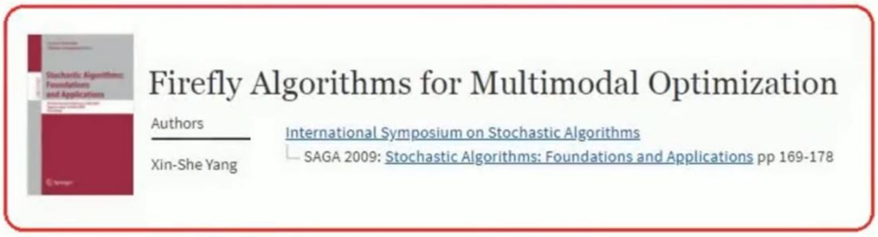
Firefly algorithm (FA) is a bio-inspired metaheuristic algorithm for optimization problems.

FA was first developed by Xin-She Yang in late 2007 and 2008.



The algorithm is inspired by the flashing patterns and behaviour of fireflies at night.

One of the three rules used to construct the algorithm is that

1) all fireflies are unisex, which means any firefly can be attracted to any other brighter one regardless of their sex.

2) the brightness of a firefly is determined from the encoded objective function.

3) The attractiveness is directly proportional to brightness, and they both decreases as their distance increases,

- It means a firefly will move towards the brighter one, and if there is no brighter one, it will move randomly.

From elementary physics, it is clear that the intensity of light is inversely proportional to the square of the distance, say "r", from the source. So, we can define the variation of attractiveness β with the distance r by

$$\beta(r) = \beta_0 e^{-\gamma r^2}$$

where β_0 is the attractiveness at $r = 0$.

The position update of the firefly located at X_i will be done as follows:

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(X_j^t - X_i^t\right) + \alpha_t \in_i^t$$

Here, the second term is due to the attraction towards X_j.

The last term is a randomization term with α_t being a randomization parameter with $0 \leq \alpha_t \leq 1$ and $\in_i^t$ is a vector of random numbers drawn from a Gaussian or uniform or other distribution at time t.

The position update of the firefly located at X_i will be done as follows:

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(X_j^t - X_i^t\right) + \alpha_t \in_i^t$$

If $\beta_0 = 0$, it becomes a simple random walk.

If $\gamma \to 0$, the FA reduces to standard Particle Swarm Optimization (PSO)

In fact, if the inner loop (for j) is removed and the brightness X_j is replaced by the current global best, then FA essentially becomes the standard PSO.

Parameter Settings

Position Update Equation

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(X_j^t - X_i^t\right) + \alpha_t \in_i^t$$

As α_t essentially controls the randomness (or, to some extent, the diversity of solutions), we can tune this parameter during iterations so that it can vary with the iteration counter t. A good way to express α_t is to use,

$$\alpha_t = \alpha_0 \delta^t, \quad 0 < \delta < 1$$

Where α_0 is the initial randomness scaling factor and δ is essentially a cooling factor.

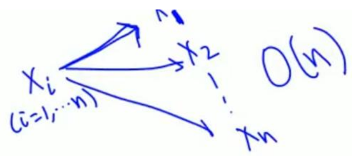
For most applications, we can use $\delta = 0.95$ to 0.97.

The parameter β_0 controls the attractiveness, and parametric studies suggest that $\beta_0 = 1$ can be used for most applications.

Regarding the initial α_0, simulations show that FA will be more efficient if α_0 is associated with the scalings of design variables.

> Let L be the average scale of the problem of interest,
>
> we can set $\alpha_0 = 0.01L$ initially.

The factor 0.01 comes from the fact that random walks require a number of steps to reach the target while balancing the local exploitation without jumping too far in a few steps.



$$\boxed{X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2}\left(X_j^t - X_i^t\right) + \alpha_t \in_i^t}$$

random.

However, γ should be also related to the scaling L.

In general, we can set $\gamma = \frac{1}{\sqrt{L}}$.

If the scaling variations are not significant, then we can set $\gamma = O(1)$.

Algorithm Complexity

Position Update Equation

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2}\left(X_j^t - X_i^t\right) + \alpha_t \in_i^t$$

Firefly algorithm has two inner loops when going through the population n, and one outer loop for iteration t.

So the complexity at the extreme case is $O(n^2 t)$.

If n is small (typically, n = 50) and t is very large (say $t = 4000$), the computational cost is relatively inexpensive because the algorithm complexity is LINEAR in terms of t.

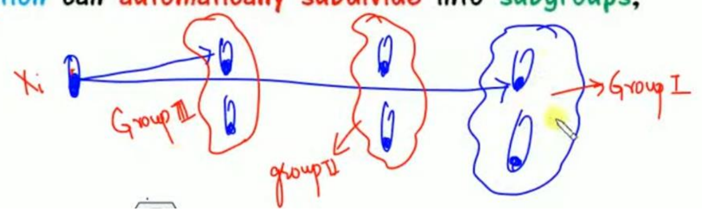
If n is relatively LARGE, it is possible to use one inner loop by ranking the attractiveness (brightness) of all fireflies using SORTING algorithms. In this case, the algorithm complexity of Fireflies algorithm will be $O(nt \log n)$.

Why FA is efficient?

FA has two major advantages over other algorithms: 3-rule

1) automatical subdivision and
2) the ability of dealing with multimodality.

First, FA is based on attraction and attractiveness decreases with distance. This leads to the fact that the whole population can automatically subdivide into subgroups,



Second, this subdivision allows the fireflies to be able to find all optima simultaneously if the population size is sufficiently higher than the number of modes.

In addition, the parameters in FA can be tuned to control the randomness as iteration proceed, so that convergence can also be speed up the tuning these parameters.

Example: Consider two functions to demonstrate the computational cost

For De-Jong's function with $d = 256$ dimensions

$$f(x) = \sum_{i=1}^{d} x_i^2$$

Algorithms	Function evaluation
GA	25412 ± 1237
PSO	17040 ± 1123
FA	5657 ± 730

This save about 78% and 67% computational cost, compared to GA and PSO, respectively.

In the nutshell, FA has three distinct advantages:

1) Automatic subdivision of the whole population into subgroups.
2) The natural capability of dealing with multi-modal optimization.
3) High ergodicity and diversity in the solutions.

Algorithm

The algorithm can be summarized as follows.

(1) Generate a random solution set X,

(2) Compute intensity (I), i.e., $I \propto f(X)$

Max

(3) Update the each firefly with Position Update Equation

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2} \left(X_j^t - X_i^t\right) + \alpha_t \in_i^t$$

(4) Perform Greedy selection

X_1 X_2 upd

$f(X_1)$ $f(X_2)$

$f_{new} > f_{old}$

(5) Terminate if a termination criterion is fulfilled otherwise go back to step 2.

FA parameters:

Population size (No. of fireflies);

Attractiveness constant (β_0);

Absorption coefficient (γ);

Maximum number of iterations;

Randomness strength 0--1 (highly random) (α_0);

Randomness reduction (δ)

$$\alpha = \alpha_0 \delta^t \, ; \; 0 < \delta < 1$$