A very brief introduction to particle swarm optimization

Radoslav Harman

Department of Applied Mathematics and Statistics,
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

Note: I am no PSO expert, and this is just a simple handout to accompany a classroom lecture. If you find some errors in the following text, let me know, please.

The purpose of PSO

The usual aim of the particle swarm optimization (PSO) algorithm is to solve an unconstrained **minimization problem**: find x* such that f(x*)<=f(x) for all d-dimensional real vectors x. The objective function f: R^d -> R is called the **fitness function**.

History of PSO

PSO has been proposed by Eberhart and Kennedy in 1995, subsequently developed in thousands of scientific papers, and applied to many diverse problems, for instance neural networks training, data mining, signal processing, and optimal design of experiments.

Basic description of PSO

PSO is a **swarm intelligence** meta-heuristic inspired by the group behavior of animals, for example bird flocks or fish schools. Similarly to genetic algorithms (GAs), it is a population-based method, that is, it represents the state of the algorithm by a population, which is iteratively modified until a termination criterion is satisfied. In PSO algorithms, the population $P=\{p_1,\ldots,p_n\}$ of the feasible solutions is often called a **swarm**. The feasible solutions $p_1,\ldots,p_n$ are called **particles**. The PSO method views the set R^d of feasible solutions as a "space" where the particles "move". For solving practical problems, the number of particles is usually chosen between 10 and 50.

Comparison of PSOs to GAs

Unlike GAs, PSOs do not change the population from generation to generation, but keep the same population, iteratively updating the positions of the members of the population (i.e., particles). PSOs have no operators of "mutation", "recombination", and no notion of the "survival of the fittest". On the other hand, similarly to GAs, an important element of PSOs is that the members of the population "interact", or "influence" each other.

Swarm topology

Each particle i has its **neighborhood** N_i (a subset of P). The structure of the neighborhoods is called the **swarm topology**, which can be represented by a graph. Usual topologies are: **fully connected** topology and **circle** topology.

Characteristics of particle i at iteration t:

- $x_i^{(t)}$ … the **position** (a d-dimensional vector)
- $p_i^{(t)}$ … the "historically" **best position**
- $l_i^{(t)}$ … the historicaly **best position of the neighboring particles**; for the fully connected topology it is the historically best known position of the entire swarm
- $v_i^{(t)}$ … the **speed**; it is the step size between $x_i^{(t)}$ and $x_i^{(t+1)}$

At the beginning of the algorithm, the particle positions are randomly initialized, and the velocities are set to 0, or to small random values.

Parameters of the algorithm:

- $w^{(t)}$ … **inertia weight**; a damping factor, usually decreasing from around 0.9 to around 0.4 during the computation
- ϕ_1, ϕ_2 … **acceleration coefficients**; usually between 0 and 4.

Update of the speed and the positions of the particles

Many versions of the particle speed update exist, for example:

$$v_i^{(t+1)}=w^{(t)}v_i^{(t)}+ \phi_1 u_1(p_i^{(t)}-x_i^{(t)})+ \phi_2 u_2(l_i^{(t)}-x_i^{(t)}).$$

The symbols u_1 and u_2 represent random variables with the U(0,1) distribution. The first part of the velocity formula is called "**inertia**", the second one "**the cognitive** (personal) **component**", the third one is "**the social** (neighborhood) **component**". Position of particle i changes according to

$$x_i^{(t+1)}= x_i^{(t)}+v_i^{(t+1)}.$$

Stopping rule

The algorithm is terminated after a given number of iterations, or once the fitness values of the particles (or the particles themselves) are close enough in some sense.

PSO variants

There is a plethora of different versions of PSOs, which usually modify the formula for the change of velocity (e.g., instead of u_1 and u_2 they use diagonal matrices $\mathbf{U}_1$ and $\mathbf{U}_2$, in other variants they use no inertia, but enforce an upper **limit on the particle speed**, there is the so-called "**fully informed**" PSO, and there is also a popular modification using a "**constriction coefficient**"). There exist versions of the PSO for constrained optimization, for discrete optimization, and for multi-objective optimization.

Advantages and disadvantages

- The fitness function can be non-differentiable (only values of the fitness function are used). The method can be applied to optimization problems of large dimensions, often producing quality solutions more rapidly than alternative methods.
- There is no general convergence theory applicable to practical, multidimensional problems. For satisfactory results, tuning of input parameters and experimenting with various versions of the PSO method is sometimes necessary. Stochastic variability of the PSO results is very high for some problems and some values of the parameters. Also, some versions of the PSO method depend on the choice of the coordinate system.