Asynchronous Steady State Particle Swarm

Carlos M. Fernandes
LARSyS: Laboratory for Robotics and Systems in Engineering and Science, University of Lisbon, Lisbon, Portugal
cfernandes@laseeb.org

Juan Julián Merelo
Department of Computer Architecture
University of Granada, Granada, Spain
jmerelo@geneura.ugr.es

Agostinho C. Rosa
LARSyS: Laboratory for Robotics and Systems in Engineering and Science, University, Lisbon, Portugal
acrosa@laseeb.org

ABSTRACT
We propose an asynchronous and steady state update strategy for the Particle Swarm Optimization inspired by the Bak-Sneppen model of co-evolution between interacting species: only the worst particle and its neighbors are updated and evaluated in each time-step. The strategy improves the quality of results and convergence speed of PSO with Moore neighborhood.

Keywords: Particle Swarm Optimization, Asynchronous PSO, Self-Organized Criticality.

1. INTRODUCTION
The standard Particle Swarm Optimization (PSO) algorithm [4] is synchronous: the fitness of all particles is computed and only then the particles update their velocity. In 2001, Carlisle and Dozier [2] proposed the asynchronous PSO (A-PSO), a variant in which the velocity is updated immediately after computing the fitness. In this case, each particle is updated knowing the current best position found by half of its neighbors and the previous best found by the other half: the population of the A-PSO move with imperfect information about the global search.

Asynchronous PSOs have been compared to the synchronous configuration (S-PSO) with contradictory results. While Carlisle and Dozier suggested that A-PSO yields better results than S-PSO, a study by Rada-Vilela *et al.* [6] reported that S-PSO is better than A-PSO in terms of the quality of the solutions and convergence speed.

In this paper, we follow an alternative approach. The goal is to design an asynchronous strategy that, unlike A-PSO, significantly improves S-PSO in a wide range of problems. With that objective in mind, we propose the steady state PSO (SS-PSO). A system is said to be in steady state when some its parts do not change for a period of time. In the SS-PSO, only a fraction of the population is updated and evaluated in each iteration.

The strategy is inspired by the Bak-Sneppen model of co-evolution between interacting species [1]. In order to investigate the dynamics of species extinction and coupled selection, P. Bak and K. Sneppen arranged a set of random fitness values (representing species) in a ring structure. Then, they replaced the worst species and its neighbors by random values (extinction event), repeating the procedure during several iterations. After a long run, the system is driven to a critical state were most species have reached a fitness above a certain threshold and avalanches of extinction events produce non-equilibrium fluctuations in the configuration of the fitness values. The Bak-Sneppen model is an example of self-organized criticality (SOC). In the past, SOC and the Bak-Sneppen model inspired alternative strategies for PSO (see [3] and [5]).

Like the Bak-Sneppen model, the population of PSO is structured by a network. With this likeness in mind, we devised an asynchronous and steady state update strategy for PSO in which only the least fit particle and its neighbors are updated and evaluated in each time step. The neighborhood is defined by the social structure: i.e., if the particles are connected by a *lbest* topology with $k = 3$, then only the worst particle and its two nearest neighbors are updated and evaluated; if a 2-dimensional lattice with Moore neighborhood is used, then the least fit and its eight nearest neighbors are updated ($k = 9$). Please note that local synchronicity is used here: the fitness values of the worst and its neighbors are first computed and only then the particles update their velocity. The SS-PSO is summarized in Algorithm 1.

1. Initialize velocity and position of each particle.
2. for (each particle j):
   2.1. Compute fitness.
3. for (each particle j):
   3.1. Compute p_i and p_g.
4. For (each particle j):
   4.1 if j is the least fit particle, update velocity and position of j and neighbors.
4. Compute fitness of j and neighbors.
5. If (stop criteria not met) return to 3; else, end.

Algorithm 1: SS-PSO

2. RESULTS AND CONCLUSIONS
The algorithm was tested on eight benchmark unimodal and multimodal functions: Sphere (f_1), Quadric (f_2), Hyperellipsoid (f_3), Rastrigin (f_4), Griewank (f_5), Schaffer (f_6), Weirstrass (f_7) and Ackley (f_8). The dimension of the search space is set to $D = 30$ (except f_6, for which $D = 2$). In order to construct square lattices with von Neumann and Moore neighborhood, the population size μ is set to 49, a value that lies within the typical range of PSO population size [4]. Following [6], the acceleration coefficients were set to 1.49618 and the inertia weight is 0.729844. $Xmax$ is defined as usual by the domain's upper limit and $Vmax = Xmax$. A total of 50 runs for each experiment were performed. *Asymmetrical initialization* is used.

GECCO'16 Companion, July 20-24, 2016, Denver, CO, USA
ACM 978-1-4503-4323-7/16/07.
http://dx.doi.org/10.1145/2908961.2909035.

Table 1. PSO and SS-PSO with Moore neighborhood. Best fitness values: mean, median and standard deviation.

.	S-PSO$_{Moore}$			SS-PSO$_{Moore}$		
	mean	median	st.dev.	mean	median	st.dev.
f_1	3.36e-42	1.03e-41	1.01e-41	**0.00e00**	**0.00e00**	0.00e00
f_2	1.38e-29	5.88e-31	6.36e-29	**1.40e-46**	**0.00e00**	9.90e-46
f_3	1.09e-42	1.40e-43	2.86e-42	**0.00e00**	**0.00e00**	0.00e00
f_4	6.36e+01	6.17e+01	1.73e+01	**5.25e+01**	**5.12e+01**	1.45e+01
f_5	**5.86e-03**	**1.08e-19**	7.22e-03	1.28e-02	9.86e-03	2.06e-02
f_6	1.94e-04	**0.00e00**	1.37e-03	**0.00e00**	**0.00e00**	0.00e00
f_7	1.94e-01	2.27e-02	5.27e-01	**1.73e-02**	**2.86e-05**	8.17e-02
f_8	**1.01e-15**	**8.88e-16**	2.01e-16	1.07e-15	**8.88e-16**	2.20e-16

For assessing the quality of solutions and speed of convergence of the algorithms, two sets of experiments were conducted. In the first, the algorithms were run for a limited amount of iterations (3000 for f_1, f_3 and f_6, 20000 for the remaining) and the fitness of the best solution found was averaged over the 50 runs. In the second set of experiments the algorithms were all run for 20000 iterations or until reaching a stop criterion. The number of iterations required to meet the criteria was recorded and averaged over the 50 runs. A success measure was defined as the number of runs in which an algorithm attains the fitness value established as the stop criterion (success rate).

The objective of the experiments is to evaluate the impact of the proposed update strategy in the performance of standard PSOs. For that purpose, the steady state update strategy was implemented on PSOs with two different population structures: *lbest* (SS-PSO$_{lbest}$) and 2-dimensional square lattices with Moore (SS-PSO$_{Moore}$) neighborhood. While no improvement has been observed with *lbest*, the experiments demonstrated that the strategy improves the quality of results and convergence speed of PSO with Moore neighborhood. The numerical results of the algorithms with Moore structure are in Tables 1 (best fitness values) and 2 (evaluations required to meet the stop criterion and success rates).

SS-PSO finds better solutions in the unimodal functions ($f_1 - f_3$). In the set of multimodal problems, SS-PSO is better in functions f_4, f_6 and f_7. (Results of Mann-Whitney U tests are significant at $p \leq 0.05$ for functions f_1, f_2, f_3, f_4, f_6 and f_7, i.e., the null hypothesis that that the two samples come from the same population is rejected.) In terms of function evaluations, SS-PSO is faster in the entire set of unimodal problems. In the multimodal problems, SS-PSO is faster in f_5, f_6, f_6 and f_8. (Results of Mann-Whitney U tests are significant at $p \leq 0.05$ for functions f_1, f_2, f_3, f_5, f_7 and f_8.). The success rates (see Table 2) are similar, except for f_7, in which SS-PSO clearly outperforms the synchronous version.

Since the experiments also demonstrated that S-PSO$_{Moore}$ is better than S-PSO$_{lbest}$ in most of the functions, ranking first in both quality of solutions and speed of convergence, we believe that these results validate the proposed steady state and asynchronous update strategy for particle swarms. However, further research is required in order to understand why the performance is not improved when using the *lbest* network.

Table 2. Number of evaluations to reach the stop criteria (mean, median and standard deviation) and success rates.

	S-PSO$_{Moore}$				SS-PSO$_{Moore}$			
	mean	median	st.dev.	SR	mean	median	st.dev.	SR
f_1	20434.0	20433. 0	840.8	**50**	**17241.3**	**17320.5**	716.2	**50**
f_2	168599.0	168119.0	12721.1	**50**	**133140.6**	**135828.0**	16854.2	**50**
f_3	22987.9	22956.5	1075.4	**50**	**19519.6**	**19561.5**	788.0	**50**
f_4	**15635.0**	**13524.0**	7771.5	**49**	15902.8	14256.0	8047.7	**49**
f_5	18671.0	18595.5	986.8	**50**	**16419.2**	**16060.5**	1300.7	**50**
f_6	11443.0	7105.0	9439.1	49	**8049.0**	**6381.0**	4852.6	**50**
f_7	37272.7	36970.5	1590.1	24	**33192.0**	**33340.5**	1184.8	**46**
f_8	21029.8	20923.0	1164.7	**50**	**17723.6**	**17752.5**	957.0	**50**

ACKNOWLEDGEMENTS

The first author wishes to thank FCT, *Ministério da Ciência e Tecnologia*, his Research Fellowship SFRH/BPD/66876/2009). This work was supported by FCT PROJECT [PEst-OE/EEI/LA0009/2013], EPHEMECH (TIN2014-56494-C4-3-P, Spanish Ministry of Economy and Competitivity), PROY-PP2015-06 (Plan Propio 2015 UGR), and project CEI2015-MP-V17 of the Microprojects program 2015 from CEI BioTIC Granada.

References

[1] Bak, P. and Sneppen. K. 1993. Punctuated equilibrium and criticality in a simple model of evolution, Physical Review Letters 71 (24), 4083–4086

[2] Carlisle, A. and Dozier, G. 2001. An off-the-shelf PSO. Workshop on Particle Swarm Optimization.

[3] Fernandes, C.M., Merelo, J.J. and Rosa, A.C. 2012. Controlling the Parameters of the Particle Swarm Optimization with Self-Organized Criticality, Proc. of the 12th Parallel Problem Solving from Nature - PPSN XII, LNCS 7492, 153-163.

[4] Kennedy, J. and Eberhart, R. 1995. Particle Swarm Optimization, Proceedings of IEEE International Conference on Neural Networks, Vol.4, 1942–1948.

[5] Løvbjerg, M. and Krink, T. 2002. Extending particle swarm optimizers with self-organized criticality, Proceedings of the 2002 IEEE Congress on Evolutionary Computation, Vol. 2, IEEE Computer Society, 1588–1593.

[6] Rada-Vilela, J., Zhang, M. and Seah, W. 2013. A Performance Study on Synchronous and Asynchrounous Updates in Particle Swarm, Soft Computing 17(6), 1019–1030.