Nature-Inspired Optimization Algorithms

– A Tutorial

Xin-She Yang

Middlesex University London

A Tutorial at IDEAL 2018

Almost Everything is Optimization

Almost everything is optimization ... or needs optimization ...

- Maximize efficiency, accuracy, profit, performance, sustainability, ...
- Minimize costs, wastage, energy consumption, travel distance/time, CO_2 emission, impact on environment, ...

Mathematical Optimization

Objectives: maximize or minimize $\boldsymbol{f}(\boldsymbol{x}) = [f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_p(\boldsymbol{x})]$,

$$\boldsymbol{x} = (x_1, x_2, ..., x_D) \in \mathbb{R}^D,$$

subject to multiple equality and/or inequality design constraints:

$$h_i(\boldsymbol{x}) = 0, \quad (i = 1, 2, ..., M),$$

$$g_j(\boldsymbol{x}) \leq 0, \quad (j = 1, 2, ..., N).$$

In case of $p = 1$, it becomes a single-objective optimization problem.

Optimization problems can usually very difficult to solve, especially large-scale, nonlinear, multimodal problems.

In general, we can solve only 3 types of optimization problems:

- Linear programming
- Convex optimization
- Problems that can be converted into the above two

Everything else seems difficult, especially for large-scale problems.
For example, combinatorial problems tend to be really hard – NP-hard!

Deep Learning

The objective in deep nets may be convex, but the domain is not convex and it's a high-dimensional problem.

$$\min \ E(\boldsymbol{w}) = \frac{1}{n} \sum_{i=1}^{n} \Big[u_i(\boldsymbol{x}_i, \boldsymbol{w}) - \bar{y}_i \Big]^2,$$

subject to various constraints.

Optimization Techniques

There are a wide spectrum of optimization techniques and tools.

Traditional techniques

- Linear programming (LP) and mixed integer programming.
- Convex optimization and quadratic programming.
- Nonlinear programming: Newton's method, Trusted Region method, Interior Point Method (N. Karmarkar), ..., Barrier Method ... etc.

But most real-world problems are not linear or convex, thus traditional techniques often struggle to cope, or simply do not work...

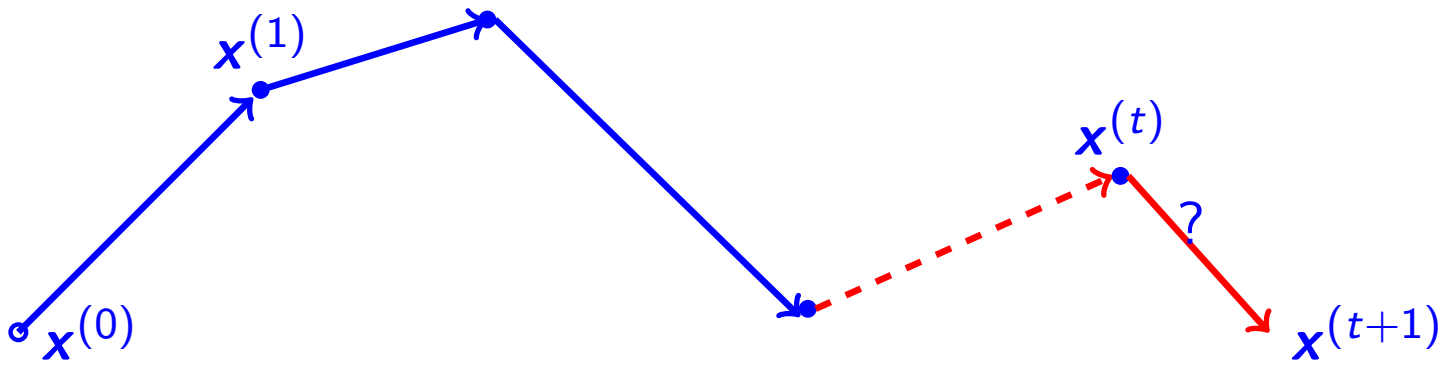
New Trends – Nature-Inspired Metaheuristic Approaches

- Evolutionary algorithms (evolutionary strategy, genetic algorithms)
- Swarm intelligence (e.g., ant colony optimization, particle swarm optimization, firefly algorithm, cuckoo search, ...)
- Stochastic, population-based, nature-inspired optimization algorithms

The Essence of an Algorithm

Essence of an Optimization Algorithm

To generate a better solution point $\boldsymbol{x}^{(t+1)}$ (a solution vector) from an existing solution $\boldsymbol{x}^{(t)}$. That, $\boldsymbol{x}^{(t+1)} = A(\boldsymbol{x}^{(t)}, \alpha)$.



Population-based algorithms use multiple, interacting paths.

Different algorithms

Different ways for generating new solutions!

Main Problems with Traditional Algorithms

What's Wrong with Traditional Algorithms?

- Traditional algorithms are mostly local search, thus cannot guarantee global optimality (except for linear and convex optimization).
- Results often depend on the initial starting points (except linear and convex problems). Methods tend to be problem-specific (e.g., k-opt, branch and bound).
- Struggle to cope problems with discontinuity.

Nature-Inspired Optimization Algorithms

Heuristic or metaheuristic algorithms tend to be a global optimizer so as to

- Increase the probability of finding the global optimality (as a global search tool),
- Solve a wider class of problems (treating them as a black-box),
- Draw inspiration from Nature (e.g., swarm intelligence).

But they can be potentially more computationally expensive.

Nature-Inspired Optimization Algorithms

- Genetic algorithms (1960s/1970s), evolutionary strategy (Rechenberg & Swefel 1960s), evolutionary programming (Fogel et al. 1960s).
- Simulated annealing (Kirkpatrick et al. 1983), Tabu search (Glover 1980s), ant colony optimization (Dorigo 1992), genetic programming (Koza 1992), particle swarm optimization (Kennedy & Eberhart 1995), differential evolution (Storn & Price 1996/1997), harmony search (Geem et al. 2001), artificial bee colony (Karaboga, 2005).
- Firefly algorithm (Yang 2008), cuckoo search (Yang & Deb 2009), bat algorithm (Yang, 2010), flower pollination algorithm (2012), ...

For details, please refer to:

- Xin-She Yang, *Nature-Inspired Optimization Algorithms*, Elsevier, (2014).
- Xin-She Yang, Optimization Techniques and Applications with Examples, Wiley, (2018).

PSO

Particle Swarm Optimization (Kennedy and Eberhart, 1995)

Swarming Starlings (YouTube Video)

$$\boldsymbol{v}_i^{t+1} = \boldsymbol{v}_i^t + \alpha\epsilon_1(\boldsymbol{g}^* - \boldsymbol{x}_i^t) + \beta\epsilon_2(\boldsymbol{x}_i^* - \boldsymbol{x}_i^t),$$

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \boldsymbol{v}_i^{t+1}.$$

$\alpha, \beta =$ learning parameters, ϵ_1, ϵ_2=random numbers.

PSO Demo and Disadvantages

The updating equations in PSO form a linear dynamical system:

$$\begin{pmatrix} \boldsymbol{x}_i \\ \\ \boldsymbol{v}_i \end{pmatrix}^{t+1} = \begin{pmatrix} 1 & 1 \\ \\ -(\alpha\epsilon_1 + \beta\epsilon_2) & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_i \\ \\ \boldsymbol{v}_i \end{pmatrix}^{t} + \begin{pmatrix} 0 \\ \\ \alpha\epsilon_1 \boldsymbol{g}^* + \beta\epsilon_2 \boldsymbol{x}_i^* \end{pmatrix}.$$

PSO Demo

Premature convergence

Quite efficient, but it can have premature convergence.

Firefly Algorithm

Firefly Video (YouTube)

Firefly Behaviour and Idealization (Yang, 2008)

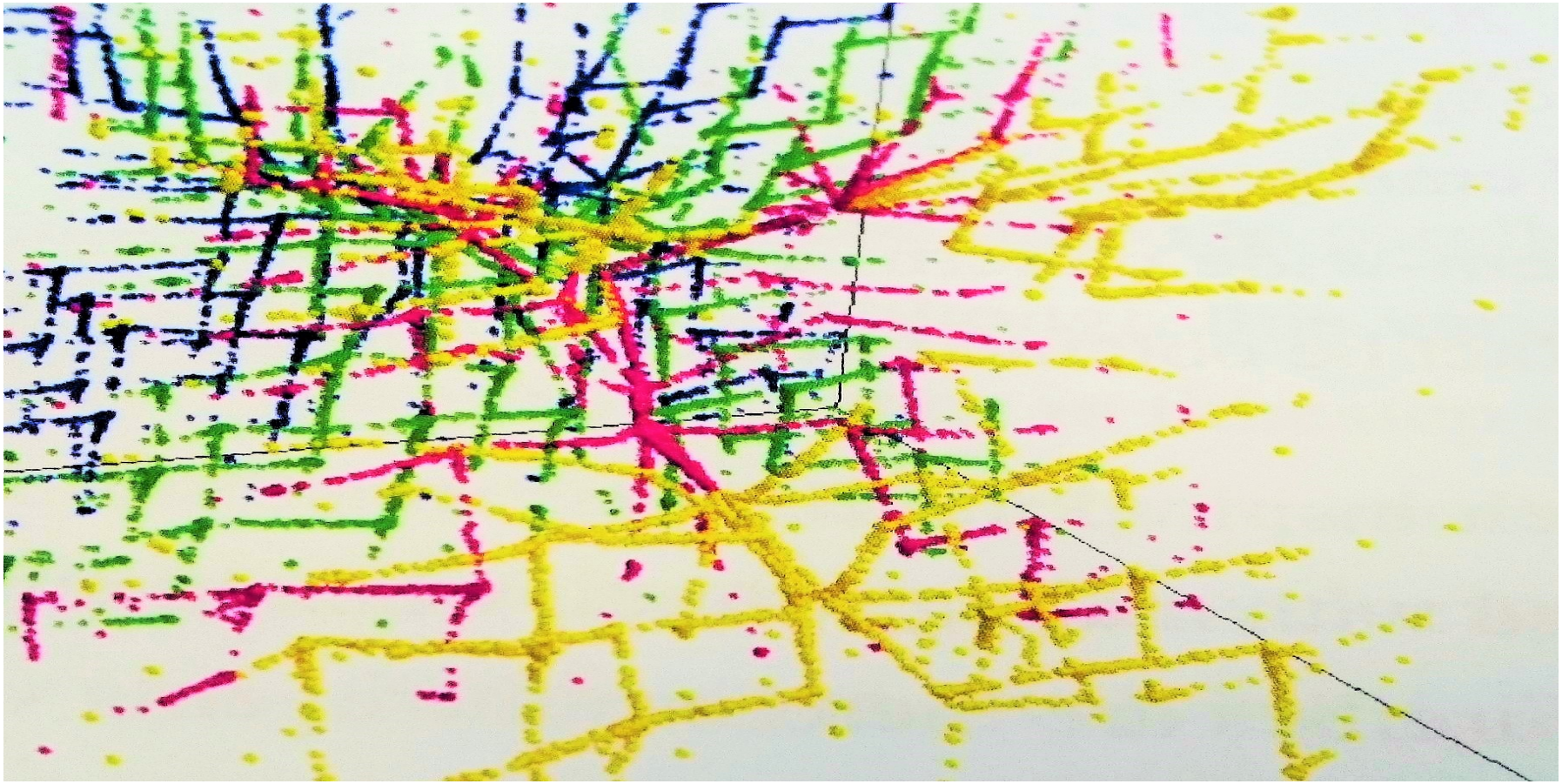
- Fireflies are unisex and brightness varies with distance.
- Less bright ones will be attracted to bright ones.
- If no brighter firefly can be seen, a firefly will move randomly.

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \beta_0 e^{-\gamma r_{ij}^2}(\boldsymbol{x}_j - \boldsymbol{x}_i) + \alpha\, \epsilon_i^t.$$

- ◇ The objective landscape maps to a light-based landscape, and fireflies swarm into the brightest points/regions.
- ◇ There is no $g*$, therefore, there is no leader. Also, a highly nonlinear iterative system, so subdivision into multiswarms is possible.

FA Demo and Advantages

Fireflies can take fractal-like search paths (sparse paths, but large coverage in the search space). E.g., 3D Rosenbrock function (Husselmann, 2014).



Why is FA so efficient?

Advantages of Firefly Algorithm over PSO

- Automatically subdivide the whole population into subgroups, and each subgroup swarms around a local mode/optimum.
- Control modes/ranges by varying γ.
- Control randomization by tuning parameters such as α.
- Suitable for multimodal, nonlinear, global optimization problems.

Firefly algorithm is Not PSO

Main differences

- FA uses a nonlinear attraction mechanism (inverse-quare law plus exponential decay). PSO mechanism is simply linear $(x_i^t - g^*)$.
- The population in the FA can subdivide into subgroups and thus can form multi-swarms automatically (PSO cannot).
- The standard FA does not use g^* (though PSO uses g^*).
  $\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \beta_0 e^{-\gamma r_{ij}^2}(\boldsymbol{x}_j - \boldsymbol{x}_i) + \alpha\, \epsilon_i^t.$
- FA can find multiple optimal solutions simultaneously (PSO cannot).
- FA has a fractal-like search structure (PSO does not).

FA Variants for specific applications:

• Continuous optimization, • Mixed integer programming, • Combinatorial optimization such as TSP, • Multiobjective FA, • FA for image processing, ...

Cuckoo Search Algorithm (Yang and Deb, 2009)

Cuckoo brood parasitism

- 59 cuckoo species (among 141 cuckoo species) engage the so-called obligate reproduction parasitism strategy.
- Cuckoos lay eggs in the nests of host birds (such as warblers) and let host birds raise their chicks.
- Eggs may be discovered/abandoned with a probability ($p_a \approx 0.25$).
- Co-evolutionary arms race between cuckoo species and host species.

Cuckoo Behaviour (BBC Video)

Cuckoo Search

Local random walk:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + s \otimes H(p_a - \epsilon) \otimes (\boldsymbol{x}_j^t - \boldsymbol{x}_k^t).$$

[$\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{x}_k$ are 3 different solutions, $H(u)$ is a Heaviside function, ϵ is a random number drawn from a uniform distribution, and s is the step size.

Global random walk via Lévy flights:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \alpha L(s, \lambda), \quad L(s, \lambda) \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}}, \ (s \gg s_0).$$

Generation of new moves by Lévy flights, random walks and elitism.

Cuckoo Search Demo: Highly Efficient!

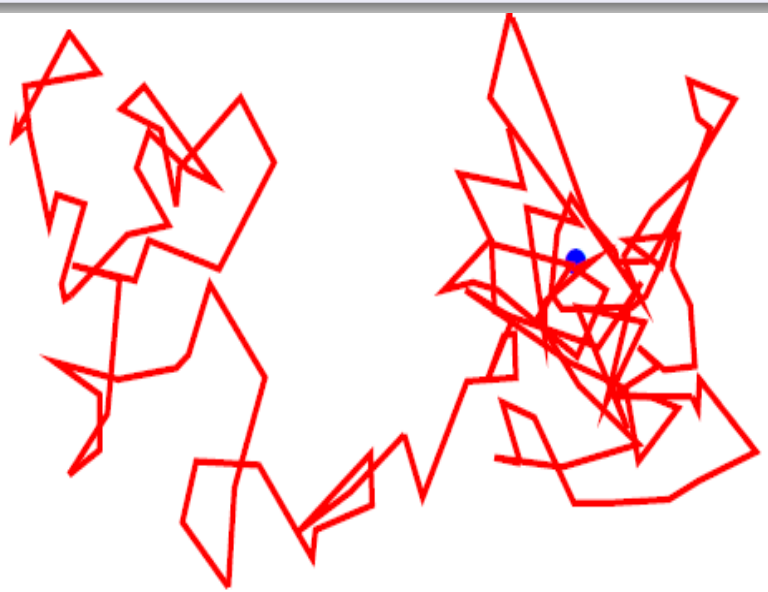
CS Demo

Efficient search with a focus

Mathematical Foundation for Cuckoo Search

Isotropic andom walks (diffusion)
Gaussian distribution

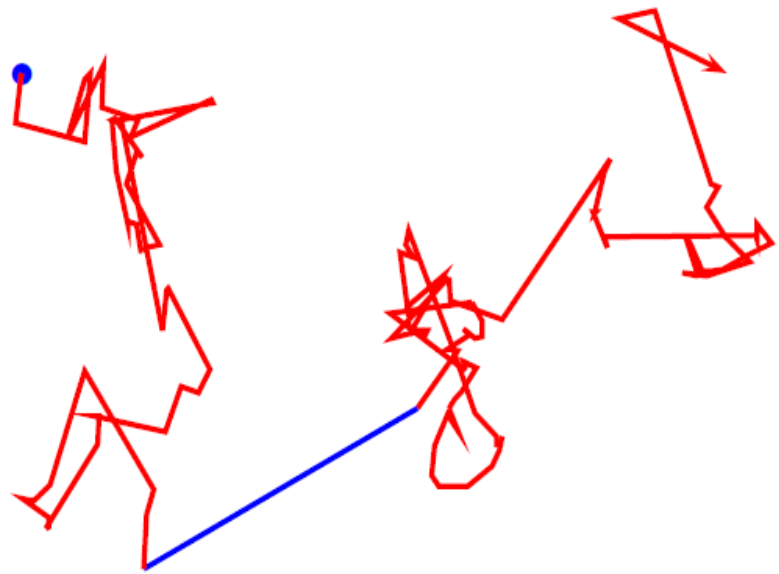
$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\Big[-\frac{(x-\mu)^2}{2\sigma^2}\Big],$$



Diffusion distance: $d \sim \sqrt{t}$

Lévy flights (superdiffusion)
Lévy distribution

$$L(x) = \frac{1}{\pi}\int_0^\infty \cos(tx)e^{-\alpha t^\lambda}\,dt.$$



$d \sim t^{(3-\lambda)/2}$ (for $1 \le \lambda \le 2$)

Other Algorithms

- Ant colony optimization (1992)
- Differential evolution (1997)
- Bee algorithms (1990s)
- Bat algorithm (2010)
- Flower pollination algorithm (2012)
- Memetic algorithm
- ... (many other algorithms) ...

Reviews

- X.S. Yang, et al., *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*, Elsevier, (2013).
- X.S. Yang, X. S., *Nature-Inspired Optimization Algorithms*, Elsevier, (2014).

Steps for Implementation

1. Problem Formulation: Proper formulation of an optimization problem may ease the solution process (e.g., L_2-norm, ideally convex or close to some known forms).
2. Choice of an Algorithm: Choose an efficient algorithm (or a possible set of algorithms) to deal with the optimization problem.
   [But such algorithm may not exist.]
3. Handling Constraints: Handling constraints properly (e.g., penalty method, dimension reduction, evolutionary approaches).
4. Implementation: Vectorization and parallelization when necessary.
5. Validation and Predictions: Test and validate the implementation thoroughly using a diverse range of benchmarks before proceeding to solve other types of problems.

Nature-inspired algorithms

For nature-inspired algorithms or swarm intelligence-based algorithms, the following suggestions should be considered:

- Population size n: A good population with sufficient diversity. Not too small, not too big. e.g., $n = 20$ to 100 (up to 500).
- Carry out a detailed parametric study (so as to get sense of the parameter ranges). [Ideally, if possible, with some theoretical basis.]
- Initialization can be important. Think it as part of Monte Carlo sampling.
- Tests and multiple independent runs. Do not select results.
- Compare fairly (e.g., the same number of function evaluations).

Do they really work?

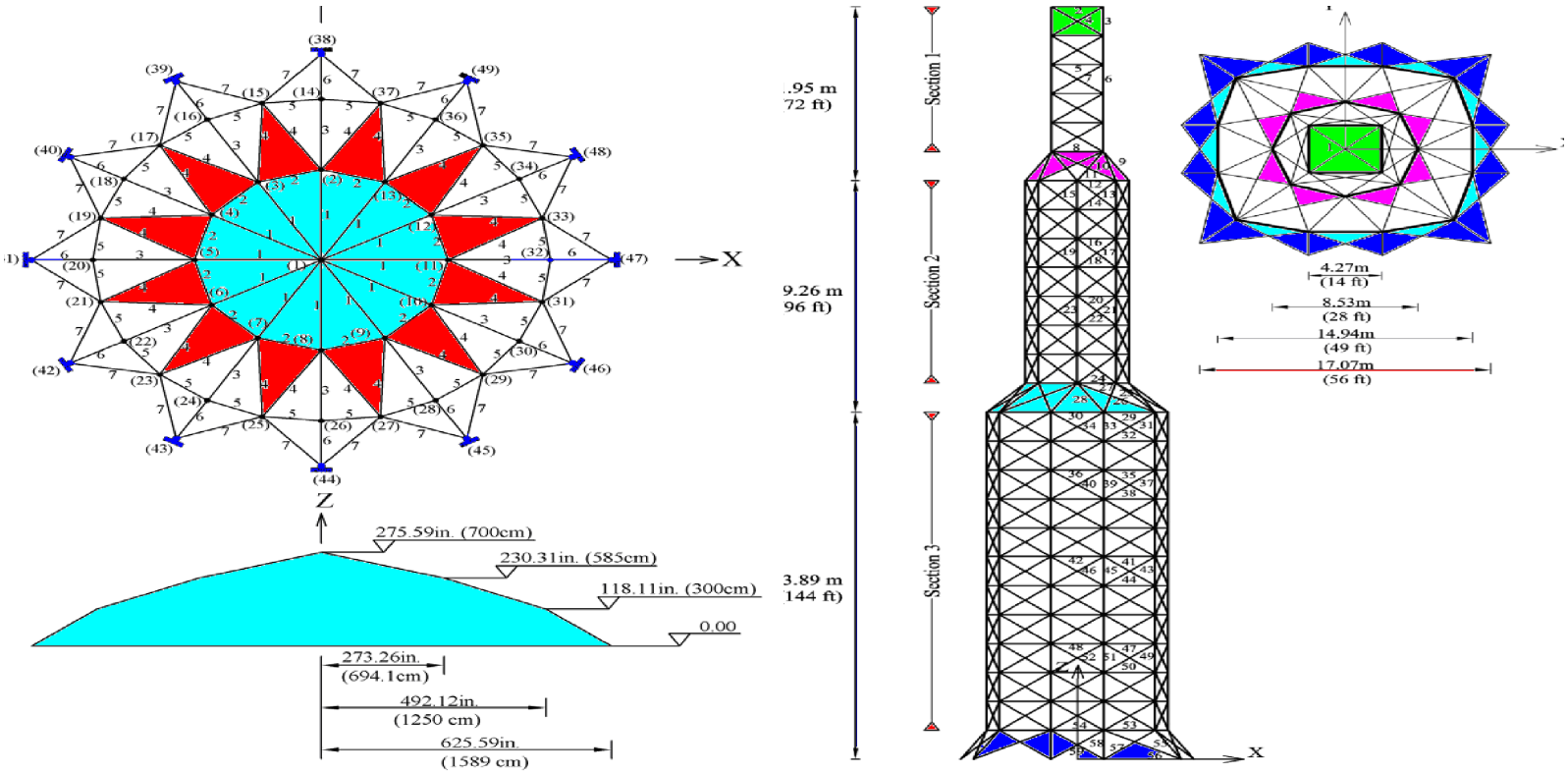
These algorithms seem to be simple and easy to implement,

can they really solve any difficult optimization problems in practice?

They can indeed work surprisingly well in practice.

Dome and Tower Designs (Example 1)

Large-scale real-world applications. Objective: to minimize steel costs.



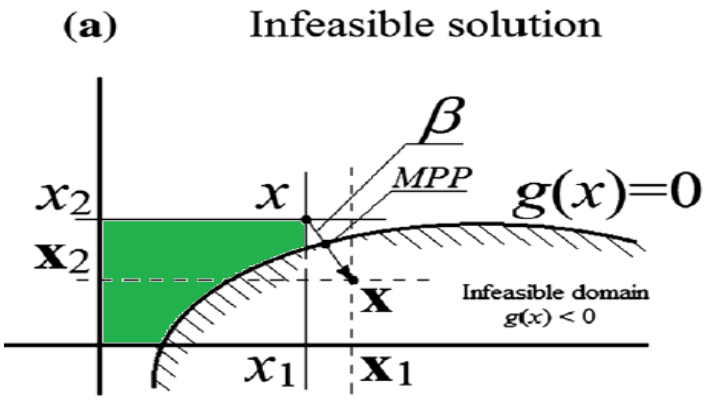
120-bar dome: Divided into 7 groups, 120 design elements, about 200 constraints.

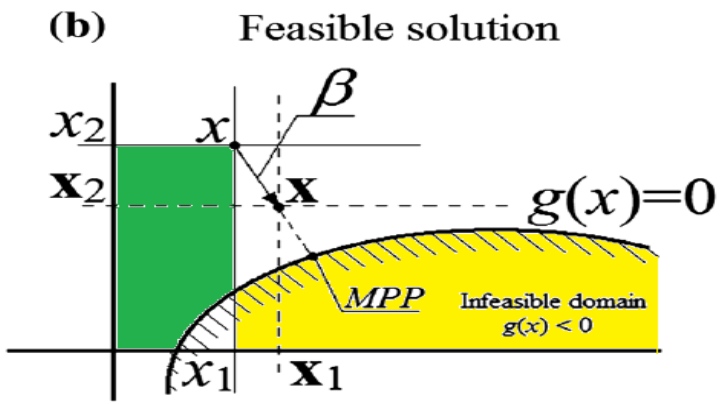
26-storey tower: 942 design elements, 244 nodal links, $>$ 4000 nonlinear constraints.

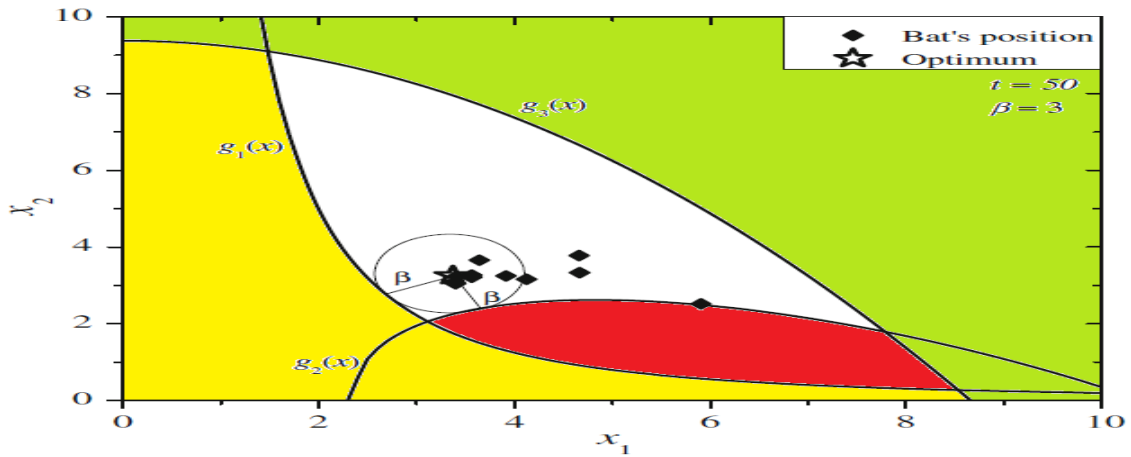
X.-S. Yang and A. H. Gandomi, Bat algorithm: a novel approach for global engineering optimization, Engineering Computations, 29(5), 464-483 (2012).

Reliability-Based Design Optimization (Example 2)

Design optimization subject to stochastic/probabilistic constraints.



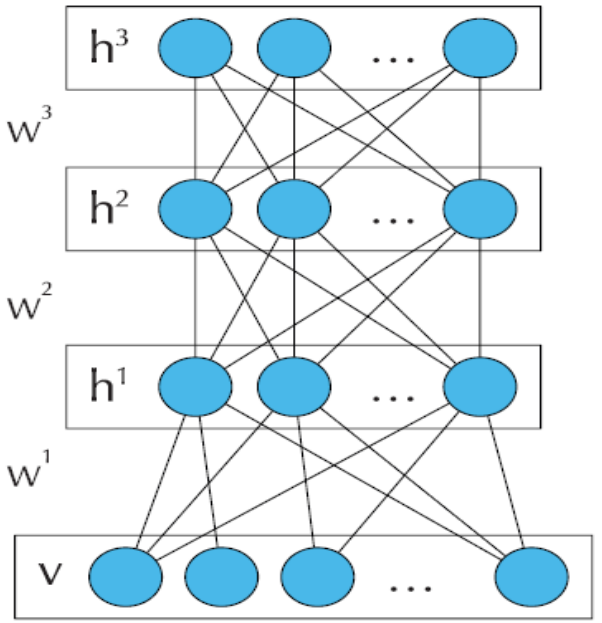




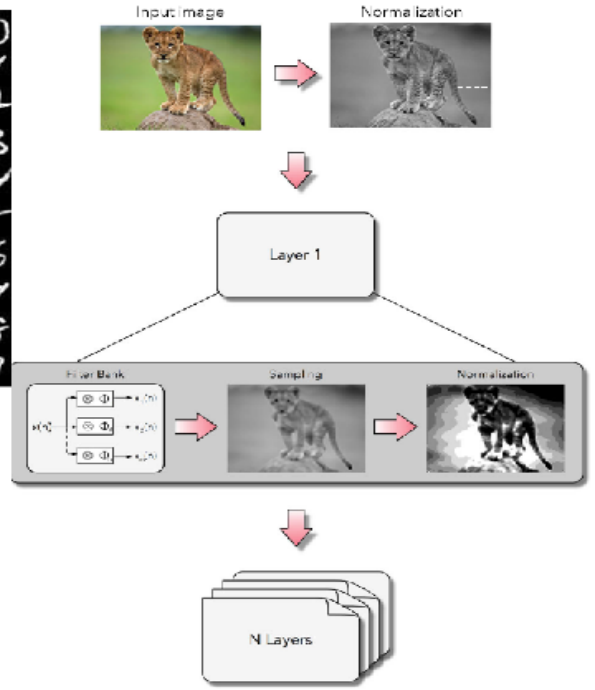
Chakri, Yang, Khelif, Benouaret, (2017). Reliability-based design optimization using the directional bat algorithm, Neural Computing and Applications, 1-22 (2017).

Deep Belief Networks Fine Tuning (Example 3)

Tuning hyper-parameters in deep belief networks by the bat algorithm (BA) achieved the best accuracy.



The DBN architecture used in this work.



Papa, Rosa, Pereira, Yang, Quaternion-based deep belief networks fine tuning, Applied Soft Computing, 60, 328–335 (2017).

Hyperspectral Band Selection (Example 4)

Data dimensionality reduction was achieved by the bat algorithm in combination with the optimal path forest.

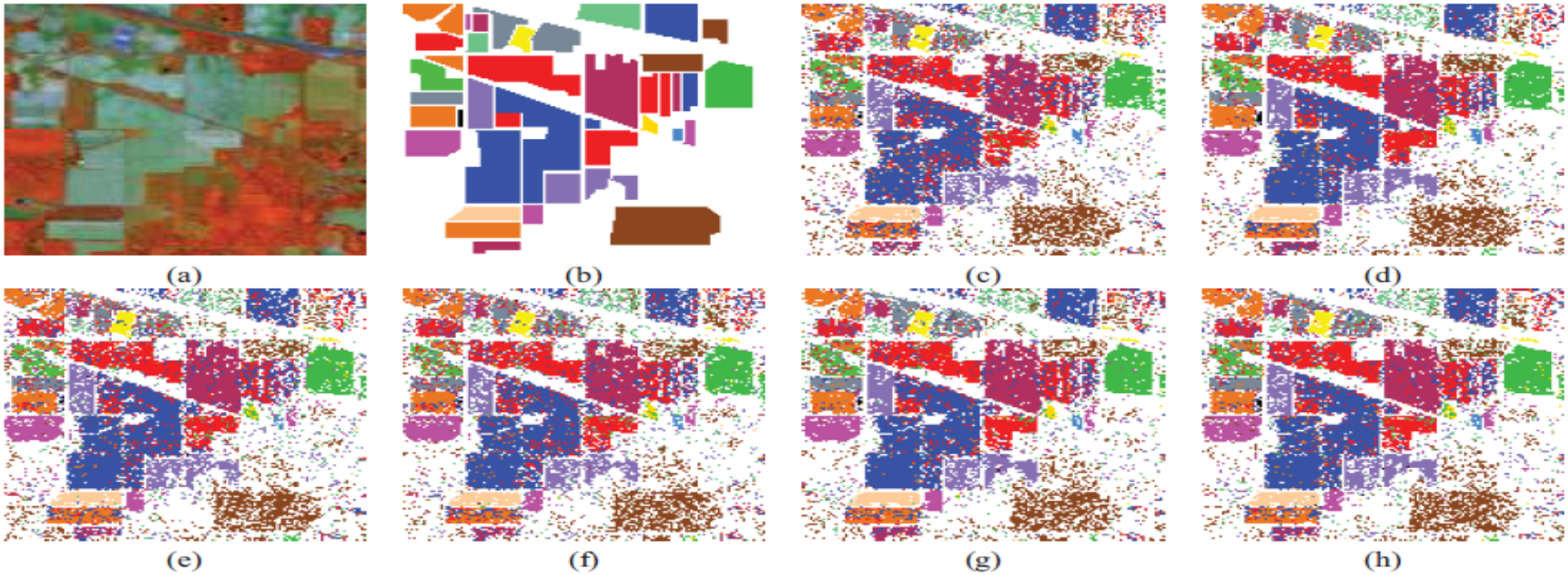
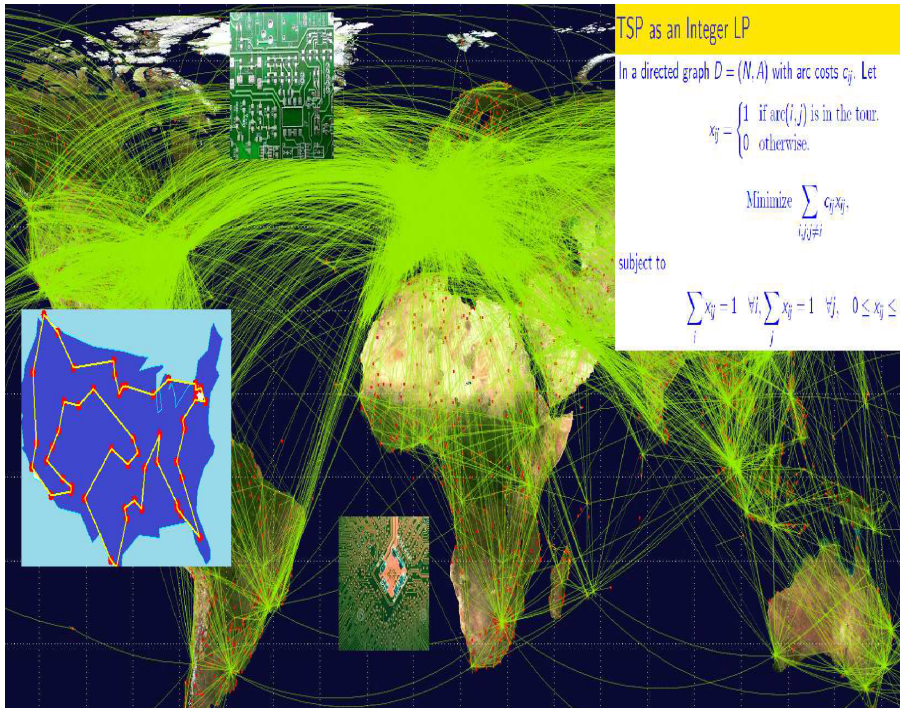


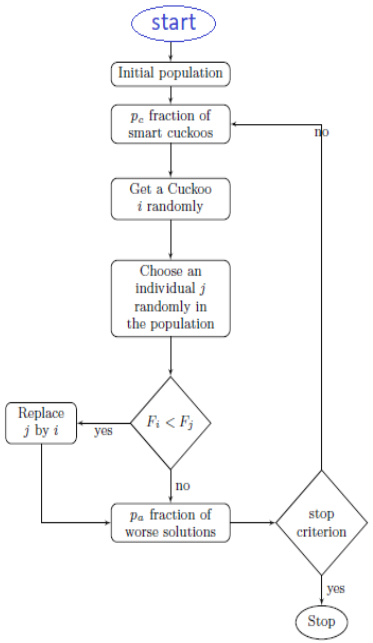
Fig. Indian Pines (a) composed image (R50 G133 B28), (b) ground truth and classified images using (c) OPF, (d) BA, (e) FA, (f) GSA, (g) HS, and (h) PSO.

Nakamura, Fonseca, Santos, Torres, Yang, Papa, (2014). Nature-inspired framework for hyperspectral band selection, IEEE Trans. Geoscience and Remote Sensing, 52(4), 2126-2137.

Travel Salesman Problem (TSP) (Example 5)

TSP benchmarks have been solved by Cuckoo Search Algorithm, which leads to the same or better results in 2/3 of the problem instances.





The flowchart of the RKCS algorithm

Table 1 Parameter settings for RKCS algorithm

Parameter	Value	Meaning
n	30	Population size
p_c	0.6	Portion of smart cuckoos
p_a	0.2	Portion of bad solutions
$MaxGen$	500	Maximum number of iterations
α	0.01	Step size
λ	1	Index

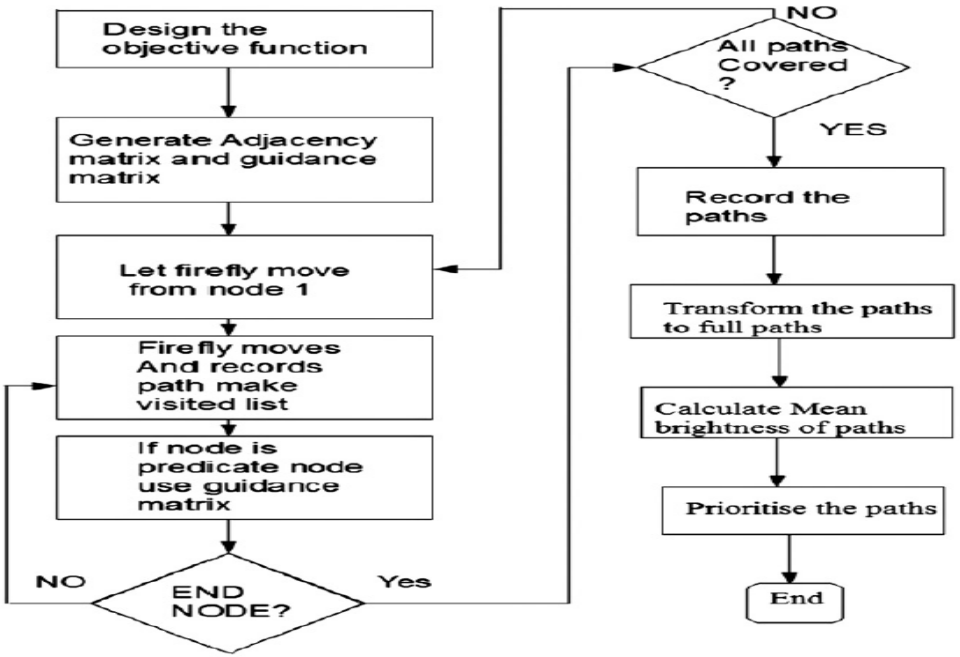
Table 2 Results of random-key cuckoo search for the travelling salesman problem

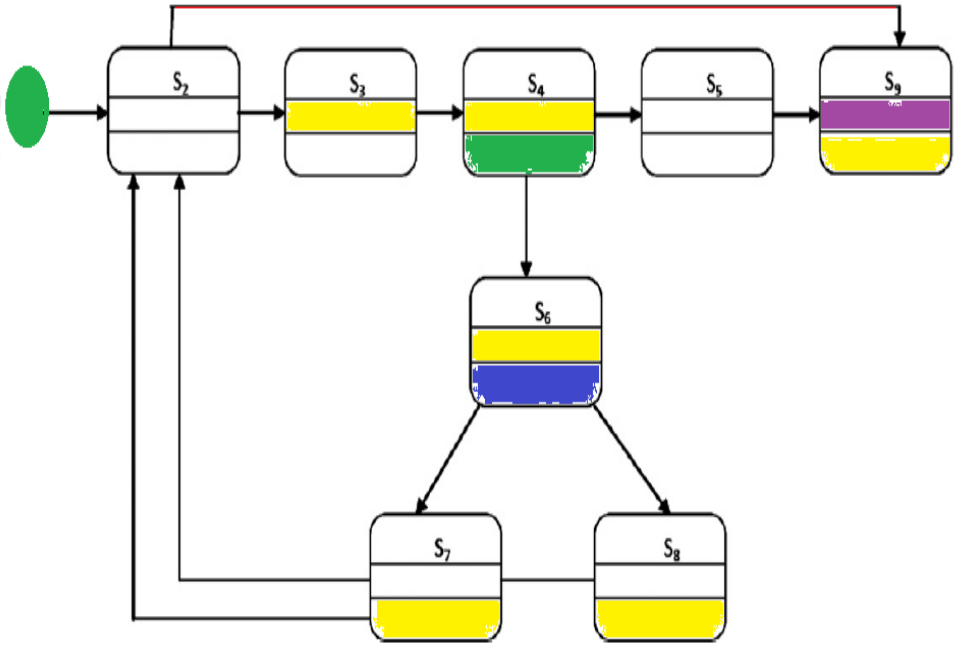
Instance(opt)	Best	Average	Worst
eil51(426)	**426**	426.9	430
berlin52(7542)	**7542**	7542	7542
st70(675)	**675**	677.3	684
pr76(108159)	**108159**	108202	109085
eil76(538)	**538**	539.1	541
kroA100(21282)	**21282**	21289.65	21343
eil101(629)	**629**	631.1	636
bier127(118282)	**118282**	118798.1	120773
pr136(96772)	97046	97708.9	98936
pr144(58537)	**58537**	58554.45	58607
ch130(6110)	6126	6163.3	6210

A. Ouaarab, B. Ahiod, X. S. Yang, Discrete cuckoo search algorithm for the travelling salesman problem, Neural Computing and Applications, 24, 1659-1669 (2014).

Software Test Path Generation (Example 6)

Software testing can take up to 60% of the overall development costs. Optimal and independent test paths are generated by the firefly algorithm.





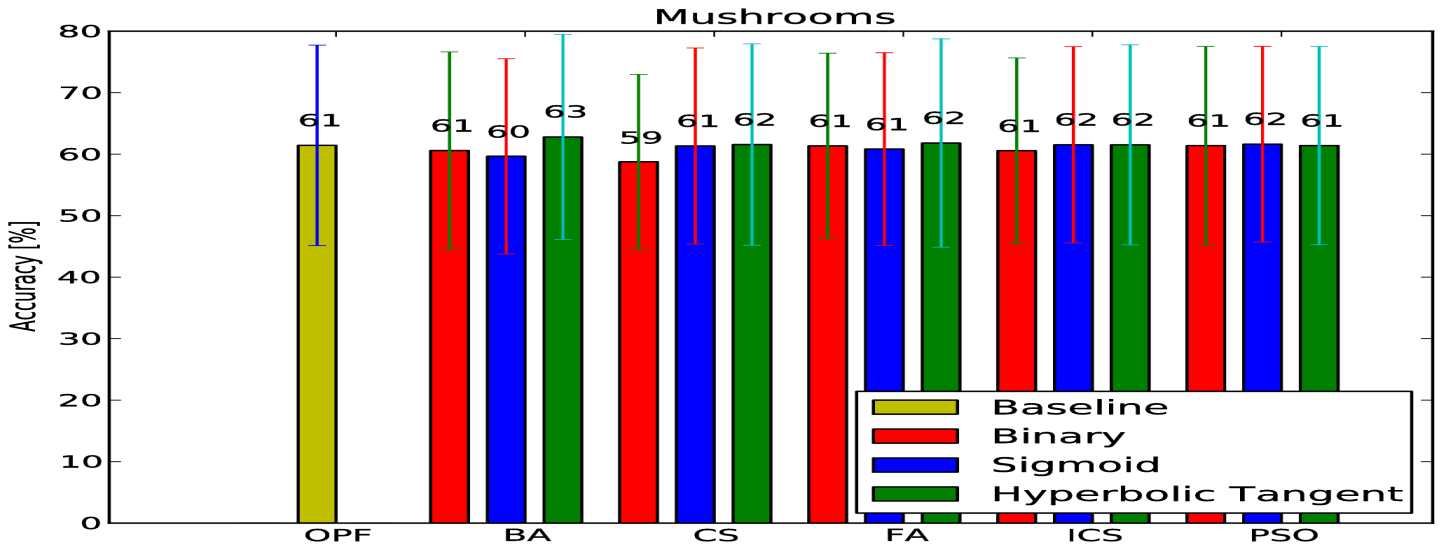
E.g., 100 'if ... then' can lead to $2^{100} \approx 1.27 \times 10^{30}$ possible paths.

P. R. Srivatsava, B. Mallikarjun, X. S. Yang, Optimal test sequence generation using firefly algorithm, *Swarm and Evolutionary Computation*, 8 (1), 44-53 (2013).

Feature Selection (Example 7)

Bat algorithm and cuckoo search are very competitive (Papa et al, 2013; 2014). Also effective at dimension reduction (in terms of number of key features).

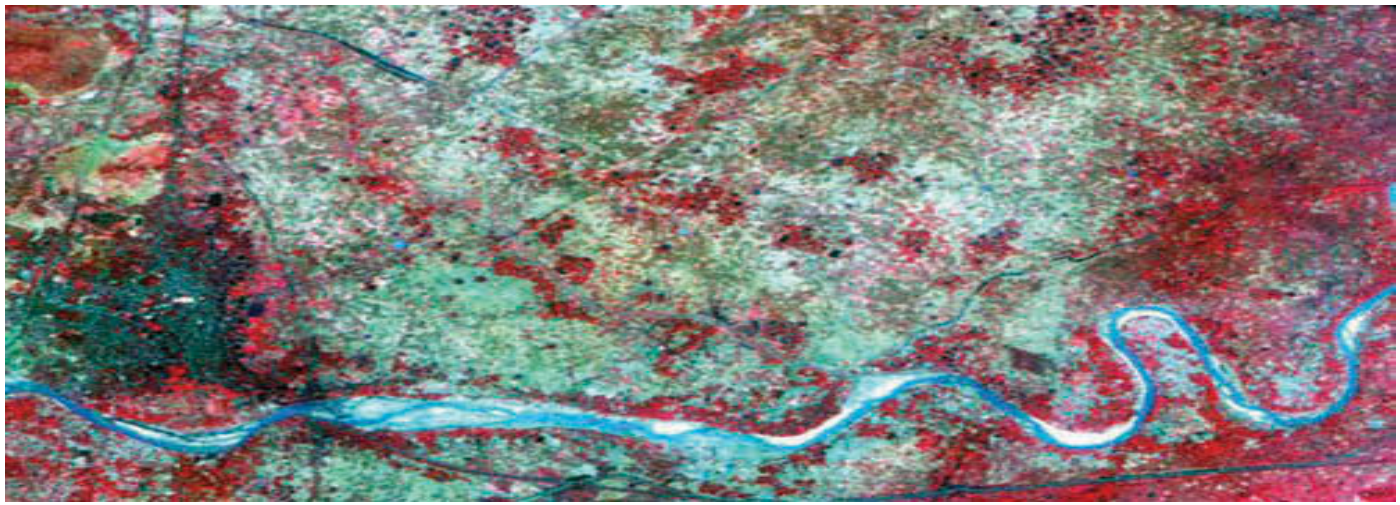
Dataset	# samples	#features	#classes
Diabetes	768	8	2
DNA	2000	180	3
Mushrooms	8124	112	2



L.A.M Pereira et al., A binary cuckoo search and its application for feature selection, in: *Cuckoo Search and Firefly Algorithm* (Edited by X.S. Yang), Springer, 2014.

Satellite Image Processing (Example 8)

Among 14 different methods compared, firefly algorithm (FA) obtained the best results with the least computing time.

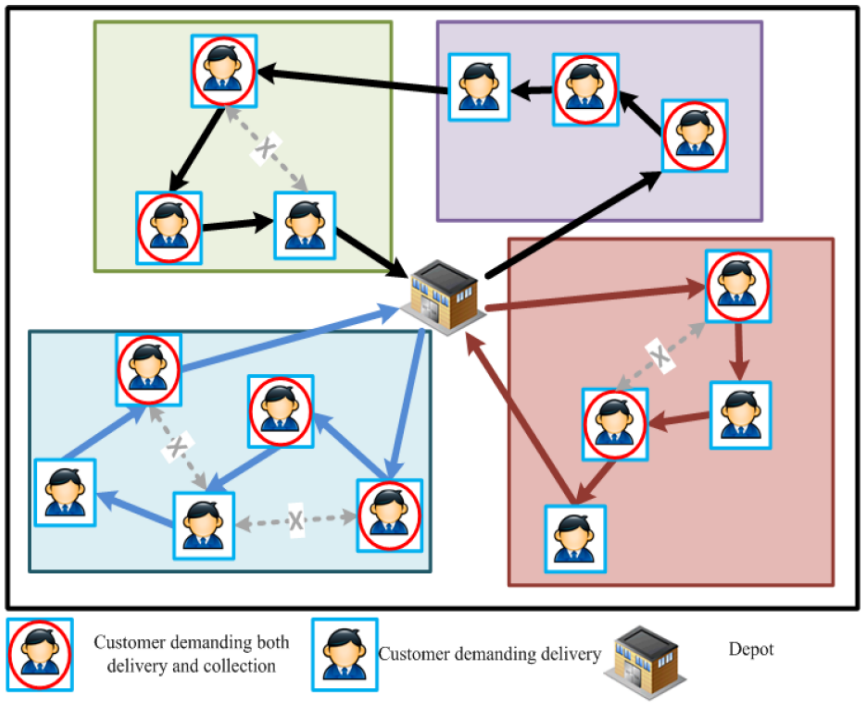


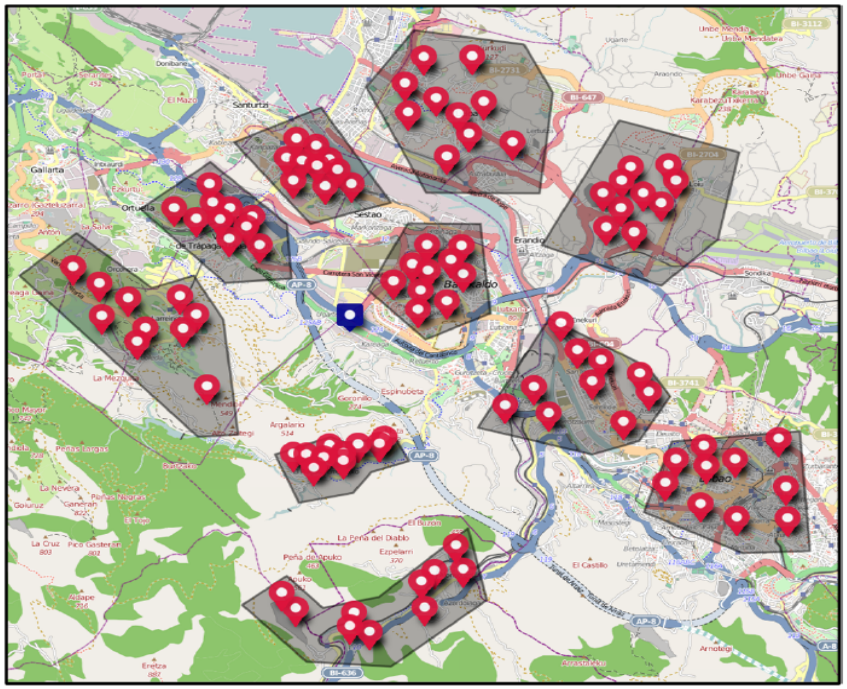
Algorithms	Optimal	Worst	Mean
GA	0.8044	0.7528	0.7824
PSO	0.8057	0.8009	0.8051
FA	0.8126	0.8019	0.8076

J. Senthilnath, X. S. Yang, J. A. Benediktsson, Automatic registration of multi-temporal remote sensing images based on nature-inspired techniques, *Int. J. Image and Data Fusion*, 5 (4), 263-284 (2014).

Vehicle Routing with Real Traffic Information (Example 9)

Optimal deployment of vehicles to distribute goods (e.g., medical supplies, newspapers) and collect recyclable goods, subject to dynamic real-time traffic. The discrete firefly algorithm (DFA) obtained the best results.

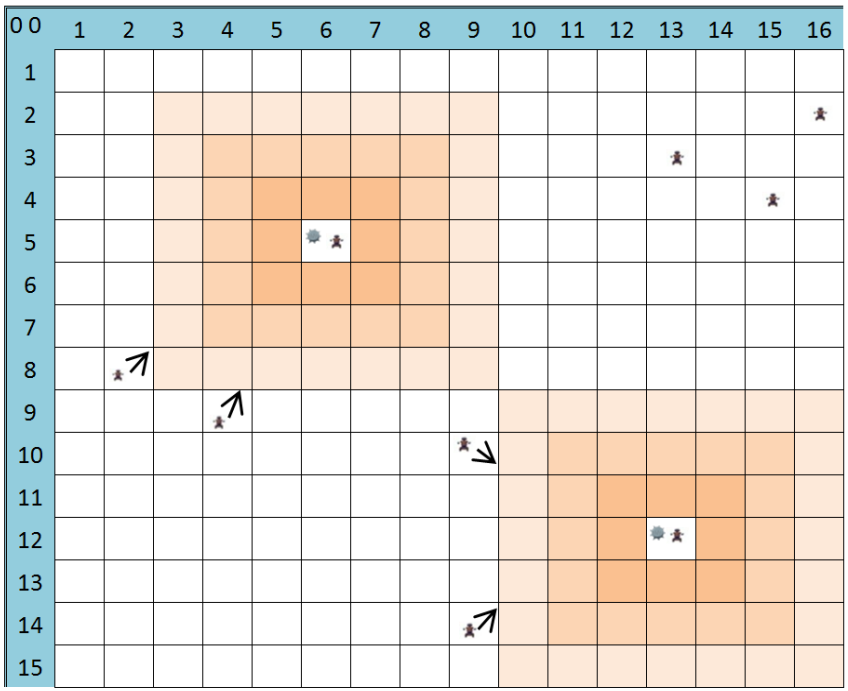


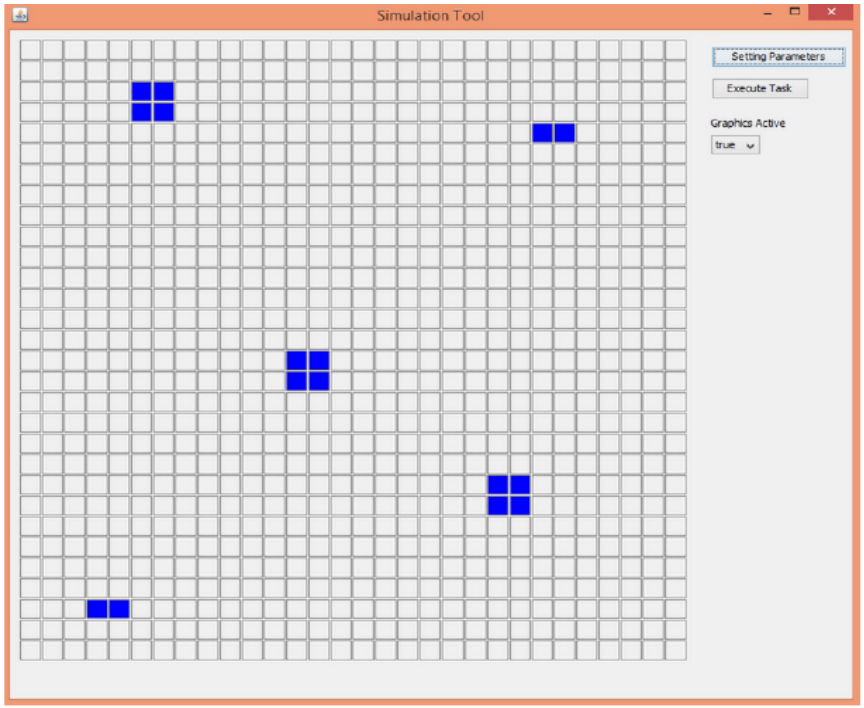


E. Osaba, X.S. Yang, F. Diaz, et al., A discrete firefly algorithm to solve a rich vehicle routing problem with recycling policy, Soft Computing, 21(18), 5295-5308 (2017).

Swarm Robots (Example 10)

A swarm of mobile robots with wireless distributed protocol and no central decision explore the unknown targets in a vast region so as to find all the targets and handle/remove these targets safely and quickly.





F, De Rango, N. Palmieri, X.S. Yang, S. Marano, Swarm robotics in wireless distribution protocol design for coordinating robots involved in cooperative tasks, Soft Computing, 22 (13), 4251-4266 (2018).

Algorithm analysis

- How do we analyze such metaheuristic/stochastic algorithms? [Any unified theory or framework?]
- How quickly can an algorithm converge? [Rate of convergence, accuracy?]
- How stable is an algorithm? [Stability, noisy?]

Algorithmic Characteristics (7 different perspectives)

- Key components and roles: genetic operators (e.g., crossover, mutation, elitism), procedure or equations, and random walks.
- Exploration and exploitation: diversification and intensification, diversity of solutions, how to use information during iterations.
- Multi-agent systems: local rules, interacting agents exploring the search space.
- Statistical measures: mean, variance, accuracy, etc.
- Stability analysis: stability of iterations, and parameter ranges.
- Convergence behaviour: the rate of convergence, under what conditions?
- Robustness analysis: sensitivity, changes of optimal solutions under uncertainty or subject to perturbations or noise.

Mathematical Analysis of Algorithms

Analyzing nature-inspired algorithms mathematically (at least 8 ways)

- Iterative system: Fixed point theory. If $\boldsymbol{x}_{k+1} = A(\boldsymbol{x}_k, \alpha)$, try to show that $\boldsymbol{x}_k \to \boldsymbol{x}_*$ (fixed solution vector) when $k \to \infty$.
- Dynamic system: Attractors and equilibrium. Writing the algorithm in terms of a dynamic system $\boldsymbol{y}_k = \boldsymbol{A}\boldsymbol{y}_k$ where $\boldsymbol{y}$ is a state vector in terms of $\boldsymbol{x}_k$ and other quantities such as velocity $\boldsymbol{v}_k$. Analyze the eigenvalues of $\boldsymbol{A}$ and the conditions for stability.
- Self-organized system: Attempt to compare with the conditions for self-organization in complex systems at far-from equilibrium so as to understand the mechanisms in algorithms and structure/pattern formation. For example, selection in algorithms acts as a driving force for system to evolve and select the states/solutions.
- Markov chain Monte Carlo: Analyze an algorithm using the Markovian framework, considering $\boldsymbol{x}_k$ as the current state and the way of generating new solutions as the transition probability. Try to get the conditions for constructing proper Markov chains from the iteration sequence x_k $(k = 1, 2, ...)$ and conditions for convergence.

- Bayesian framework: Using Bayes theory to figure out the relationship of *prior* probability distributions for initialization and the randomization in the algorithm, try to analyze the posterior distributions of the population and then try to understand the algorithm behaviour.
- Filter theory: Using the theory of filters to understand how different solutions evolve. Some solutions with higher variations will gradually be filtered out, while other solutions will smooth out or remain in the population.
- Computational Complexity: The analysis of algorithmic complexity and see if it can be linked to the time complexity of the problems to be solved. Try to resolve the mystery why simple algorithms can potentially solve hard problems (including NP-hard problems).
- Multidisciplinary perspectives: It seems that it is unlikely to understand algorithms from a single perspective. A multiple disciplinary approach is needed to gain insight into algorithms from different angles and perspectives.

No free lunch theorems

No best algorithms for *all* problems.

Example 1: Genetic Algorithm

For a binary genetic algorithm with $p_0 = 0.5$ with n chromosomes of length m, the probability of premature convergence at any time/iteration t is

$$P(t, m) = \left[1 - \frac{6p_0(1-p_0)}{n}\left(1 - \frac{2}{n}\right)^t\right]^m.$$

For a population of $n = 40$, $m = 100$, $t = 100$ generations, we have

$$P(t, m) = \left[-1\frac{6 \times 0.5(1-0.5)}{40}(1 - \frac{2}{40})^{100}\right]^{100} \approx 0.978.$$

For a given convergence probability ζ, the number of iterations $t(\zeta)$ needed is

$$t(\zeta) \leq \left[\frac{\ln(1-\zeta)}{\ln\left\{1 - \min[(1-\mu)^{nL}, \mu^{nL}]\right\}}\right],$$

where μ =mutation rate, L =string length, and n =population size.

Villalobos-Arias et al., Asymptoptic convergence of metaheuristics for multiobjective optimization problems, Soft Computing, 10(11), 1001-5 (2005).

Example 2: Convergence Analysis of the Bat Algorithm

Global convergence analysis of the Bat Algorithm as a dynamical system.

The main equations of the bat algorithm can be simplified as

$$\begin{cases} f_i = f_{\min} + (f_{\max} - f_{\min})\beta, \\ \boldsymbol{v}_i^{(k+1)} = \boldsymbol{v}_i^{(k)} + (g - \boldsymbol{x}_i^{(k)}) f_i, \\ \boldsymbol{x}_i^{(k+1)} = \boldsymbol{x}_i^{(k)} + \boldsymbol{v}_i^{(k+1)}. \end{cases}$$

We rewrite them more generally as

$$\boldsymbol{v}_i^{(k+1)} = \theta \boldsymbol{v}_i^{(k)} + \mu(g - \boldsymbol{x}_i^{(k)}), \quad \boldsymbol{x}_i^{(k+1)} = \boldsymbol{x}_i^{(k)} + \boldsymbol{v}_i^{(k+1)},$$

where θ and μ are parameters. This system can equivalently be written as

$$Y_{k+1} = AY_k + Mg, \quad Y_k = \begin{pmatrix} \boldsymbol{x}_i^{(k} \\ \boldsymbol{v}_i^{(k)} \end{pmatrix}, A = \begin{pmatrix} 1-\mu & \theta \\ -\mu & \theta \end{pmatrix}, \quad M = \begin{pmatrix} \mu \\ \mu \end{pmatrix}.$$

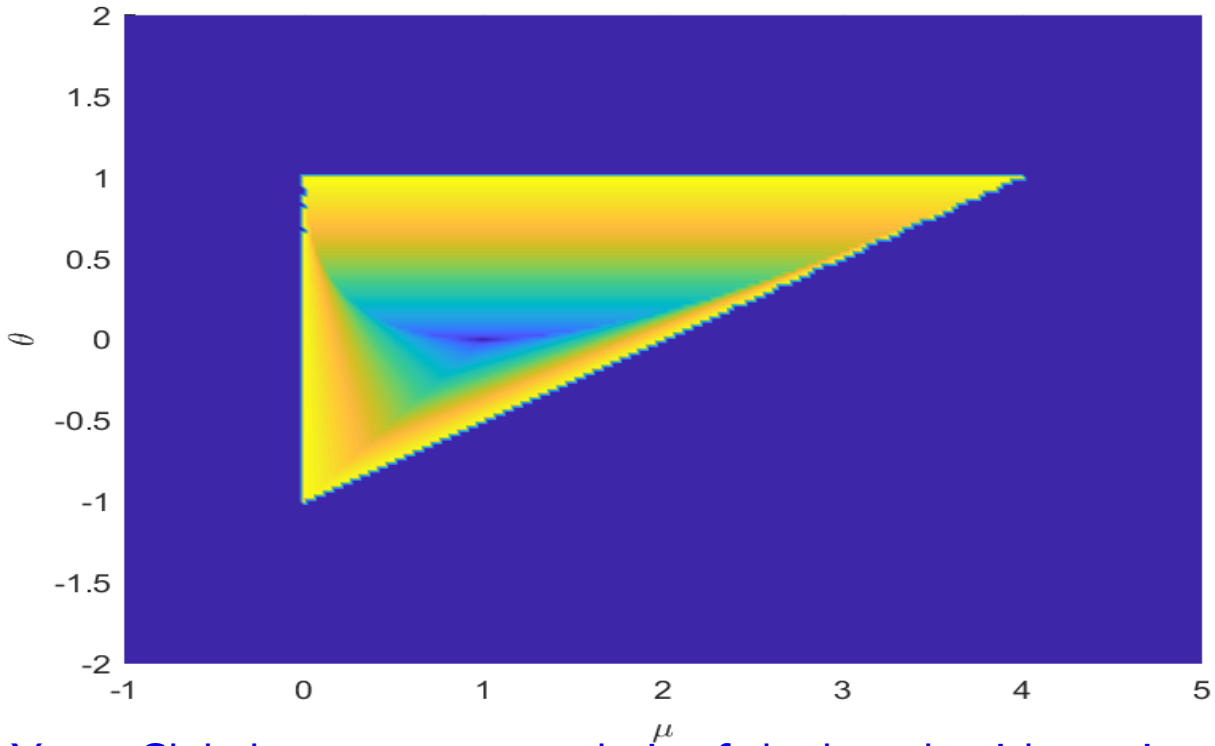
Lyapunov stability requires that $|\lambda| < 1$ and

$$\det \begin{pmatrix} 1-\mu-\lambda & \theta \\ -\mu & \theta - \lambda \end{pmatrix} = 0.$$

The parameter ranges become

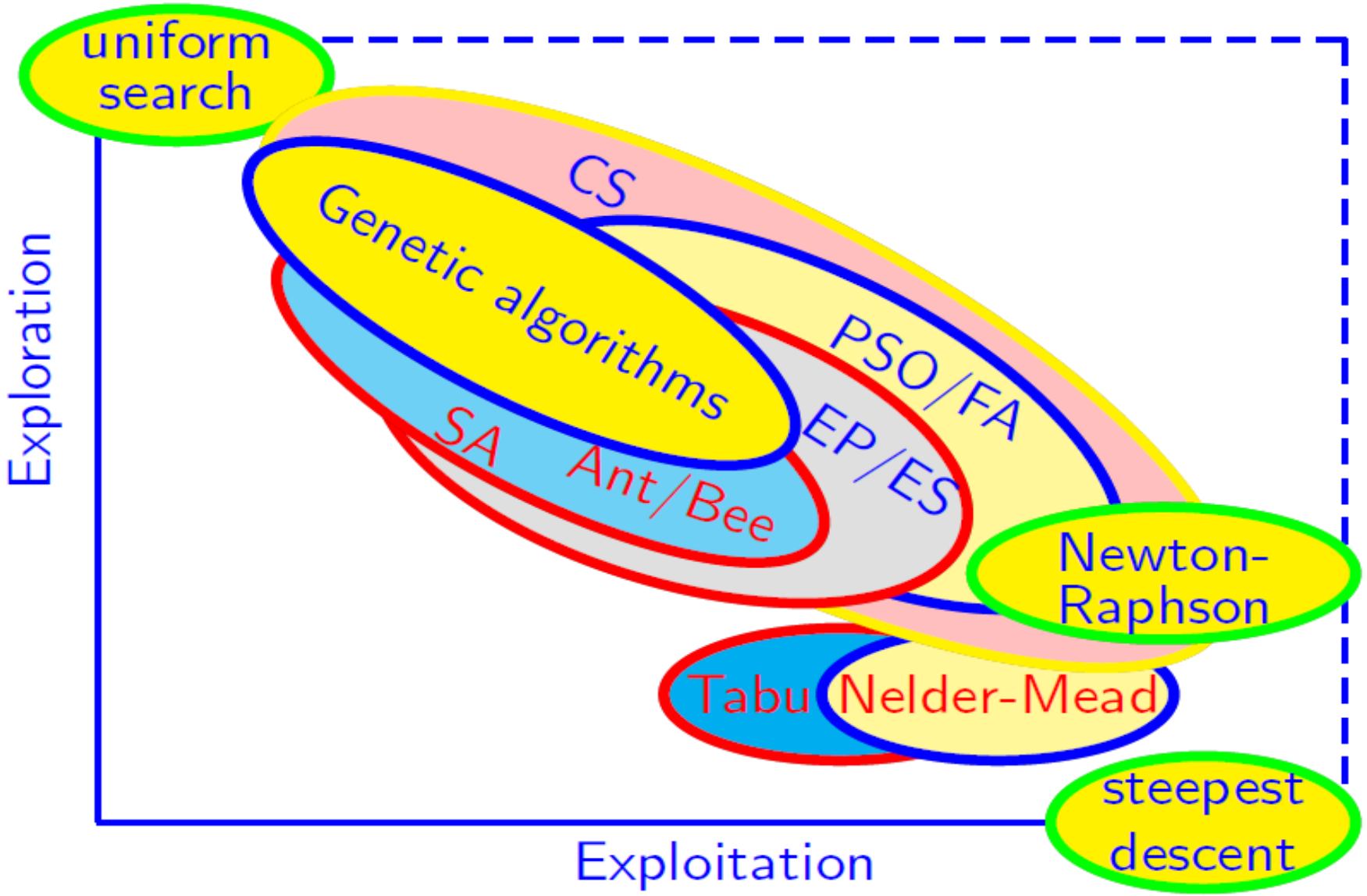
$$\begin{cases} -1 \le \theta \le +1, \\ \mu \ge 0, \\ 2\theta - \mu + 2 \ge 0, \end{cases}$$

which is a triangular region for stability.



Chen, Pan, He, Yang, Global convergence analysis of the bat algorithm using a Markovian framework and dynamical system theory, *Expert Systems and Applications*, 114(2018) 173-182.

Exploration and Exploitation

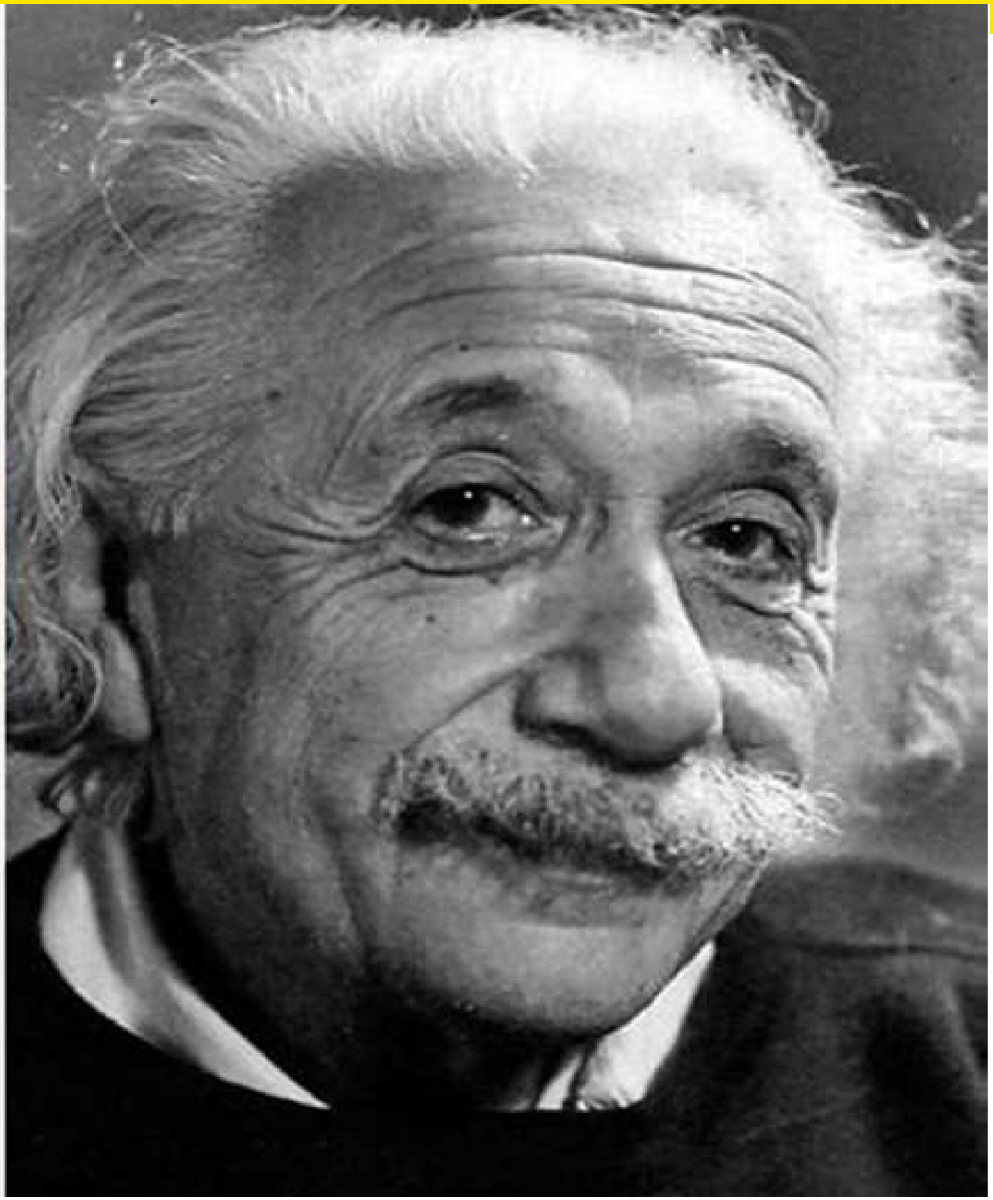


Future Research and Open Problems

Nature-inspired metaheuristic algorithms can be very efficient, even potentially for large-scale problems. There are more research opportunities:

- Mathematical analysis: Why such algorithms work? Under what conditions? How quickly can they converge?
- Parallelism and large-scale applications: How to parallel them effectively to solve large-scale problems in practice?
- Performance comparison: Gain insight by comparing different algorithms, and thus potentially select the most suitable algorithms for a given set of applications.
- Parameter tuning: Fine tuning of algorithms so as to perform optimally and ideally fast enough for real-world applications.
- Smart algorithms: What do you mean by 'smart'? Are they really smart/intelligent?

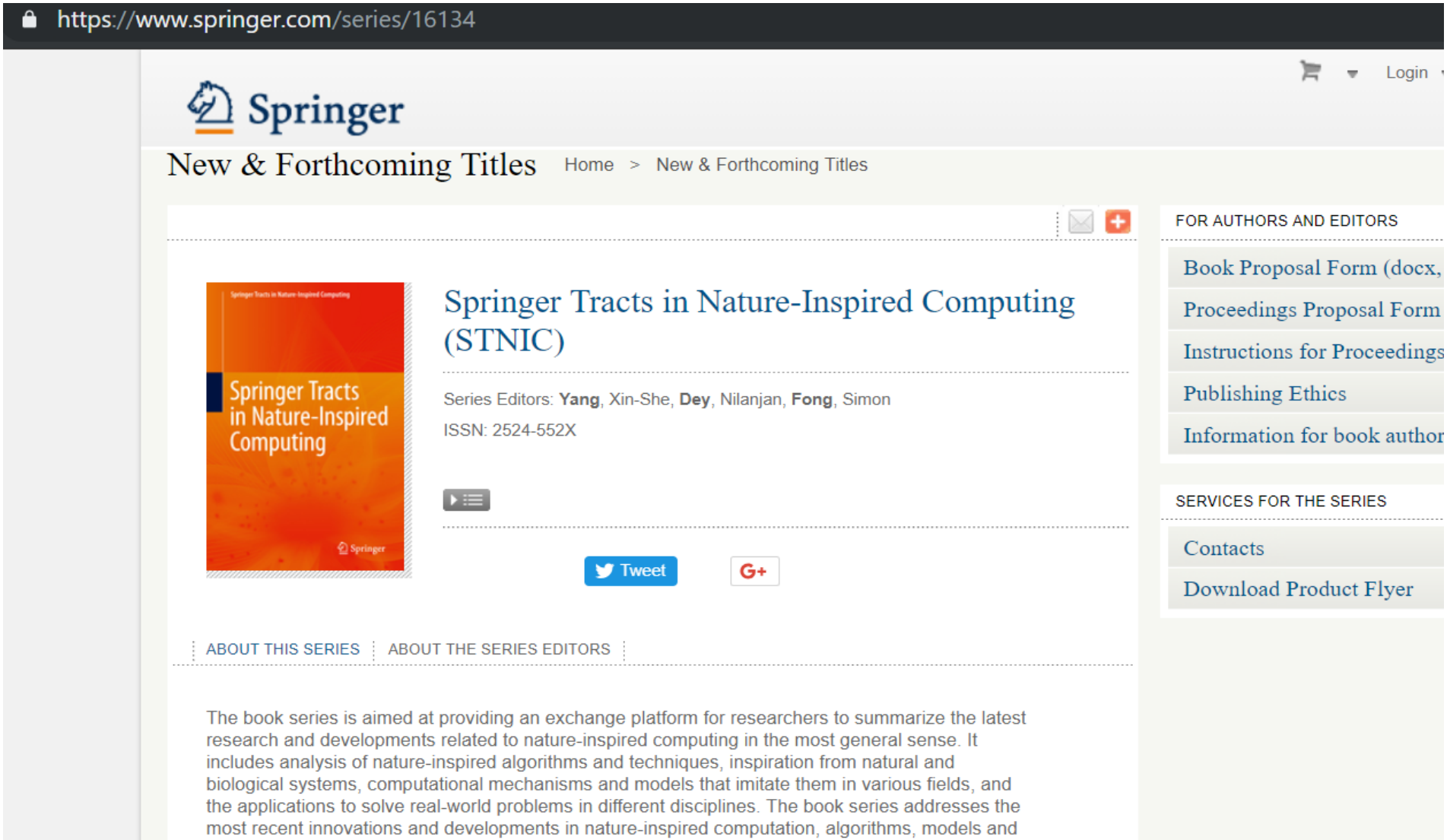
Not smart enough yet?



News and announcements...

Springer Tracts in Nature-Inspired Computing

http://www.springer.com/series/16134



Two Conferences

NICE2019 (of SSCI2019), Xiamen, China, 6-9 Dec 2019

Nature-Inspired Computation in Engineering (NICE2019)
[A Symposium of IEEE SSCI2019]

2019 IEEE Symposium Series on Computational Intelligence

Home | Committee | TimeLine | Topics | Transportation | Sightseeing | History of SSCI | Contact

Nature-Inspired Computation in Engineering (NICE)

Nature-inspired computation and its optimization algorithms such as particle swarm optimization and firefly algorithm have become effective and popular in recent years, and they have

COMML2020

COMML2020, Xi'an, China, 10-12 April 2020

International Conference on Optimization, Metaheuristics and Machine Learning (COMML2020)

COMML 2020 : International Conference on Optimization, Metaheuristics and Machine Learning

Link: https://easychair.org/conferences/?conf=comml2020

When	Apr 10, 2020 - Apr 12, 2020
Where	Xi'an
Submission Deadline	May 31, 2019
Notification Due	Aug 31, 2019
Final Version Due	Oct 15, 2019

Categories optimization machine learning data mining computer science

Call For Papers

International Conference on Optimization, Metaheuristics and Machine Learning (COMML2020)

10-12 April 2020, Xi'an, China

The International Conference on Optimization, Metaheuristics and Machine Learning (COMML2020) will be held in Xi'an, China, 10 to 12 April 2020. COMML

Thank you :) Any questions?

- Xin-She Yang, Nature-Inspired Optimization Algorithms, Elsevier, (2014).
- Xin-She Yang, Cuckoo Search and Firefly Algorithm, Springer, (2014).
- Xin-She Yang, Optimization Techniques and Applications with Examples, Wiley, (2018).