IO-INSPIRED ALGORITHMS

”From Natural Systems to Computational Solution“

Lama Ayash

Optimization

What’s optimization?

It aims to minimize costs and energy or maximize profit, performance, and efficiency.

Why optimization?

Because resources like time, money, and materials are limited, so we must use them effectively under constraints.

What is an Algorithm?

An algorithm is a step-by-step procedure for providing calculations or instructions.

- Many algorithms are iterative.
- Algorithms for optimization have more emphasis on iterative procedures for constructing algorithms.

Optimization

Optimization algorithm

Deterministic algorithms

Hill climbing always follows the same route if started from the same point

Stochastic Algorithms

Genetic algorithms generate different solution populations each time due to random elements.

Hybrid Algorithms

Hill climbing with random restart uses deterministic search but restarts randomly to avoid local optima.

Optimization

Stochastic Algorithms

Heuristic algorithms

Use trial and error to find solutions. Simple and problem-specific.

Metaheuristic algorithms

Advanced, “higher-level” heuristics designed to perform better than simple heuristics.
Combine randomization and local search to balance exploration and exploitation.

Optimization

Metaheuristic algorithms

Population-based

Use multiple agents or solutions.
Examples: Genetic Algorithms, Particle Swarm Optimization (PSO), Firefly Algorithm (FA), Cuckoo Search.

Trajectory-based:

Use a single solution that moves step-by-step through the search space.
Example: Simulated Annealing — accepts better moves, and sometimes worse ones with a certain probability to escape local optima.

GENETIC ALGORITHMS

Inspiration

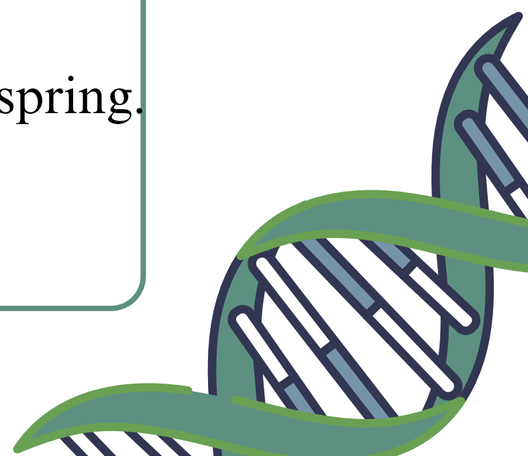
Genetic Algorithms

1-Darwin's Theory of Evolution (1859)

- Proposed by Charles Darwin in On the Origin of Species.
- **Core idea:** Species evolve over generations through natural selection.
- **Natural Selection:** Individuals with traits better suited to the environment are more likely to survive and reproduce.
- Over time, advantageous traits become more common in the population.

2. Key Biological Concepts Behind Evolution

- **Population**: A Group of individuals.
- **Chromosomes**: Structures carrying genetic information.
- **Genes**: Segments of DNA coding for traits.
- **Crossover** (Recombination): Mixing genetic material from parents to form offspring.
- **Mutation**: Random changes in genes, introducing diversity.
- **Selection**: Fittest individuals have higher reproduction probability.



Genetic Algorithms

Genetic Algorithms :

- Developed by John Holland (1960s–1970s), inspired by Darwin's natural selection.
- Uses crossover, mutation, and selection for adaptive systems.

Genetic Algorithm Workflow

1. One iteration = a Generation
   - Population of fixed-length strings (often binary).
   - Each generation applies three genetic operators:
     a. Selection
     b. Crossover
     c. Mutation

Genetic Algorithms

1 Crossover

- Swap chromosome segments between parents.
- Single-point or multi-point crossover.
- Mix solutions converge in a subspace.

Works in a limited subspace; identical parents produce identical offspring.

2 Mutation

- Randomly flip bits in a chromosome.
- Single-site or multi-site mutation.
- Increases diversity, escapes local optima.

Can generate far-away solutions, altering the search path.

3 Selection

- Choose individuals based on fitness.
- **Methods:**
- Roulette wheel
- Fitness-proportional
- Ranking
- Tournament
- Role: Keep best solutions → drives evolution (**elitism**).

Genetic Algorithms

Exercise

1 Initialization

- Generate an initial population of candidate solutions (chromosomes).
  - A1 = 01100
  - A2 = 11001
  - A3 = 00101
  - A4 = 10011
  - Each chromosome is a string of bits; each bit is called a gene.

2 Fitness Assignment

- Calculate fitness for each chromosome based on the fitness function.
- Example: **f(x)=x2**, where x is the decimal value of the chromosome.

3 Goal

Achieve fitness ≥ 800

Genetic Algorithms

Exercise

1-Initial Population

String No	Chromosome	Decimal x	Fitness f(x)=x2	Probability (Fitness / Sum)	Expected Count	Actual Count
1	1100	12	144	0.14	0.59	1
2	11001	25	625	0.64	2.56	3
3	101	5	25	0.02	0.1	0
4	10011	19	361	0.37	0.74	1

Fitness Value= 625

Genetic Algorithms

Exercise

2-Selection

String No	Chromosome	Decimal x	Fitness f(x)=x2	Probability (Fitness / Sum)	Expected Count	Actual Count
1	1100	12	144	0.14	0.59	1
2	11001	25	625	0.64	2.56	3
3	101	5	25	0.02	0.1	0
4	10011	19	361	0.37	0.74	1

Fitness Value= 625

Genetic Algorithms

Exercise

2-Crossover

String no	Mating pool	Crossover point	Offspring after crossover	X	Fitness Function
1	1100	**2**	1001	**9**	**81**
2	11001	**2**	11100	**28**	**784**
2	11001	**3**	11011	**27**	**729**
4	10011	**3**	10001	**17**	**289**

Fitness Value= 784

Genetic Algorithms

Exercise

4-Mutation

String no	Offspring after crossover	Mutation chromosome for flipping	Offspring after flipping	X	F(x)
1	1001	10000	11001	**25**	**625**
2	11100	0	11100	**28**	**784**
4	11011	0	11011	**27**	**729**
5	10001	101	10100	**20**	**400**

Fitness Value= 784

PARTICLE SWARM OPTIMISATION

Inspiration

Particle Swarm

1. Swarm Intelligence (SI)

- In reality, a swarm works through local interactions between individuals, without a central leader, yet the group as a whole behaves in a coordinated way.

2. Individuals Follow Simple Local Rules

- Each bird, fish, or insect doesn't know the entire plan.
- They just react to their immediate surroundings:
- **Alignment** → Match direction with nearby members.
- **Cohesion** → Move toward the center of the group.
- **Separation** → Avoid collisions.

Inspiration

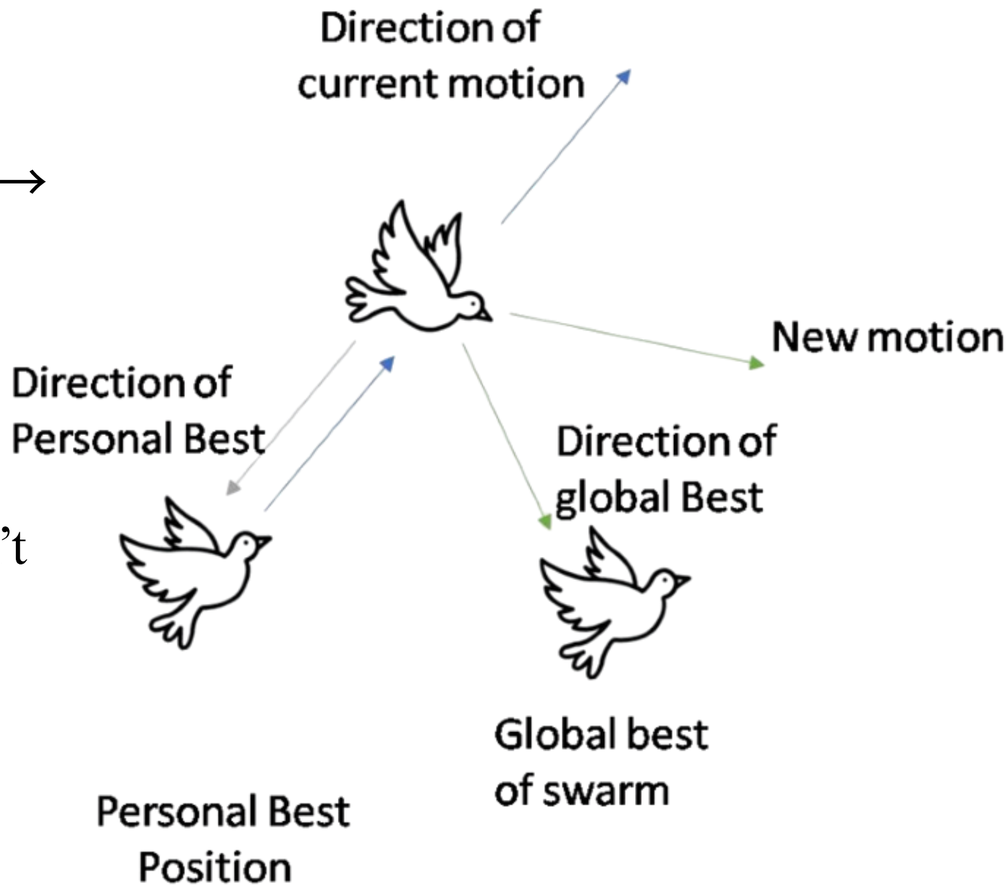
Particle Swarm

3. Information Spreads Indirectly

- If a few members change direction (e.g., avoid a predator or move toward food), their neighbors copy them.
- This change ripples through the swarm like a wave.
- Example: One fish spots a predator → turns sharply → neighbors follow instantly → whole school pivots.

4.No Leader, Just Decentralized Control

- Coordination comes from self-organization.
- Rules are built into the animals' instincts — they don't "vote" or "plan."
- This makes swarms fast and adaptive in chaotic environments.



Particle Swarm

Particles

- Represent candidate solutions in the search space.
- Each particle has a position (current solution) and a velocity (direction and speed of movement).

Position (xi)

- The location of the particle in the solution space.
- This is the actual solution the algorithm is evaluating.

Velocity (vi)

- Determines how far and in what direction the particle will move in the next iteration.
- Updated based on:
- **Inertia weight (a)** → keeps some of the old velocity.
- **Cognitive component (b1)** → pull toward the particle's personal best.
- **Social component (b2)** → pull toward the global best the swarm finds.

Particle Swarm

Exercise

Initilization

- a=0.5
- b1=b2=1.5
- r1=0.6
- r2=0.3

- Initial swarm:
- Particle 1: x=4,v=0
- Particle 2: x=−2,v=0

- **Goal: Minimize f(x)=x2**

Particle	v	x	f(x)	pbest	gbest
1	1	4	16	4	-2
2	-1	-2	4	-2	-2

Particle Swarm

Exercise

Velocity updates:

- **v1**=0.5 (1)+1.5 (0.6) (4−4)+1.5 (0.3)(−2−4) =0.5+0−2.7=−2.2
- **v2**=0.5 (−1)+1.5(0.6)(−2−(−2))+1.5(0.3)(−2−(−2))=−0.5+0+0=−0.5

Position updates:

- **x1**=4+(−2.2)=1.8
- **x2=**−2+(−0.5)=−2.5

Evaluate fitness:

- **f(1.8)**=3.24 → update pbest for particle 1 (was 4).
- **f(−2.5)**=6.25 → no change for particle 2's pbest (-2 is better).
- gbest now x=1.8 (best fitness = 3.24)

Particle	v	x	f(x)	pbest	gbest
1	-2.2	1.8	3.24	1.8	1.8
2	-0.5	-2.5	6.25	-2	1.8

Particle Swarm

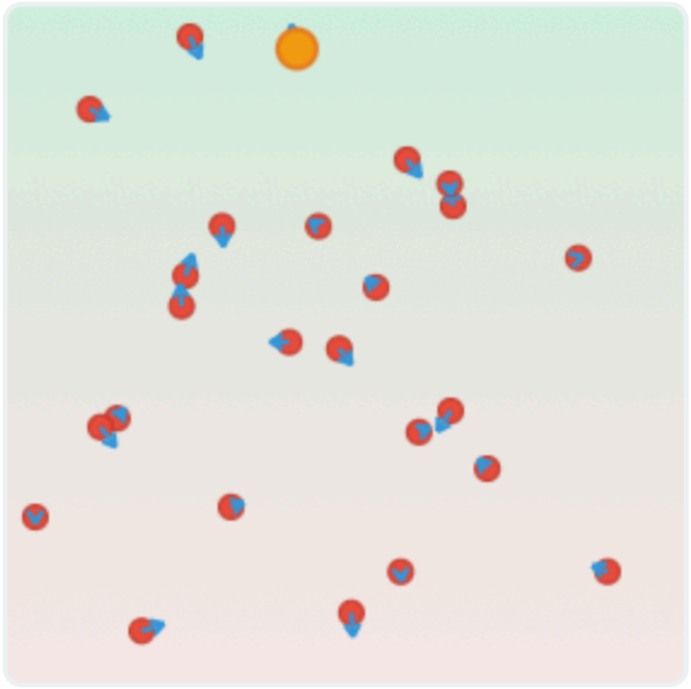
Exercise

After 1 Iteration

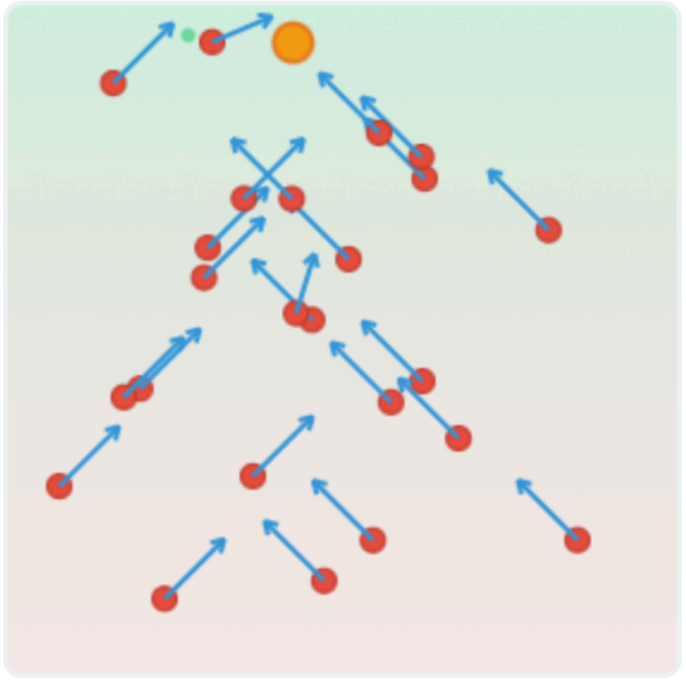
Particle	v	x	f(x)	pbest	gbest
1	-1.1	0.7	0.49	0.7	-0.365
2	2.135	-0.365	0.133	-0.365	-0.365

Particle Swarm

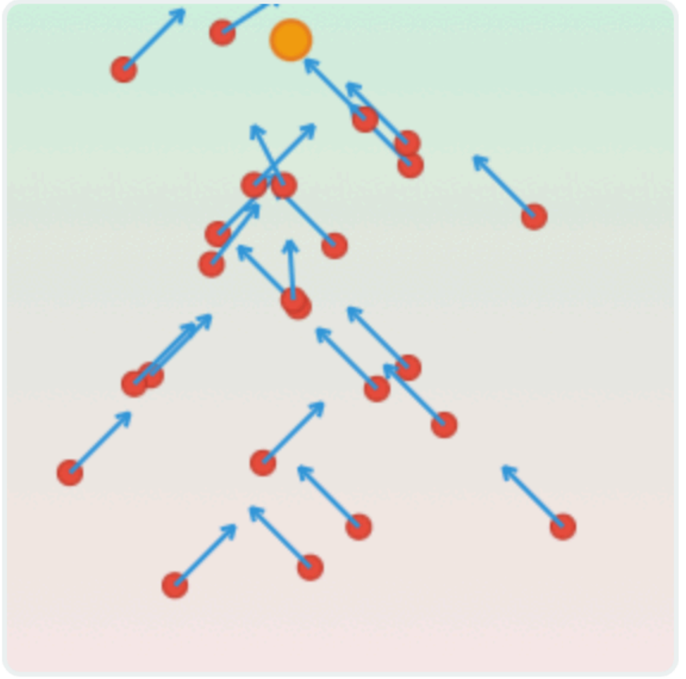
Initial Distribution

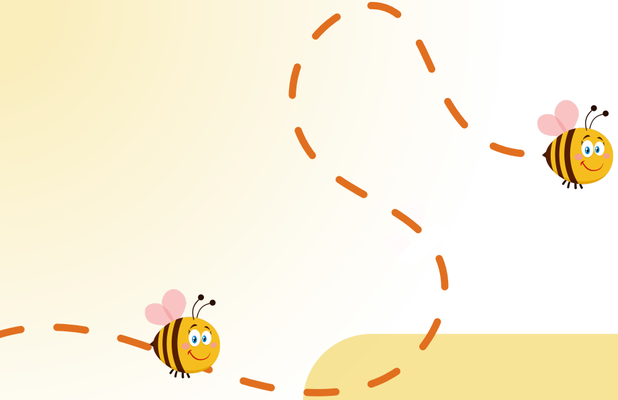


After 2 Iterations



After 3 Iterations





ARTIFICIAL BEE COLONY

Inspiration

Bee Colony

Three Types of Bees

🐝 **Employed Bees:**

- Work with known food sources
- Perform waggle dance to share location & quality.

🕵️ **Scout Bees:**

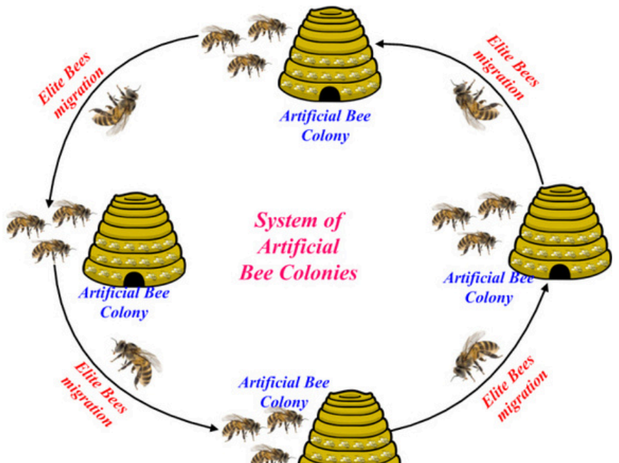
- Explore randomly for new food sources
- Become employed bees when they find good sources.

👀 **Onlooker Bees:**

- Watch waggle dances.
- Choose food sources based on dance intensity.

The Process

1. Scout bees explore randomly
2. Employed bees exploit known sources and dance
3. Onlooker bees select sources based on dances
4. Poor sources are abandoned, good ones get more workers



Bee Colony

-Food Source = Solution

- Position in the search space represents a potential solution.
- Nectar amount = objective function value.

-Waggle Dance = Information Sharing

- Employed bees share solution quality.
- Probability calculation: $Pi = Fi / \Sigma(Fj)$.

-Abandonment Criteria

- If no improvement after set trials → abandon.
- Scout bee generates a new random solution.

ABC Process Flow

- Initialize population randomly
- Employed bee phase: Generate new solutions
- Onlooker bee phase: Select and improve solutions
- Scout bee phase: Replace abandoned solutions
- Repeat until convergence

Bee Colony

Exercise

Step 1: Initialization

- Population Size: 4 food sources
- Variables: 2 (x_1, x_2)
- Search Range: [-5, 5]
- Max Iterations: 3

Minimize $f(x_1, x_2) = x_1^2 + x_2^2$

Fitness = 1/(1 + f(x)) (higher is better)

Bee	x1	x2	f(x)	Fitness
1	2	3	13	0.071
2	-1	4	17	0.056
3	3	-2	13	0.071
4	-4	1	17	0.056

Bee Colony

Exercise

Step 2: Employed Bee Phase

- Generate new solutions: $v_i = x_i + \varphi(x_i - x_j)$
- φ = random number [-1, 1]

Bee	Old (x1,x2)	Old f(x)	New Solution (x1,x2)	New f(x)	Accept?
1	(2, 3)	13	(1.5, 2.8)	10.1	✓ Better
2	(-1, 4)	17	(-0.8, 3.9)	15.8	✓ Better
3	(3, -2)	13	(2.7, -1.6)	9.8	✓ Better
4	(-4, 1)	17	(-3.8, 0.9)	15.2	✓ Better

Bee Colony

Exercise

Step 3: Onlooker Bee Phase

Calculate probabilities: $P_i = Fitness_i / \Sigma(Fitness)$

Bee	New Solution (x1,x2)	New f(x)	Fitness	Probability	Selected?
1	(1.5, 2.8)	10.1	0.09	0.28	✓
2	(-0.8, 3.9)	15.8	0.059	0.18	–
3	(2.7, -1.6)	9.8	0.092	0.29	✓
4	(-3.8, 0.9)	15.2	0.062	0.19	–

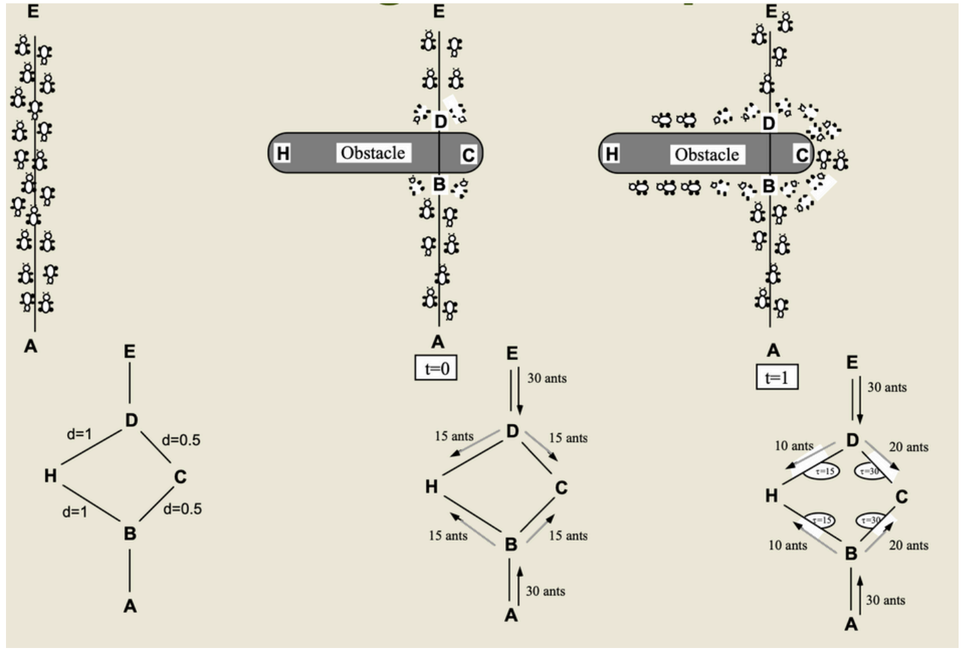
ANT COLONY ALGORITHM

Inspiration

Ant Colony

1-Individual Ant

- Initially, explore randomly to find food.
- Leave pheromone trails while walking.
- Follow existing pheromone trails with higher probability.
- Carry food back to the nest.



Ant Colony

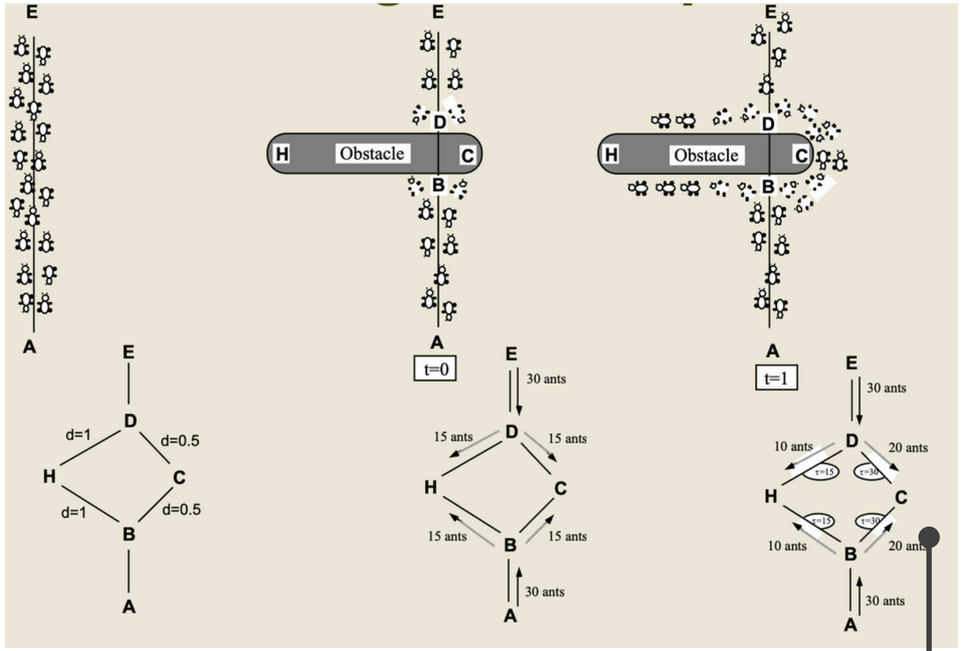
2-Pheromone Communication System:

Trail Formation

- Ants deposit chemical trails (pheromones) as they move.
- Stronger trails indicate better/shorter paths.
- Pheromones evaporate over time.

Trail Following

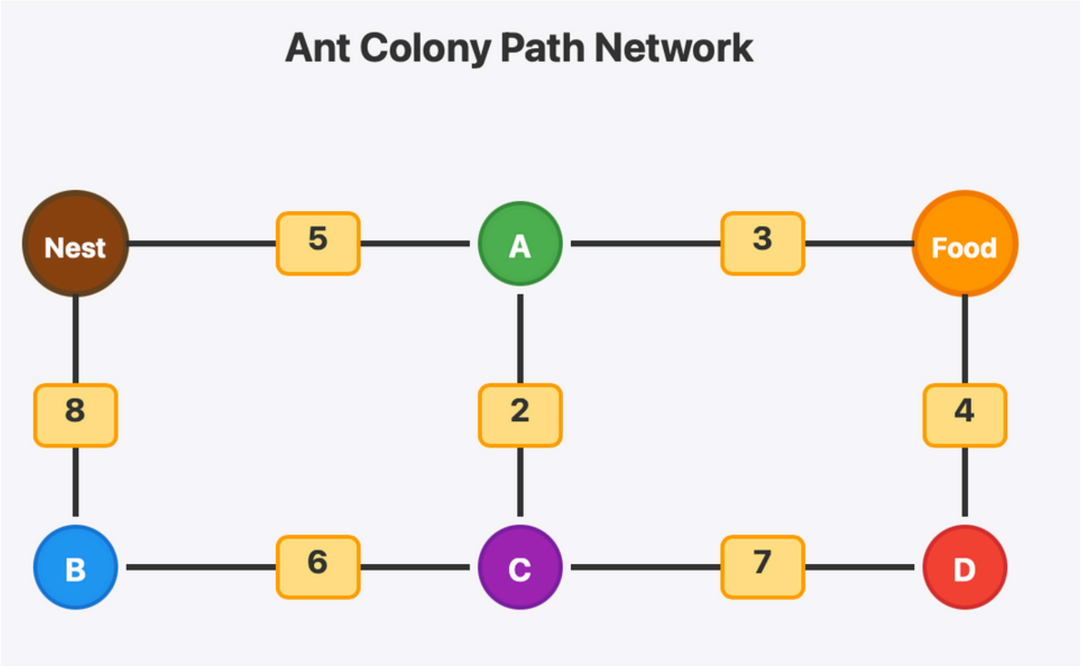
- Ants probabilistically choose paths
- Higher pheromone concentration = higher probability
- Not 100% deterministic - some exploration remains.

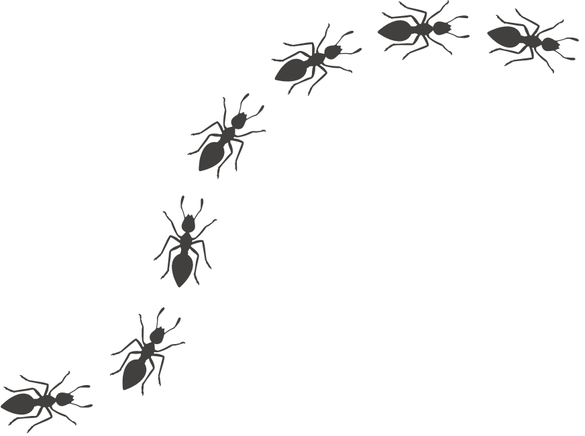


Ant Colony

3-The Shortest Path Discovery

- Random exploration - ants spread out randomly.
- Path marking - all ants leave pheromone trails
- Feedback loop - shorter paths get traversed faster.
- Reinforcement - more ants use shorter paths.
- Convergence - colony finds optimal route.





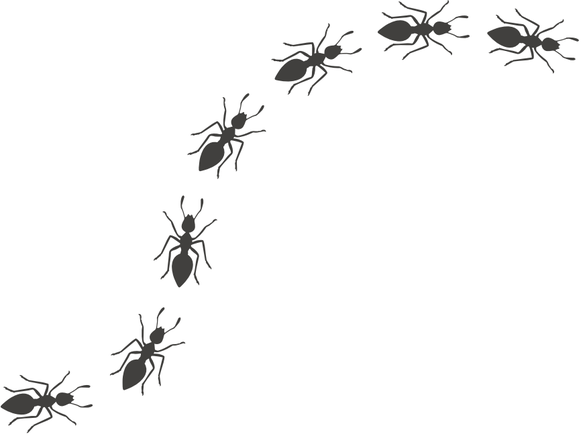
Ant Colony

Exercise

Path Options:

- Path 1: Nest → A → Food
  (distance = 5 + 3 = 8)
- Path 2: Nest → B → C → A → Food
  (distance = 8 + 6 + 2 + 3 = 19)
- Path 3: Nest → B → C → D → Food
  (distance = 8 + 6 + 7 + 4 = 25)

Edge	Distance	Pheromone
Nest–A	5	1
Nest–B	8	1
A–Food	3	1
A–C	2	1
B–C	6	1
C–D	7	1
D–Food	4	1



Ant Colony

Exercise

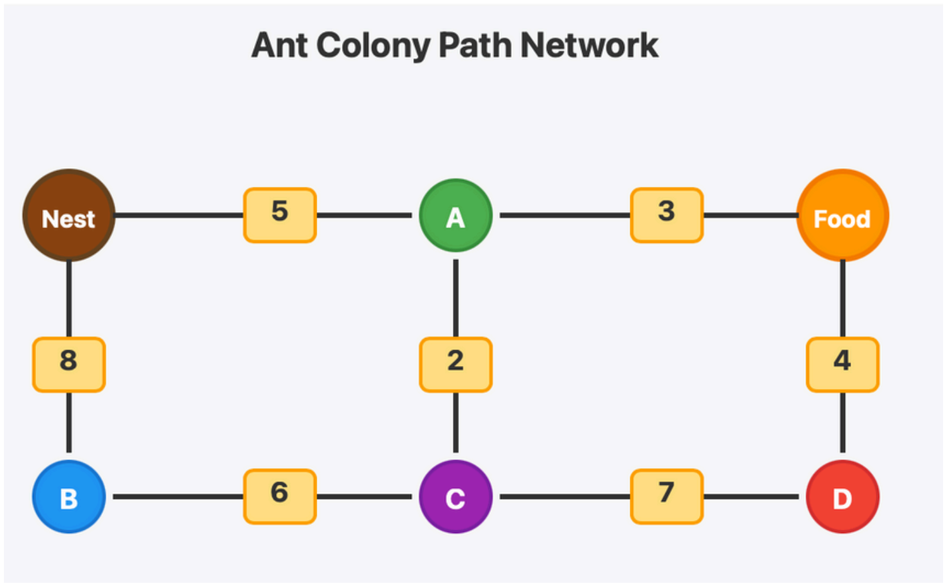
After 3 Ants Complete Their Journeys:

- Ant 1: Nest → A → Food (Path 1, distance = 8) Ant 2: Nest → A → Food (Path 1, distance = 8)
- Ant 3: Nest → B → C → D → Food (Path 3, distance = 25)

Edge	Pheromone deposited	New pheromone
Nest–A	25	26
A–Food	25	26
Nest–B	4	5
B–C	4	5
C–D	4	5
D–Food	4	5
A–C	0	1

Ant Colony

Exercise



Path Choice	Pheromone	Probability
To A	26	26.031.0≈84%
To B	5	5.031.0≈16%

CUCKOO SEARCH

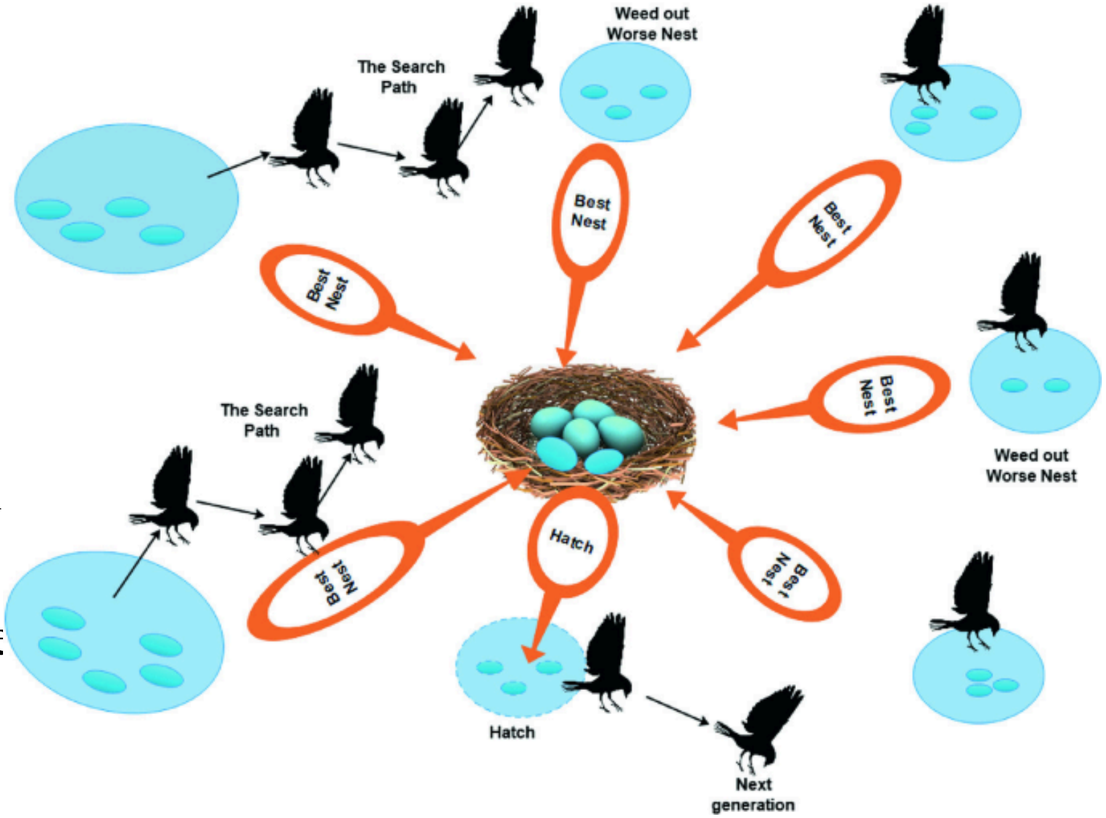
Cuckoo Search

Egg Laying Process:

- Search for suitable host nests
- Mimic the host bird's egg appearance
  - Replace or add an egg to the host nest
- Abandon nest - let the host raise the chick

Host Bird Response:

- Accept the egg if it looks similar to their own
- Reject the egg if detected as foreign
- Abandon the entire nest if too many foreign e



Cuckoo Search

Lévy Flight Pattern

- Cuckoos search using Lévy flights.
- Mix of short local searches + long jumps.
- Efficient exploration of large areas.
- Found in many foraging animals.

Egg Quality

- Better mimicry = higher survival chance.
- Quality eggs are less likely to be discovered.
- Natural selection favors better eggs.

Discovery Probability

- Some eggs are always discovered.
- Host birds have varying detection abilities.
- Trade-off between quality and discovery risk.

Cuckoo Search

Initial Setup

- **Population**: 4 nests (solutions)
- **Search Range:** [-10, 10]
- **Discovery Probability:** Pa = 0.25
- **Target: Find the minimum x= (x−2)2**
- **Step 1: Initialize Nests**

Nest	x-value	f(x)	Quality 1f(x)
1	-3	25	0.04
2	1	1	1
3	6	16	0.063
4	-1	9	0.111

Cuckoo Search

Best Nest: **Nest 2 with f(x) = 1**

Step 2: Lévy Flight Search

Generate new solutions using Lévy flight:

$x_{new} = x_{old} + \alpha \times \text{Lévy}(\lambda)$

Nest	Old x	Lévy Step	New x	New f(x)	Accept?
1	-3	4.2	1.2	0.64	✓ Better
2	1	0.8	1.8	0.04	✓ Better
3	6	-2.1	3.9	0.81	✓ Better
4	-1	1.5	0.5	2.25	✗ Worse

Cuckoo Search

Step 3: Discovery and Abandonment

Check each nest with probability Pa = 0.25

Nest	Random Value	Pa=0.25	Discovered?	Action
1	0.15	< 0.25	YES	Generate new random
2	0.4	> 0.25	NO	Keep solution
3	0.8	> 0.25	NO	Keep solution
4	0.2	< 0.25	YES	Generate new random

OTHER ALGORTHIMS

Other Algorithms

2001 - Harmony Search (HS)

- Inspiration: Musical improvisation process
- Details: Musicians improvise harmonies from memory, pitch adjustment, or random selection. Optimization variables are like musical notes.

2002 - Bacterial Foraging Optimization (BFO)

- Inspiration: Foraging behavior of E. coli bacteria
- Details: Mimics chemotaxis, swarming, reproduction, and elimination-dispersal events in bacterial colonies.

Other Algorithm

2005 - Fish School Search (FSS)

- Inspiration: Fish schooling behavior
- Details: Fish adjust their positions based on individual and collective experiences. Uses a weight-based selection mechanism.

2006 - Gravitational Search Algorithm (GSA)

- Inspiration: Newton's law of gravity and mass interactions
- Details: Agents have masses that attract each other. Heavier masses move more slowly and represent better solutions.

2007 - Biogeography-Based Optimization (BBO)

- Inspiration: Biogeography and species migration
- Details: Solutions share features based on their habitat suitability index (HSI). Migration occurs between habitats.

Other Algorithm

2013 - Grey Wolf Optimizer (GWO)

- Inspiration: Leadership hierarchy and hunting of grey wolves
- Details: Alpha, beta, and delta wolves lead the pack. Omega wolves follow the three leaders in hunting prey (optimization).

2014 - Coyote Optimization Algorithm (COA)

- Inspiration: Social behavior of coyotes
- Details: Coyotes live in packs with alpha leadership. Birth and death of pups maintain population diversity.

2015 - Whale Optimization Algorithm (WOA)

- Inspiration: Humpback whale bubble-net hunting strategy
- Details: Whales encircle prey, create bubble nets, and perform spiral movements to capture prey efficiently.

Other Algorithm

2020 - Chimp Optimization Algorithm (ChOA)

- Inspiration: Intelligent behavior and hunting strategies of chimps
- Details: Chimps hunt in groups with different roles: attacker, blocker, chaser, and driver. Social hierarchy influences hunting success.

2021 - Artificial Gorilla Troops Optimizer (GTO)

- Inspiration: Social intelligence of gorilla troops
- Details: Gorillas follow the silverback leader, migrate to new locations, and compete for leadership. Balances exploration and exploitation.

2023 - Coati Optimization Algorithm (COA)

- Inspiration: Coati behavior and social organization
- Details: Coatis organize in groups with flexible social structures, forage collectively, and adapt to environmental changes.

BIO-INSPIRED ALGORITHMS IN AI

Bio-Inspired algorithms in AI

1. Machine Learning Optimization

Purpose: Optimize model parameters, hyperparameters, and architecture.

- **Hyperparameter Tuning:** ABC, GWO for finding optimal learning rates, batch sizes, and regularization parameters.
- **Feature Selection:** ACO, GA to select the most relevant features from high-dimensional datasets.

Bio-Inspired algorithms in AI

2. Deep Learning Applications:

Purpose: Enhance deep learning model performance and efficiency.

- **Convolutional Neural Networks:** Optimizing filter sizes, layer depths, activation functions
- **Recurrent Networks:** Finding optimal LSTM/GRU architectures for sequence prediction
- **Generative Models**: Optimizing GAN architectures and training parameters
- **Transfer Learning:** Selecting optimal pre-trained models and fine-tuning strategies

Bio-Inspired algorithms in AI

3. Reinforcement Learning

Purpose: Improve policy learning and exploration strategies.

- **Policy Optimization:** Using evolutionary strategies for policy gradient methods
- **Exploration Strategies:** Bio-inspired algorithms for better exploration-exploitation balance
- **Multi-Agent Systems:** Swarm intelligence for coordinated agent behavior
- Reward Function Design: Evolutionary approaches to shape reward functions.

ANY QUESTION?

THANK YOU