UNIVERSIDAD COMPLUTENSE MADRID

Swarm Intelligence

Universidad Complutense
Ismael Rodríguez, Fernando Rubio

Introduction

- **Swarm Intelligence**. Nature inspired algorithms to solve:
  - **Combinatorial** domain optimization problems (ACO, MTPSO, RFD…)
  - **Continuous** domain optimization problems (PSO, ABC, DE…)
- Many (partially) **independent entities** cooperating to find a solution.
- Individuals are relatively **homogeneous** and interact with **simple rules**
- Balance:
  - **Exploration** of the overall search space
  - **Exploitation** of the most promising areas

Introduction

- There are many (many, many, many) swarm intelligence metaheuristics.
- EvolutionaryComputationBestiary https://github.com/fcampelo/EC-Bestiary

- Let's try to **focus** on the **mathematical basis** of classical metaheuristics
- Most of the *original* metaheuristics only provide a *new analogy*, but are essentially the same as other more classical metaheuristics

Swarm Intelligence: Overall view

- Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration versus Algorithmic Behavior, Critical Analysis and Recommendations Clasification by behavior:
  - **Differential vector movement** (207/323 => **64.09%**).
    - All population (13/323 => 4.02%). FA, CDA, …
    - Groups based (25/323 => 7.75%). CSO, BA, …
    - Representative based (169/323 => **52.32%**). **PSO**, ABC, …
  - **Solution creation** (116/323 => 35.91%):
    - **Combination** (108/323 => **33.43%**). **GA**, Cuckoo Search,…
    - **Stigmergy** (8/323 => **2.48%**). **ACO**, RFD,…

Swarm Intelligence: Particle Swarm Optimization

- Inspired on social behavior of a colony of bird flocks
- Minimize fitness function $f : R^n \rightarrow R$
- Each particle has an **initial random** position and an initial random **v** vector
- In each iteration: $\mathbf{v}_{i,d} \leftarrow \omega\, \mathbf{v}_{i,d} + \phi_p\, r_p\, (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g\, r_g\, (\mathbf{g}_d - \mathbf{x}_{i,d})$
  - **Inertia**
  - Vector towards **best local** position $\mathbf{p}_{i,d}$ found by itself
  - Vector towards **best global** position $\mathbf{g}_d$ found by the swarm

Swarm Intelligence: Particle Swarm Optimization

- In each iteration:
  - $\mathbf{v}_{i,d} \leftarrow \omega\, \mathbf{v}_{i,d} + \phi_p\, r_p\, (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g\, r_g\, (\mathbf{g}_d - \mathbf{x}_{i,d})$
  - $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
  - **if** $f(\mathbf{x}_i) < f(\mathbf{p}_i)$ **then** Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  - **if** $f(\mathbf{p}_i) < f(\mathbf{g})$ **then** Update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
  - **r_p and r_g** are **random** numbers between 0 and 1
  - **ω ϕ_p ϕ_g** are **parameters** of the metaheuristic. (inertia, cognitive component, and social component)

Swarm Intelligence: Particle Swarm Optimization

- **Initialization:**
  - Randomly distributed or around candidate solution
  - Velocity vectors randomly distributed or zero
- Implementation **details:**
  - Keep positions (and velocity) **inside search space**
    - Closest position inside search space
    - Velocity zero or inverse (whip back)

Swarm Intelligence: Particle Swarm Optimization

- Parameter tunning:
  - Classical values: social and cognitive around 2.
  - Pedersen values:

https://www.researchgate.net/profile/Mohamed-Mourad-Lafifi/post/Which-is-the-best-swarm-size-in-PSO/attachment/5b5b6f85b53d2f89289c14e1/AS%3A653084896288769%401532718981208/download/Good+Parameters+for+Particle+Swarm+Optimization.pdf

https://eprints.soton.ac.uk/71755/

Swarm Intelligence: Particle Swarm Optimization

- Variations
  - Neighborhood (ring, adaptive topologies, etc.)
  - Mutations, hybrid methods, simplifications, discrete version, etc.
  - Multiobjective versions

Swarm Intelligence: Particle Swarm Optimization

- **Your next task:**
  - Develop **your own PSO** program
  - Check it with classical benchmark functions (Sphere, Rastringin, Rosenbrock, etc.) Evolutionary programming made faster
  - Compare results with different parameters. Apply statistical tests:
    - https://www.isa.us.es/3.0/tool/statservice/
    - http://tec.citius.usc.es/stac/ranking.html
    - https://www.statskingdom.com/kruskal-wallis-calculator.html