Particle Swarm Optimization

Ginny Hogan
November 6, 2012

Outline

- Introduction
- Algorithm
- Benefits
- Downsides
- Convergence
- Variants
- Applications

Introduction

Background

- Kennedy, Eberhart, Shi 1995.
- Observations from nature.
- Swarm Intelligence
- Moves particles in search-space, searching for a "roost".

Set-Up

- Cost (or fitness) function $f : R^n \to R$.

Set-Up

- Cost (or fitness) function $f : R^n \rightarrow R$.
- Goal: minimize (or maximize) f over all possible positions in the search space, defined by upper and lower boundaries.

Set-Up

- Cost (or fitness) function $f : R^n \to R$.
- Goal: minimize (or maximize) f over all possible positions in the search space, defined by upper and lower boundaries.
- Parameters ω, ϕ_p and ϕ_g, chosen by practitioner.

Set-Up

- Cost (or fitness) function $f : R^n \to R$.
- Goal: minimize (or maximize) f over all possible positions in the search space, defined by upper and lower boundaries.
- Parameters ω, ϕ_p and ϕ_g, chosen by practitioner.
- Termination criterion.

Algorithm, Initialization

For each particle in the swarm:

- Initialize current position x_i with uniform random vector.

Algorithm, Initialization

For each particle in the swarm:

- Initialize current position x_i with uniform random vector.
- Set best position p_i to x_i.

Algorithm, Initialization

For each particle in the swarm:

- Initialize current position x_i with uniform random vector.
- Set best position p_i to x_i.
- If $f(x_i) < f(g)$, set g to x_i.

Algorithm, Initialization

For each particle in the swarm:

- Initialize current position x_i with uniform random vector.
- Set best position p_i to x_i.
- If $f(x_i) < f(g)$, set g to x_i.
- Initialize velocity v_i with uniform random vector.

Algorithm, Iteration

Until termination criterion, for each particle in the swarm, for each dimension, update position and velocity in the following way:

(Note that I am leaving out indices for dimension, just to make more readable.)

- To update velocity:

- To update velocity:
- Uniformly pick random numbers $r_p, r_g \in (0, 1)$.

- To update velocity:
- Uniformly pick random numbers $r_p, r_g \in (0, 1)$.
- Update velocity based on parameters and randomly selected numbers:

- To update velocity:
- Uniformly pick random numbers $r_p, r_g \in (0, 1)$.
- Update velocity based on parameters and randomly selected numbers:
- $v_i = \omega v_i + \phi_p r_p (p_i - x_i) + \phi_g r_g (g - x_i)$.

- Update current position:

- Update current position:
- $x_i = x_i + v_i$.

- Update current position:
- $x_i = x_i + v_i$.
- Check new $f(x_i)$ against particle and swarm, update if improved.

- Update current position:
- $x_i = x_i + v_i$.
- Check new $f(x_i)$ against particle and swarm, update if improved.
- Keep going until termination.

Acceleration Coefficients

- Recall:
- $v_i = \omega v_i + \phi_p r_p (p_i - x_i) + \phi_g r_g (g - x_i)$.

Acceleration Coefficients

- Recall:
- $v_i = \omega v_i + \phi_p r_p (p_i - x_i) + \phi_g r_g (g - x_i)$.
- Cognitive component: models tendency of particles to return to previous best positions.

Acceleration Coefficients

- Recall:
- $v_i = \omega v_i + \phi_p r_p (p_i - x_i) + \phi_g r_g (g - x_i)$.
- Cognitive component: models tendency of particles to return to previous best positions.
- Social component: quantifies performance relative to neighbors.

Inertia Weight ω

- Craziness
- Memory of the previous direction, prevents drastic change in directions.
- Bigger ω means more searching ability for whole swarm (exploration, don't get trapped in local minima).
- Smaller ω means more searching ability for partial swarm (exploitation, gets to know local search area very well).
- Experimental results: fastest convergence when $\omega \in (0.8, 1.2)$.

Topology

- Particles receive information from their neighbors.
- Network of neighborhoods forms a graph.
- Imitates different societies.
- Characterize neighborhoods by connectivity, clustering.

Types of Topologies

- Fully Connected Topology (gbest)
- Square Topology (Von Neumann)
- Ring Topology

Benefits

- Makes few assumptions about the problem.
- Doesn't require differentiability (doesn't use gradient).
- Large spaces of candidate solutions.
- Simple to implement.

Downsides

- Does not guarantee optimality.
- If maximum velocity too small, will only converge to local min.
- Weak theoretical foundation.
- Biased; solution more easily found if it is on axes.

Convergence

- Based on experimental studies, relative to other evolutionary algorithms, PSO has fast convergence ability but slow fine-tuning ability.
- Linearly decreasing inertia weight leads to better performance, but lacks global search ability.

Multiobjective/Constrained Optimization

- Originally, PSO for single objective continuous problem.
- Without constraints, sometimes particles want to go outside search space.
- Particles initialized with only feasible solutions (speeds up search process).
- Only select feasible solutions as best values.
- Initialization takes longer.

Pareto Optimality

- Search for multiple solutions.

Pareto Optimality

- Search for multiple solutions.
- Pick non-dominated solution.

Pareto Optimality

- Search for multiple solutions.
- Pick non-dominated solution.
- Requires decision-maker at the end.

Pareto Optimality

- Search for multiple solutions.
- Pick non-dominated solution.
- Requires decision-maker at the end.
- Aggregation: sum objective functions together using weighted aggregation.

Discrete Domains

- Basic idea: map discrete search space to continuous space, use a PSO, map result back to discrete space.
- Binary Particle Swarm Optimization – position is discrete, velocity is continuous.
- In velocity vector for agent i, v_i, v_{i_j} is probability that $x_{i_j} = 1$.

Applications

- Human tremor analysis.

Applications

- Human tremor analysis.
- Biomedical Engineering.

Applications

- Human tremor analysis.
- Biomedical Engineering.
- Electric power and voltage management.

Applications

- Human tremor analysis.
- Biomedical Engineering.
- Electric power and voltage management.
- Machine scheduling.

Applications

- Human tremor analysis.
- Biomedical Engineering.
- Electric power and voltage management.
- Machine scheduling.
- Point Pattern Matching.

Open Shop Scheduling Problem

- n jobs, m machines, each job has to be processed by each machine at least once (m operations per job).
- Order irrevelant, processing time can be zero.
- Multi-objective: want to minimize completion time (makespan), minimize idle machine time.
- NP-hard
- To use PSO, decode particle position into an active schedule.

Permutation-Based PSO for Open Shop Scheduling Problem

- Randomly generate group of particles represented by a permutation sequence (ordered list of operations)
- For n-job, m-machine problem, position of a particle is in m x n matrix.
- Let o_{ij} be operation of job j that must be processed on machine i (these are the particle positions).
- Let the objective function be $f_{(i,j)}$, the earliest time at which o_{ij} can be finished.
- Minimize f, add the corresponding operations to the schedule.

Bibliography

Bibliography

- Azlina, Nor, Mohamad Alias, Ammar Mohemmed, Kamarulzan Aziz. "Particle Swarm Optimization for Constrained and Multiobjective Problems: A Brief Review." 2011 Conference on Management and Artificial Intelligence. PDER vol. 6 (2011).
- Bai, Qinghai. "Analysis of Particle Swarm Optimization Algorithm." Computer and Information Science. Vol. 3, No. 1, February 2010.

- Clerc, Maurice and James Kennedy. "The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space." IEE Transcations on Evolutionary Computation, Vol. 6, No 1, February 2002.
- McCaffrey, James. "Particle Swarm Optimization." MSDN Magazine. August 2011.
- Montes de Oca, Marco. "Particle Swarm Optimization." University of Delaware, 2011.
- P. N. Suganthan. "Particle Swarm Optimiser with Neighbourhood Operator." Proceedings of the IEEE Congress on Evolutionary Computation (CEC), pages 1958-1962, 1999.
- http://tracer.uc3m.es/tws/pso/neighborhood.html