), local/personal influence, global/neighboring/social influence

Step 4: Update position of each particle using updated velocity

Step 5: Repeat Steps 2-5 until convergence (the number of maximum iterations, no change in position/objective value of the particles in swarm)

Update Velocity and Position of particle in a swarm

- Each ith particle has velocity V_i^t and position S_i^t at each tth time-stamp/iteration

- **Velocity Update:**
  Using its current own position and the positions corresponding to pbest (PB_i^t) and gbest (GB^t)

$$V_i^{t+1} = V_i^t + C_1 * rand() * (PB_i^t - S_i^t) + C_2 * rand() * (GB^t - S_i^t)$$

Three components: Inertia, personal influence, social influence
rand():random number between 0 and 1
C1:weight for personal influence
C2:weight for social influence
Usually C1 + C2 = 4 (empirically chosen value)

- **Position Update:**
  Using updated velocity and its own current position, each particle updates its position as:

$$S_i^{t+1} = S_i^t + V_i^{t+1}$$

Comparison with GA

- No operators like GA
- Every particle needs to store its own best position and global best position; whereas no such extra memory is needed for GA
- Information sharing is only by crossover in GA ; by gbest in PSO

Complete state-formulation (Clustering)

Formulate the following problem for intelligent agent:

Divide the N students in k groups based on their marks in AI such as diversity in each group is minimized.

For demonstration, you may use:

N=5, k=2, set of marks={50, 20,10, 5, 15, 45} (out of 60)

Complete state-formulation (Version 2) (Clustering)

Set of students/marks $A=\{a_1, a_2, a_3,\ldots\ldots,a_N\}=\{50, 20,10, 5, 15, 45\}$

- State representation:

  $S=[C_1\ C_2\ C_3\ \ldots\ldots\ C_k]$

  if C_j is representative of the jth group, then $C_j \in \{a_1, a_2, a_3,\ldots,a_N\}$

- Initial state: (example for demonstration)

  $S^0=[50\ 20]$ (assigned in a group with nearest representative; then same as [1 2 2 2 2 1])

  $F(S^0)=[(50-50)^2+(50-45)^2]+[(20-20)^2+(20-10)^2+(20-5)^2+(20-15)^2]$

- Goal state/test:

  No binary goal state

  report the state when we have the minimum value of objective function

- Successor function: Successor is generated by changing a group representative at a time
- Objective function: summation of diversity in each cluster

PSO for clustering

Step1: Randomly initialize particles (position and velocity) in swarm (set of states)

Step2: Compute objective value of each particle along with corresponding pbest and gbest

Number of particles: 3

Iteration No: 1 (Initialization)

A={50, 20,10, 5, 15, 45}

Particles (position)	Particles (velocity)	Objective value	Position/Pbest	Position/Gbest
[50 20]	[0 0]	375	[50 20]/375	[50 20]/375
[20 5]	[0 0]	1575	[20 5]/1575	[50 20]/375
[15 20]	[0 0]	1650	[15 20]/1650	[50 20]/375

PSO for clustering (update velocity and position)

Iteration No: 2

Particle 1: (C1=2,C2=2,rand()=1)

$$V_1^2 = V_1^1 + C_1 * rand() * (PB_1^1 - S_1^1) + C_2 * rand() * (GB^1 - S_1^1)$$

$$S_1^2 = S_1^1 + V_1^2$$

V_1^2 =[0 0]+2*1*([50 20]-[50 20])+2*1*([50 20]-[50 20])
=[0 0]

S_1^2 =[50 20]+[0 0]=[50 20]

PSO for clustering (update velocity and position)

Iteration No: 2

Particle 2:

V_2^2 =[0 0]+2*1*([20 5]-[20 5])+2*1*([50 20]-[20 5])
=[60 30]

S_2^2 =[20 5]+[60 30]=[80 35]

PSO for clustering (update velocity and position)

Iteration No: 2

Particle 3:

V_2^2 =[0 0]+2*1*([15 20]-[15 20])+2*1*([50 20]-[15 20])
=[70 0]

S_2^2 =[15 20]+[70 0]=[85 20]

PSO for clustering (update velocity and position)

Iteration No: 2

Particles (Updated position)	Particles (Updated velocity)	Objective value	Position/Pbest	Position/Gbest
[50 20]	[0 0]	375	[50 20]/375	[50 20]/375
[80 35]	[60 30]	1575	[20 5]/1575	[50 20]/375
[85 20]	[70 0]	1650	[15 20]/1650	[50 20]/375

What next?

- Re-calculate the objective function for each particle corresponding to the current position
- Update Pbest(position) if required
- Update Gbest(position) if required
- Repeat the same process until convergence

Important points

- Particle velocity on each dimension is clamped to a maximum velocity Vmax. If the sum of accelerations would cause the velocity on that dimension to exceed Vmax, which is a parameter specified by the user. Then the velocity on that dimension is limited to Vmax.

- Instead of Gbest, we may consider the local best also (lbest); best by considering the topological neighbors of any particle.