Dilatational Particle Swarm Optimization

Mohammad Hasanzadeh Mofrad
hasanzadeh@cs.pitt.edu

November 4, 2016

Abstract

Particle Swarm Optimization (PSO) is an Evolutionary Computation (EC) technique that utilizes a swarm of particles to solve an optimization problem. Also, Slow Intelligence System (SIS) is a learning framework which slowly learns the solution to a problem performing a series of operations. In this project, we propose the Dilatational Particle Swarm Optimization (SPSO) which tries to model the swarm's learning episodes of the PSO algorithm within the slow/fast context of SIS. Also, We will show how the mapping from PSO to SIS may improve the overall performance of PSO.

1 Introduction

The SIS [1] is a slow learner with multiple decision cycles. In each decision cycle a set of operations are applied to the existing solutions of the target problem. In a SIS Abstract Machine, these operations could be any combination of *Enumeration*, *Propagation*, *Adaptation*, *Elimination*, and *Concentration* operators. In each decision cycle of SIS, a predicate that is constructed from these operators is shielded by a *guard* operator which controls the flow of operation from computationally inexpensive decision cycles to expensive ones.

The PSO [2] is an evolutionary algorithm utilizes a swarm of particles that iteratively calculates the optimal solution to a problem using particle's personal best position and swarm's global best position. In PSO, the position of particles are updated using the velocity formula which contains the current position, personal best position of a particle and global best position of the swarm. Calculating the distance of each particle from its personal best position and global best position of the swarm, in each iteration of PSO algorithm, the particles move toward the latest optimal position of the swarm. This individual and social moves of each particle will eventually lead to finding the optimal result of the candidate problem.

Both SIS and PSO are learning frameworks that iteratively approximate the best solution to a problem. In this project, we are going to combine these two approaches and exploit their advantages. Such a hybrid framework will contain the followings:

1. A PSO that has the ability to control its convergence speed using the slow and fast decision cycles of the SIS.
2. A set of new updating equations for particle's position and velocity which are written based on the SIS operators.
3. The proposed framework will be tested using the available evolutionary computation benchmarks which are written in Python e.g. benchmarks for the Real Optimization session on IEEE Congress on Evolutionary Computation CEC'2005 [3] or other publicly available benchmarks.
4. If time allows, in case of handling continues attributes for health care data, first we may use *linear regression* or *gradient descent* to construct the objective function for the PSO and then apply our evolutionary framework to the input data in order to find the optimal set of features for a dataset.

2 Framework

The Dilatational PSO (DPSO) is a combination of PSO and SIS. We map the current framework of SIS into DPSO as follows:

1. **Enumeration** consists of calculating the fitness of all available particles in the problem space. The fitness information will be passed to the next phases for future use.
2. **Propagation** in DPSO defines as the *personal best position* for each particle (*pbest*) and *global best position* for the entire swarm (*gbest*). This information is calculated in each decision cycle and SIS can utilize this information in order to propagate the current experience to other subcomponents.
3. **Adaptation** utilizes the current solutions to produce an elitist next generation of particles. In DPSO, the adaptation behaviour can be seen when the particles move toward the *pbest* and *gbest*.
4. **Elimination** rules out a set of infeasible solutions and keep the feasible solutions for the next decision cycle. The elimination operator is implemented in the context of DPSO by extracting the the k best generated solutions for the current decisions cycle where k is the enumeration factor and lets SIS to extract an elite subset of DPSO's solution.
5. **Concentration** tries to concentrate on the elite population of particles. In DPSO, the concentration step consists of updating the velocity and position of elite particles along with other non-elite particles. But the difference between these two groups of particles will be the fact that the non-elite particles will have a random mutation to see whether their fitness can be improved or not.

Moreover, we borrow the notion of *guard* from SIS and add the slow and fast decision cycles to the DPSO as follows:

1. Fast decision cycle with large step size:

$$w = w_{max} - (\frac{3}{2}((w_{max} - w_{min})/i_{max}) * i)$$

2. Slow decision cycle with small step size:

$$w = w_{max} - (((w_{max} - w_{min})/i_{max}) * i)$$

where w, $\mathrm{w_{max}}$, and $\mathrm{w_{min}}$ are weight, maximum allowed weight and minimum allowed weight and i and $\mathrm{i_{max}}$ are current decision cycle and maximum number of decision cycles.

In addition to the above contributions, we will change the standard PSO formulation to convey the groundtruth of SIS and follow the SIS's modular architecture.

3 Implementation

Currently, the project is under active development and we are writing the project in Python programming language. Also, The PSO implementation is quite done and we are trying to integre PSO and SIS. Moreover, there is a repository for the project in GitHub. To see the code and implementation details of this project you can browse this url: `https://github.com/hmofrad/pso` [4].

References

[1] Shi-Kuo Chang. A general framework for slow intelligence systems. *International Journal of Software Engineering and Knowledge Engineering*, 20(1):1–15, 2010.

[2] James Kennedy. Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer, 2011.

[3] Ponnuthurai N Suganthan, Nikolaus Hansen, Jing J Liang, Kalyanmoy Deb, Ying-Ping Chen, Anne Auger, and Santosh Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. *KanGAL report*, 2005005:2005, 2005.

[4] Particle swarm optimization (pso) implementation in python, Date accessed: 2016-11-03. https://github.com/hmofrad/pso.