COS710: Assignment 1

Myron Guanhao Ouyang
Department of Computer Science
University of Pretoria
South Africa
u16008368@tuks.co.za

***Abstract*—This report empirically compares Particle Swarm Optimization (PSO) and Standard Particle Swarm Optimization 2011 (SPSO2011). The analysis of this report will show that SPSO2011, although more complex than PSO, has no substantial benefit in terms of overall results**

I. Introduction

A successful search algorithm, known as Particle swarm optimization (PSO), developed by Kennedy and Eberhart in 1995 has inspired numerous variations of the original algorithm. This diversity of options (each with their own benefits and drawbacks) poses the problem of choice to the user trying to pick the most suitable PSO. This report will provided an empirical comparison between the basic PSO and the Standard PSO 2011 proposed by Clerc. Each algorithm will be run through 9 boundary constrained objective functions and the results for each function will be evaluated using the Mann-Whitney U statistical test.

II. Background

A. Basic Particle Swarm Optimization

The basic PSO implementation is rather simplistic in design but seems to be the core structure that most PSO variations are build on top of. The algorithm consists of randomly distributed particles (within a certain range) and after each iteration of the algorithm each particle updates its current position $x_i(t)$ by adding a velocity vector $v_i(t)$ to it.

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{1}$$

The velocity is computed as follows:

$$\begin{aligned} v_i(t+1) = (W * v_i(t)) + (c_1 * r_1 * (p_i(t) - x_i(t))) \\ +(c_2 * r_2 * (g_i(t) - x_i(t))) \end{aligned} \tag{2}$$

where the three coefficients Inertia weight, Cognitive Coefficient and Social Coefficient (W, c_1, c_2 respectively) are user defined based on the problem space. $v_i(t)$ refers to the velocity of the particle i at the time t. Velocity has components equal to the Dimension D of the search space. The global and local best position of the particle and particles in the swarm at the time t are denoted by $p_i(t)$ and $g_i(t)$ respectively. The random values r_1 and r_2 are chosen from a uniform distribution between the range of $[0, 1]$.

The global and local best positions of particles are determined by evaluating the fitness of the particles position. The fitness value is calculated by running the particles positions $x_i(t)$ through the objective function $f(x)$ (also referred to as: the benchmark function). In this case the lowest value is stored as the ”best” value.

Since the velocity is dependant on the global best position, it is clear that a Ring topology is utilized as the social network structure of the PSO.

B. Standard Particle Swarm Optimization 2011

One variant of the PSO algorithm is the Standard Particle Swarm Optimization 2011 (SPSO2011). The core structure of the algorithm is similar to the Basic PSO, with a couple of major changes to how the particles are initialized and how the velocity is computed.

The first key difference is the topology utilized; an adaptive random topology is used Clerc (2012). This implies that instead of a global best position, each particle in the swarm has its own local neighbourhood of particles which it will compare its own best position to. The topology is 'adaptive', due to its ability to change each particles neighbours if after an iteration of the algorithm, there is no change to the overall global best position.

The various attributes of a particle are initialized as follows Clerc (2012):

$$x_i(0) = U(min_d, max_d) \tag{3}$$

$$v_i(0) = U(min_d - x_i(0), max_d - x_i(0)) \tag{4}$$

$$p_i(0) = x_i(0) \tag{5}$$

$$n_i(0) = argmin_{j \epsilon_{N_i(0)}}(f(p_j(0))) \tag{6}$$

The velocity computation has been altered so as to not be fully dependant on just the positions $x_i(t)$ of the particle. This is done to ensure a better coverage done by the particles over the search space Clerc (2012). The velocity is computed as follows Cleghorn (n.d.*b*):

$$v_i(t+1) = W * v_i(t) + H_i(ce, ra) - x_i(t) \tag{7}$$

$$ce = (x_i(t) + \alpha_i(t) + \beta_i(t))/3 \tag{8}$$

$$ra = ||ce - x_i|| \tag{9}$$

$$\alpha_i(t) = x_i(t) + c_1 * r_1 * (p_i(t) - x_i(t)) \tag{10}$$

$$\beta_i(t) = x_i(t) + c_2 * r_2 * (n_i(t) - x_i(t)) \tag{11}$$

The function $H(ce, ra)$ will return a vector of random positions from the hyper-sphere centered around ce. The function $H(ce, ra)$ is calculated as follows:

1) Construct a vector re of length D with random scalar values between the range of $(0, 1)$.
2) Generate a scalar value s from a uniform distribution of range $(0, ra)$.
3) Return the vector value $s(re) + ce$

The following pseudo code was was taken from the provided slides Cleghorn (n.d.*a*):

```
repeat
    for all particles i = 1, ..., N do
        if f(x_i) < f(y_i) then
            p_i = x_i
        end
        for all particles i with particle i in their
          NBD do
            if f(p_i) < f(n_i) then
                n_i = p_i
                if f(n_i) < f(g) then
                    g = n_i
                end
            end
        end
    end
    for all particles i = 1, ..., N do
        update velocity of particle i using the
          chosen update equation
        update position of particle i using the
          chosen update equation
    end
until stopping condition is met;
```

Algorithm 1: Pseudo code for SPSO2011

III. Implementation

A. Basic Particle Swarm Optimization

In the basic PSO algorithm, particle positions are initialized randomly within the range of the objective functions bounds. The velocity $v_i(0)$ of each particle is initialized to a random number $U[0, 1)$. Each particles best fitness value $p_i(t)$ and the global best fitness value $g_i(t)$ is set to infinity. For each iteration the particles are evaluated; each particle is checked to ensure their position is within bounds. If the particle is within bounds, the particle goes through the objective function to produce a fitness value, if not the particle is ignored. This fitness value is compared to both the $p_i(t)$ and the $g_i(t)$, if it is lower than either fitness values, those values are replaced and the particles position is stored.

After all the particles have been evaluated, they are then moved based on the updated velocity $v_i(t+1)$. This is repeated until the max iterations is completed.

B. Standard Particle Swarm Optimization 2011

In the SPSO2011 algorithm, during initialization, the particle's neighborhood is assigned and the best neighborhood fitness value $n_i(t)$ is defined. The neighborhood consists of the particle itself and 3 random particles from the swarm (the same particle may be chosen multiple times). A global best fitness value $g_i(t)$ is also defined and assigned to infinity. The $g_i(t)$ is used to determine whether or not to reassign each particles neighborhood after each iteration.

Alike to the PSO algorithm, during each iteration, each particle is checked whether or not it is within bounds. If not it is ignored, else the particle will be evaluated. The evaluation process is the same to that of PSO, with the addition of running a check for the $n_i(t)$ in the particles neighborhood. After all the particles are evaluated and if the $g_i(t)$ has not changed, all the particles are reassigned a new neighborhood.

After the evaluation phase, the particle will move based on the updated velocity $v_i(t+1)$.

C. Objective Functions

The coefficients W, c_1 and c_2 have all been tuned for each individual function to produce adequate results. The tuning process was alike to a brute forcing attempt over a relatively short span of time, in which an attempt was made to guess the optimal values for the coefficients based on the starting values $W = 0.5, c_1 = c_2 = 1.75$ and steadily decreasing or increasing the coefficients based on the output the objective function gave.

The shifting and rotating of the functions were computed by applying f_x^{SHR} to the various functions. The shifting was done by subtracting the vector position $x_i(t)$ from a randomly generated vector of length D with each element within the range of $[-80, 80]$. After the shift, $x_i(t)$ is multiplied with a randomly generated orthonormal rotation matrix of size $[D, D]$. Finally the result of f_x is added to an arbitrarily chosen constant C.

D. General

Each algorithm (PSO and SPSO2011) is initialized with:

$D = 20$,

$SwarmSize = 20$,

$MaxIteration = 1000$

Each algorithm will then run through each objective function $MaxIteration$ times independently, multiplied by 30. After every run the best fitness value for that run will be stored in a textfile, resulting in 30 results for each Objective function per algorithm.

IV. Research Results

The Mann-Whitney U statistical test was used to compare each algorithm's results for each objective function against one another, to determine whether or not there was any difference between the two results that are statistically significant. The significance level measured was $\alpha = 0.05$. The H_0 for the experiment states that 'There is no difference in results obtained from either algorithm'.

Refer to Table II on page 6 to view the summarized results of each algorithm's performance on each objective function. It is clear that SPSO performs rather poorly in comparison with PSO, especially with regards to the non shifted and rotated functions. However SPSO is able to get relatively similar results with PSO when it comes to functions that have been shifted and rotated. In the case of the f_4^{SHR} , f_5^{SHR}, f_4 and f_5 it can be noted that SPSO has a smaller σ meaning there is a higher probability of more consistent results.

Table I on page 6 indicates the results of the Mann-Whitney U statistical test on both algorithms. Based on the first 5 objective function's results, it can be noted that there is without a doubt a significant difference between results. Due to the $U-value = 0$. Overall, all the data is statistically significant due to the $p-value$ being lower than the α. This means that the H_0 is rejected.

These results were not initially expected, as SPSO2011 was meant to be an improvement on the standard PSO. SPSO2011 was meant to be able to provide not necessarily the better results but to provide more consistent results, which based on the σ comparison between the two algorithm doesn't seem to be the case.

The difference between the results of SPSO2011 and PSO is probably due to the fact that the particles in SPSO2011 have a lower chance of locating the global minimum due to the fact that their $v_i(t+1)$ calculation is reliant on the best position of a neighborhood that might not even be heading towards the global minimum. There might be better results if each particle changes their neighborhood if the $n_i(t)$ isn't improved after each iteration instead of the $g_i(t)$.

V. Conclusion

From the data provided, it is evident that SPSO2011 does not provide the results necessary to support its added complexity compared to the Basic PSO. Both algorithms have been empirically compared to one another, by means of using the Mann-Whitney U statistical test on the results of each algorithm. The results are obtained by testing each algorithm on a set of 9 objective functions.

Although SPSO2011 does provide some consistency compared to PSO, it is not enough to justify its use over PSO with regards to overall results.

References

Cleghorn, C. W. (n.d.*a*), 'Particle swarm optimization: Part 2'.
URL: *https://cs.up.ac.za/files/COS710/Download/4396/*

Cleghorn, C. W. (n.d.*b*), 'Particle swarm optimization: Part 3'.
URL: *https://cs.up.ac.za/files/COS710/Download/4432/*

Clerc, M. (2012), 'Standard particle swarm optimisation', pp. 1–14.

Engelbrecht, A. (2013), 'Particle swarm optimization: Global best or local best?', *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence* pp. 124–125, 133–135.

Liang, J., Y. Qu, B. & Suganthan, P. (2013), 'Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization'.

Appendix

Each objective function that is referenced in this paper is defined here, including each functions specifications such as boundaries and constants added during shifting SH and rotating R of the function. All the function definitions where taken from Liang et al. (2013) and Engelbrecht (2013).

f_1, Sphere function is defined as follows:

$$f_1(x) = \sum_{j=1}^{n_x} x_i^2 \tag{12}$$

with boundaries $[-5.12, 5.12]$

f_2, Ackley function is defined as follows:

$$f_2(x) = -20e(\sqrt[-0.2]{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - e(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e \tag{13}$$

with boundaries $[-30, 30]$

f_3, Katsuura function is defined as follows:

$$f_4(x) = \frac{10}{D^2}\prod_{i=1}^{D}(1 + i\sum_{j=1}^{32}\frac{\left|2^j x_i - round(2^j x_i)\right|}{2^j})^{\frac{10}{D^{12}}} - \frac{10}{D^2} \tag{14}$$

with boundaries $[-5, 5]$

f_4, Michalewicz function is defined as follows:

$$f_4(x) = -\sum_{j=1}^{n_x}\sin(x_j)(\sin(\frac{jx_j^2}{\pi}))^{2m} \tag{15}$$

with boundaries $[0, \pi]$

f_5, Shubert function is defined as follows:

$$f_5(x) = \prod_{j=1}^{n_x}(\sum_{i=1}^{5}(i\cos(i+1)x_j + i)) \tag{16}$$

with boundaries $[-10, 10]$

f_x^{SHR}, General formula to shift and rotate is defined as follows:

$$f_x^{SHR} = f_x(M(x - o)) + C \tag{17}$$

with boundaries $[-100, 100]$

Table I

RESULTS OF MANN-WHITNEY U TEST

	$p-values$	$U-value$	$Z-value$
f_1	2.87196e-11	0.0	-6.652991
f_2	2.87196e-11	0.0	-6.652991
f_3	2.87196e-11	0.0	-6.652991
f_4	2.87196e-11	0.0	-6.652991
f_5	2.87196e-11	0.0	-6.652991
f_2^{SHR}	1.36902e-8	66.00	-5.677219
f_3^{SHR}	1.86473e-10	19.00	-6.372087
f_4^{SHR}	0.000107275	188.00	-3.873519
f_5^{SHR}	8.48723e-10	35.00	-6.135537

Table II

COMPARISON BETWEEN PSO AND SPSO2011

	PSO			**SPSO**		
	σ	$\bar{x}$	$BestFitness$	σ	$\bar{x}$	$BestFitness$
f_1	4.91750e-22	1.72695e-22	4.61624114757436e-29	6.372548	14.747555	5.97588870417864
f_2	0.978143	0.939943	9.89912725124497e-08	1.809612	12.881326	12.4652772761545
f_3	0.0352098	0.0703650	0.0716387488186742	0.616035	2.419839	1.8350687825835
f_4	1.053225	-16.767770	-17.6208125763153	0.990326	7.660159	-7.50092502788343
f_5	21318146e+15	-1942526+e17	-227986114808639	33619986e+7	-14622033e+7	-141515641406440
f_2^{SHR}	0.0992571	-119.146046	-119.075795307226	0.119489	-118.908359	-119.103929814662
f_3^{SHR}	0.327419	131.308175	131.018510769177	0.535162	132.464890	131.74051924033
f_4^{SHR}	0.655649	-104.816894	-105.110874327928	0.567094	-104.193308	-105.097808638903
f_5^{SHR}	19087559e+9	-6403336873364475.	-64216211656987.6	151469602e+6	-5858953e+7	-11656297621122