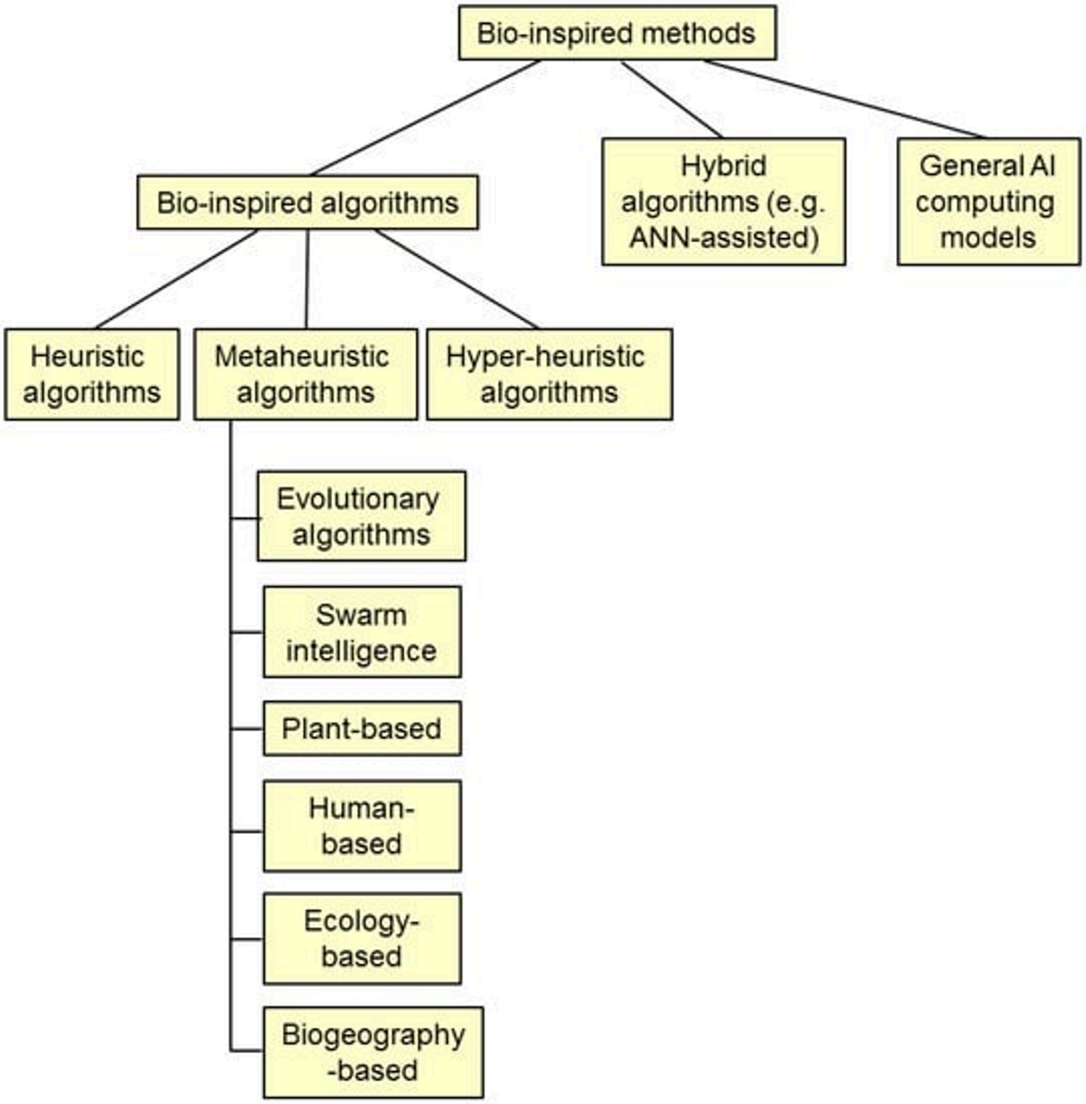
Conventional vs swarm intelligence:



Nature-inspired algorithms can be stronger at non-continuous, non-differentiable and noisy problems than conventional algorithms such as gradient descent. Swarm intelligence algorithms are a subset of nature-inspired optimization algorithms. SI is good at tackling non-deterministic polynomial time problems. (NP). NP-hard problems are a class of problems that are at least as hard as the hardest problems in NP

Key differences from conventional algorithms:

- Individual units have dynamic components.
    - This includes exploitation parameters and randomization
- Information is being exchanged between units

Leads to these properties:

- Less computational cost, time

- Less human intervention

Flowchart of conventional algorithms:

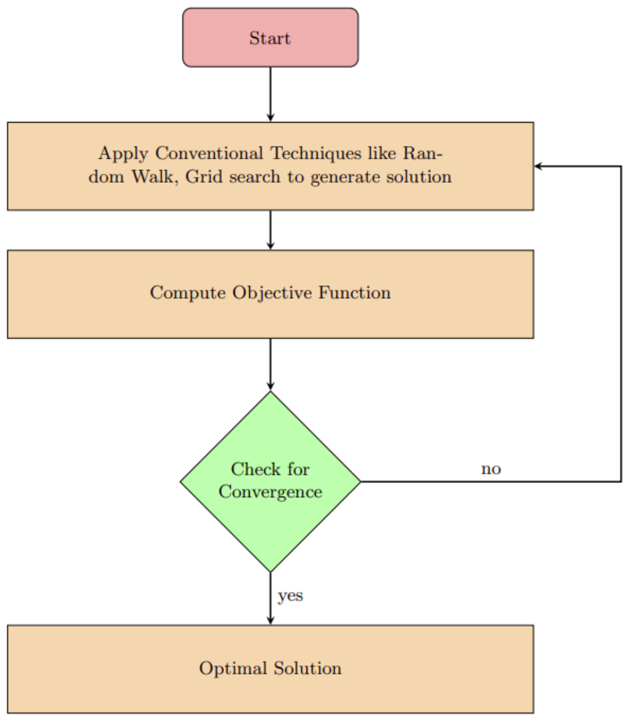
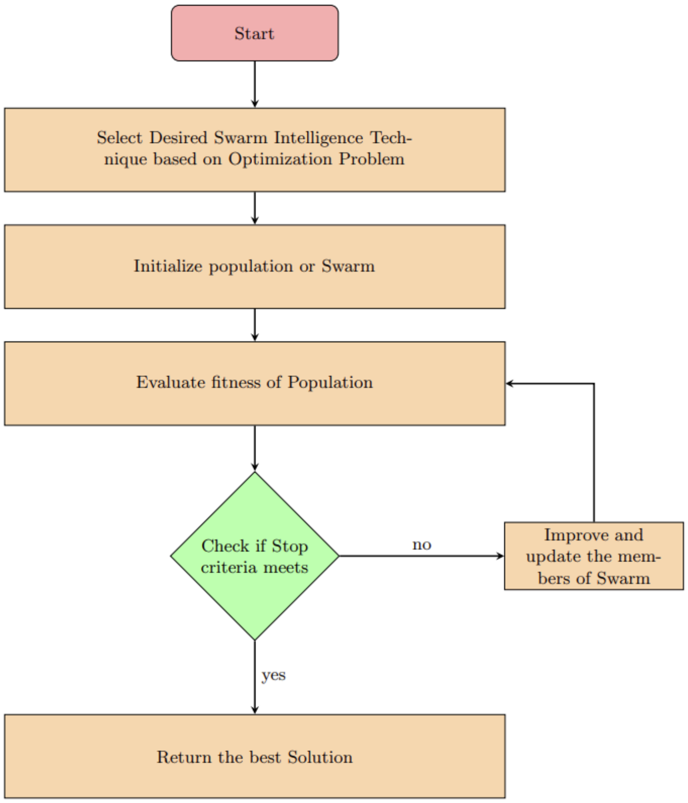
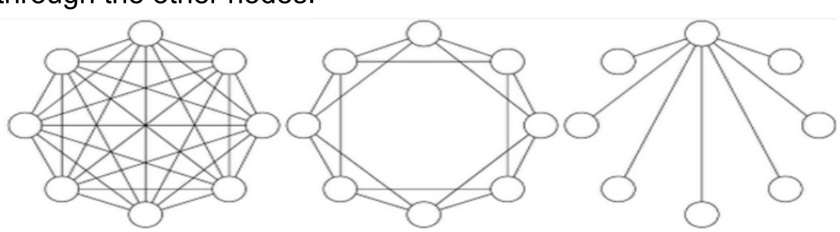


Fig. 1. Generic Flowchart of optimization using Traditional Techniques

General flowchart for swarm intelligence optimization:



- Initialization
- Evaluate objective function/fitness function: calculate $f(x_i, t)$, the objective function. Attempt to $f(x^*)$ where the objective value is maximized. $f(x)$ could be the pollen density, or pest density within a particular sensing radius.
- Update: Changes in the velocity and position of individual members
    1. Personal best: if $f(x_i, t) \geq f(x_i, best)$: then $f(x_i, best) = f(x_i, t)$ and $p = x_i, t$
    2. Global best: if $f(x_i, t) \geq f(x_i, best)$: then $f(x_i, best) = f(x_i, t)$ and $g = x_i, t$
- Population topology: How many other nodes each node is connected to will affect the swarm algorithm efficiency. Local best could refer to using the information passing through the other nodes.



- Modifications:
    - Can be measured by the convergence rate
    - Since there are multiple points with pests, the global best should be updated over time.

Efficiency comparison:

Best and robust on low - medium dimensional problems:
Bald eagle search optimization algorithm, Chimp optimization algorithm, Electrostatic discharge algorithm, Fbi-inspired meta-optimization, Fractal search algorithm.
The average, best and worse run times and standard deviation of these algorithms were the top 5 of all tested algorithms. A low SD is more desirable and it indicates consistency across different parameters.

Best and robust functions across search space sizes:
Coronavirus herd immunity optimizer. Highest average, best, worst, and standard deviation for all functions. This means that increasing the maximum number of iterations did not affect the summary statistics for the algorithm performance.

Fastest algorithms:
Light spectrum optimizer, Differential evolution, and Reptile search algorithm. These algorithms run with the least time on benchmark functions in low to medium dimensions. However, RSA has a higher time complexity than the other two algorithms.

Experimental conditions:
They are tested against 45 benchmark functions, with low/medium dimensions, narrow and wide search spaces, simple to medium to complex functions. Population size = 30.
Max iteration = 100, 300, 500 and 800
Each function is independently run 40 times
The unit is not explicitly defined but is likely time in seconds.

Update rules:

Particle swarm optimization

Velocity updates: $v_i^{t+1} = v_i^t + r_1 * \alpha(g - x_i^t) + r_2 * \beta(p - x_i^t)$

Position updates: $x_i^{t+1} = x_i^t + v_i^{t+1}$

p = local best
g = global best
α, β = randomization parameters as weights ranging from 0 - 1

Firefly algorithm

Assumptions:

1. The intensity of flashing lights is the sole factor that causes fireflies to be attracted to each other.
2. The brighter the light, the more attractive. If there is no light then fireflies perform a random walk.
3. The Intensity can be calculated from the objective function

$$x^{t+1} = x_i^{t} + \; randomization\; term \; + \; attractiveness\; term$$

$$= x^{t} + \; \alpha\epsilon_i^{t} + \beta_0 e^{-\gamma r_{i,j}^{2}} (x_j^{t} - x_i^{t})$$

Interpretation: The randomization term + attractiveness term are very similar to the velocity term above but written in terms of x instead. The exponential term might be related to flying in a circular way or to model how brightness decays as the radius increases, therefore becoming less attractive.

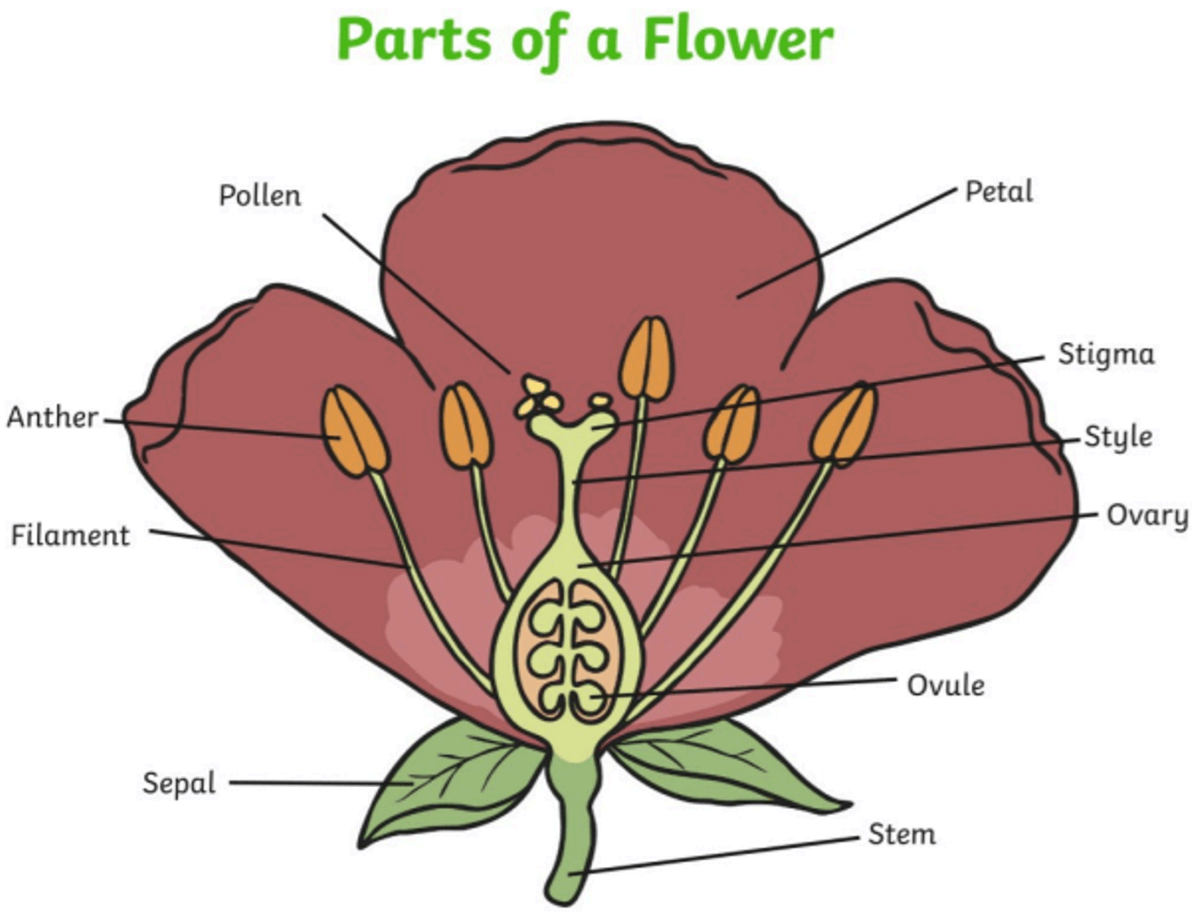
α = random parameter

ϵ = a vector of random numbers, presumably with dimensionally many elements, obtained by sampling from any distribution.

β_0 = the brightness at r = 0

FA can be reduced down to other algorithms

Flower pollination algorithm



Forms of pollination:

Cross-pollination: pollen goes from one flower to another. Pollen goes from the surface of the anther → the surface of the stigma. Pollen is attached to pollinators (hence the name). Within the algorithm, a pollen symbolizes a solution vector that is being selected.

Self-pollination: Pollen transfers within the same flower. Only used by 10% of plants.

Algorithm:
Exploration probability to decide exploration or exploitation, p.

The main loop: For iterations < t, for flower in all flowers:

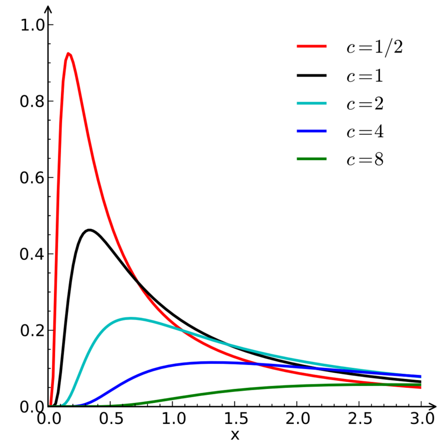
If the random number generated < p:

Use exploration/cross-pollination. The pollinator explores according to Levy's flight, represented by a d-dimensional step vector from a levy distribution.

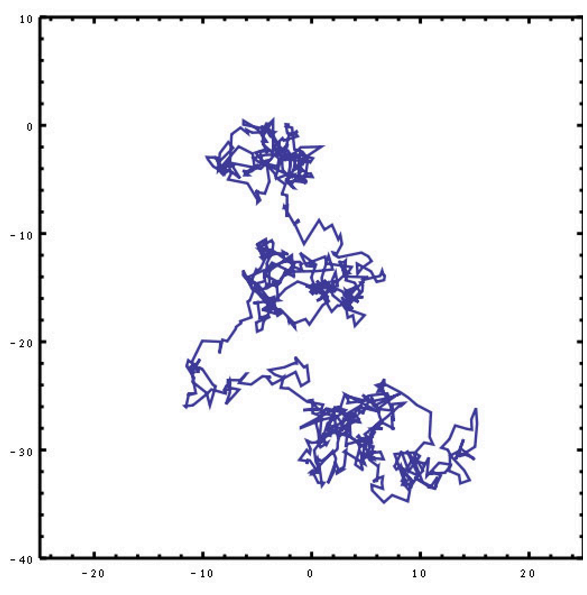
Levy flight simulates how fruit flies move and how many animals walk, containing a mix of short and long steps. In Levy Flight, the step sizes are drawn from a probability distribution that is heavy-tailed (specifically called a Levy distribution). Short steps are more probable than long steps. Because it is a probability distribution, the area under the curve sums to one.

$$x_i^{t+1} = x_i^t + L(g - x_i^t)$$

Levy distribution, L(x):



Levy's flight, 2d visualization.



If random number > p: Perform local pollination/exploitation. Draw a step from a uniform distribution.

$x_i^{t+1} = x_i^t + u(x_j^t - x_k^t)$

x_j^t, x_k^t: two randomly selected flowers at time t. This difference vector $x_j^t - x_k^t$ helps to explore the search space.

u: drawing from a uniform distribution

Cuckoo search algorithm:

Assumptions for an idealized cuckoo:

1. Each cuckoo lays one egg at a randomly selected nest. In the algorithm, the basic unit is the egg. Each egg represents a solution, the new egg laid by the cuckoo represents a new solution.
2. Best nests with high-quality eggs are carried over to the next generation (of cuckoo egg). the cuckoo egg will be selected when it's the most similar to the high-quality eggs are solutions near-optimal value. the new solution will replace the worse solution. For simplicity, you may assume each nest only has a single egg which can be replaced. can be generalized to having multiple eggs
3. Host discovers a cuckoo egg with probability p_a: If the cuckoo egg fitness is less than the host bird, the egg is thrown away. Following, cuckoo may create a new one with random probability (simple random)

Nature inspiration:

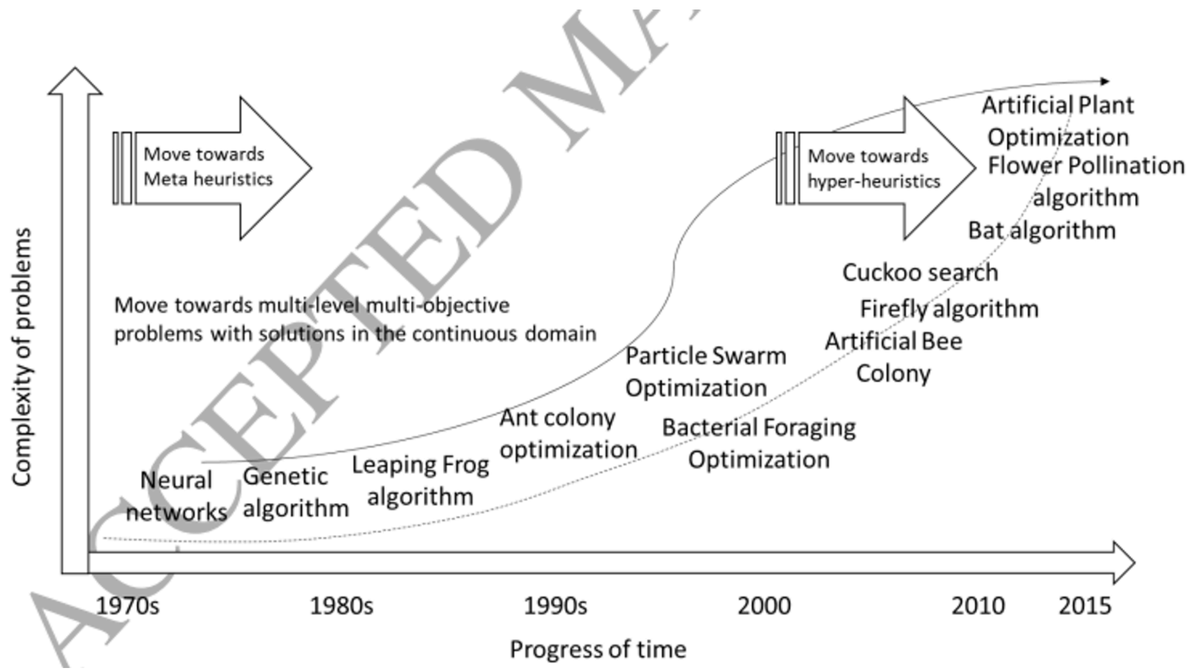
Parasitic cuckoos often choose a nest where the host bird just laid its own eggs. Cuckoo hatchlings can kick eggs out of the nest.

Some cuckoo species such as the New World brood-parasitic Tapera have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry of colour and pattern of the eggs of a few chosen host species.

Studies also show that a cuckoo chick can also mimic the call of host chicks to gain access to more feeding opportunities.

Test functions for swarm algorithms

History



Move towards
Meta heuristics
Move towards
hyper-heuristics
Artificial Plant
Optimization
Flower Pollination
algorithm
Bat algorithm
Cuckoo search
Firefly algorithm
Artificial Bee
Colony
Move towards multi-level multi-objective
problems with solutions in the continuous domain
Particle Swarm
Optimization
Ant colony
optimization
Bacterial Foraging
Optimization
Neural
networks
Genetic
algorithm
Leaping Frog
algorithm
Complexity of problems
1970s
1980s
1990s
2000
2010
2015
Progress of time

Applications

Quadrant 1: Zone of Theory Development	Quadrant 2: Zone of Applications
• Amoeba (Zhang et al., 2013) • Artificial plant optimization (Cui & Cai, 2013) • Bean optimization (Zhang et al., 2010) • Dove (Su et al., 2009) • Eagle (Yang & Deb, 2010) • Fruit fly (Pan, 2012) • Glow-worm (Krishnanand & Ghose, 2005) • Grey wolf algorithm (Mirjalili et al. 2014) • Krill-herd (Gandomi & Alavi, 2012) • Lion (Yazdani & Jolai, 2015) • Monkey (Mucherino & Seref, 2007) • Wolf (Liu et al., 2011)	• Bacterial foraging (Passino, 2002) • Bat algorithm (Yang, 2010) • Bee colony (Karaboga, 2005) • Cuckoo search (Yang & Deb, 2009) • Firefly algorithm (Yang, 2009) • Flower pollination (Yang, 2012)
Quadrant 3: Zone of Rediscovery	**Quadrant 4: Zone of Commercialization**
• Leaping Frog (Snyman, 1982) • Shark (Hersovici et al., 1998) • Wasp (Theraulaz et al. 1991)	• Ant colony optimization (Dorigo et al., 2006) • Genetic algorithm (Holland, 1975) • Neural Networks (Grossberg, 1988) • Particle swarm (Shi & Eberhart, 1999)

Applications: the bio-inspired algorithms have been explored for various telecommunication applications, including routing in sensor networks (Chaudhry et al., 2019), electromagnetic antenna design (Yang et al., 2002; Mohammed et al., 2016), mobility management (Swayamsiddha et al., 2018, 2019; Parija et al., 2017), filter design (Storn, 1996), home automation networks (Wang et al., 2015), spectrum sensing in cognitive radio (Azmat et al., 2015), and Internet of Things (IoT)

Glossary

Flower pollination is good for non-linear and multimodal optimization problems

Non-linear: The term "non-linear" in the context of an optimization problem means that the relationship between the decision variables and the objective function is not linear. This implies that the objective function does not change proportionally with changes in the variables. Non-linear problems can exhibit a wide range of behaviors, such as curvature, where the rate of improvement in the objective function varies across the decision space.

Multimodal: A "multimodal" optimization problem is one that has multiple optima (i.e., more than one solution that is either a maximum or a minimum, depending on whether the problem involves maximization or minimization).

There's a risk of algorithms getting "trapped" in local optima, especially if they employ strategies that rely on gradient descent or similar approaches that follow the steepest path to nearby optima without considering the broader search space.

Random walk: A random walk can be thought of as a Markov chain, whose next status (the location) only depends on the current location and the transition probability (the second term). For example, where are you going to be in the city with some probability of l, f, r at an intersection?

A stochastic random walk is when step sizes aren't fixed. Within the cuckoo search algorithm, the step sizes are generated by Levy distribution. This formulation is called the Levy flight.

Important sources:

SI Review: https://arxiv.org/pdf/2209.12823
Evaluations
https://www.researchgate.net/publication/373088021_Swarm_Intelligence_Algorithms_Evaluation