The Effects of Neighborhood Topology and Neighborhood Influence Structure in Particle Swarm Optimization

Richardo Hopkins, Amy Sham, Luis Rosias

May 6, 2013

1 Introduction and Testing Methodology

The goal of this report is to analyze the impact of various parameter combinations on the performance of the particle swarm optimization (PSO) algorithm on a set of five standard benchmark functions. We determined which combinations produced the best performance and which parameters work best for certain neighborhood topology configurations.

This report summarizes the results produced from running PSO on all of the possible combinations of functions, swarm sizes, and parameters (of the ones we implemented). All results for each combination were averaged across 50 trials. The swarm operated in 30 dimensions (with the exception of Griewank, which operated in 10 dimensions) with 2000 iterations per trial. We recorded averages of the following: function value at each iteration, final function value, and the final function value's standard deviation.

We tested all 320 possible combinations of the following parameters:

- Function: Sphere, Rosenbrock, Griewank, Ackley, Rastrigin
- Swarm size: 10, 30, 50, 100
- Neighborhood topology: gbest ("global"), lbest ("right"), Von Neumann, random
- Particle inclusion: self included, self not included
- Neighborhood influence structure: personal-global (PG), fully-informed particle swarm (FIPS)

2 Results

The tests yielded strong patterns throughout the different solution spaces.

Changes in swarm size result in much more significant changes in fitness for lower swarm size values than for higher ones. The change in performance between swarms of size 9 and size 36 is much larger than that for swarms between 49 and 100. Plotting fitness as a function of swarm size would result in a function that increases logarithmically; that is, a function increasing rapidly at lower values and leveling off at higher values.

Neighborhood influence structure followed swarm size in impact on performance; specifically, whether or the swarm used PG (table columns 2 and 4) or FIPS (table columns 2 and 4). Swarms using PG consistently and significantly outperformed swarms using FIPS. For PG, self-inclusion did not affect performance; it generally did well regardless.

The combination of FIPS and no self-inclusion consistently yielded the worst results across all combinations. Combinations using FIPS did not use self-inclusion performed very poorly, with the worse combinations including the global and random neighborhood topologies. This is likely because all of the particles in its neighborhood have equal weight and no personal best that determines movement, so each particle just moves around randomly towards other particles. With the global neighborhood topology, the particles do not stay around a particular area; instead, they move towards the center of the swarm.

These patterns held pretty well in optimizing most of the functions, with the exception being Ackley function. With the Ackley function, most of the configurations ended up at a fitness of around 20.

3 Clarification on the Data

How to read the configurations in the graphs and tables:

z - [function type] - [swarm size] - [neighborhood topology] - [include self? yes/no] - [pg/fips].log

Because iteration count and dimension count did not change (except when noted in the introduction above), we did not incorporate those parameters in the configuration names.