A Parallel Hybrid GA-PSO Approach with Dynamic Rule-Based Parameter Setting

Abtin Nourmohammadzadeh
Clausthal University of Technology
Clausthal-Zellerfeld, Germany
an14@tu-clausthal.de

Sven Hartmann
Clausthal University of Technology
Clausthal-Zellerfeld, Germany
sven.hartmann@tu-clausthal.de

Hui Ma
Victoria University of Wellington
Wellington, New Zealand
hui.ma@ecs.vuw.ac.nz

ABSTRACT

We present an new optimisation framework combining two meta-heuristics: Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO). In contrast to the usual hybridisation models in which the second algorithm is applied to work on the final results of the first one, our approach uses both algorithms in parallel on the same population in a competetive manner. The algorithms can work on and improve the solutions of each other, thus more diversity and better quality can be achieved in the population. Another improving factor is resetting the population size and the parameters in every iteration according to the diversity and quality of the solutions in the last population. Our approach is tested on five well-known benchmark problems. The merit of our approach is verified by comparing its performance with the pure GA and PSO, hybrids where PSO works after GA, and vice versa, as well as another hybrid approach of these algorithms from the literature.

KEYWORDS

Parallel hybridaration, Genetic Algorithm, Particle Swarm Optimisation, Dynamic Parameter Setting

1 INTRODUCTION

While meta-heuristic optimisation algorithms are promising techniques for solving complex problems, all of them are known for particular strengths and weaknesses. Hence, the idea of combining multiple meta-heuristics (called hybridisation) has attracted much attention; for a survey see [6]. We assume that the reader is familiar with Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO). Though both GA and PSO have been proved to be successful on a variety of problems, each has its own weaknesses [4, 5]. For example, in GA if an individual is not selected, its inherent information will be lost, while in PSO the memory avoids this problem. On the other hand, PSO suffers from the lack of a selection operator, and so it may waste its resources on poor individuals.

To address this challenge, hybrid approaches have been proposed in the literature [2, 7]. In [7] a hybrid GA-PSO approach named Breeding Swarms is introduced by combining the velocity and position update rules of PSO with the GA operators selection, crossover and mutation. In [2] a hybrid approach starting with PSO is presented in which a GA is embedded in each iteration to improve a specific number of particles. Further, there is an increasing interest in using hybrid GA-PSO techniques in industrial applications [1, 3]. A review of the existing research shows that the full potential of hybridisation techniques for GA and PSO has not yet been explored, and reveals the lack of a parallel model which gives both algorithms a fair chance to contribute their individual strengths.

GECCO '17 Companion, Berlin, Germany
 978-1-4503-4939-0/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3067695.3076051

2 OUR PARALLEL HYBRID GA-PSO

We aim to develop advanced hybridisation techniques for combining GA and PSO in order to better compensate for drawbacks of one with the strengths of the other. We propose a novel parallel hybrid GA-PSO approach (called P_GA-PSO). A special characteristic of our approach is that both GA and PSO are given the same opportunity to simultaneously work on the same population in each iteration and show their capabilities. We enforce a kind of competitiveness among both by giving the chance to the algorithm which has performed better in one iteration to have a bigger share in the next iteration. Moreover, in each iteration the parameters of GA and PSO are adjusted dynamically after testing the diversity and quality improvements of the resulting population.

Now we present the stepwise description of our approach:

1. Initialise a random population *Pop(1)* of size *Sbase*
2. Sort *Pop(1)* based on the fitness of the individuals
3. Consider the population individuals as the particles of PSO, and also as the first generation of GA
4. Update velocity and position of particles by the PSO rules, and apply the GA operators selection, crossover and mutation
5. Evaluate the new particle positions of PSO, and add the offspring individuals obtained in GA from crossover and mutation to the population, and then sort the pooled population based on fitness
6. Calculate the average fitness $MeanQ_{PSO}$ an $MeanQ_{GA}$ of PSO and GA, respectively
7. Determine the shares in the next iteration (S_{GA} and S_{PSO}) of each algorithm based on the mean quality, i.e., 0.6 for the better one and 0.4 for the other
8. Fill *Basepop* from the best of the PSO and GA results based on the shares S_{GA} and S_{PSO}, respectively
9. Calculate the diversity of *Basepop*
10. If the diversity is more than *Mindiv* consider it as *Newpop* and go to step 13; otherwise go to the next step
11. Consider an extension *Divpop* with size *SDivpop* and fill it randomly with some unique individuals from the rest of the PSO and GA results based on the shares
12. Merge *Divpop* into *Basepop* to obtain *Newpop*
13. If the termination criterion is met, then report the best individual in *Newpop* as the global best and final result of the algorithm, and stop; otherwise go to the next step

14. Increase the iteration counter by 1
15. Consider $Newpop$ as the population of the new iteration
16. Examine the improvement of the population of the new iteration over those of previous iterations, and reset the exploration and exploitation parameters of GA and PSO, accordingly
17. Return to Step 2

In Step 9 the diversity is defined as the variance $Varpop = \frac{\sum_{i=1}^{Sbase} \|x_i - \mu_{pop}\|^2}{Sbase-1}$ of the individuals in the solution space, where μ_{pop} is the mean point of the population.

In Step 16 a vector $Imp = [imp_1, imp_2, imp_3]$ is used that measures the improvements of the new iteration over the 1, 5 and 10 iterations ago. The GA parameters crossover rate CR and mutation rate MR, and the PSO parameters W, $C1$, $C2$ are reset by problem-specific rules whose general structure is as follows:

- if $imp_1 \geq L1$ then set $CR = cr_1, CM = cm_1, W = w_1, C1 = c1_1, C2 = c2_1$
- if $imp_1 \leq L1$ and $imp_2 \geq L2$ then set $CR = cr_2, CM = cm_2, W = w_2, C1 = c1_2, C2 = c2_2$
- if $imp_1 \leq L1$ and $imp_2 \leq L2$ and $imp_3 \geq L3$ then set $CR = cr_3, CM = cm_3, W = w_3, C1 = c1_3, C2 = c2_3$
- if $imp_1 \leq L1$ and $imp_2 \leq L2$ and $imp_4 \leq L3$ then set $CR = cr_4, CM = cm_4, W = w_4, C1 = c1_4, C2 = c2_4$

3 EXPERIMENTAL RESULTS

We have conducted experiments to compare the performance of our approach with five other approaches: the pure GA, the pure PSO, the hybrids where GA and PSO are applied consecutively ($GA \longrightarrow PSO$ or $PSO \longrightarrow GA$, resp.), and the hybrid GA-PSO approach in [7]. For our experiments we used MATLAB on a PC with an Intel(R) Core(TM) i7, 3.10GHz CPU, and 16GB RAM.

We have tested against the following five well-known benchmark functions $f_i(x)$ that have already been used in [7]:

1. Ellipsoidal $$f_1(x) = \sum_{i=1}^{n} ix_i^2$$
2. Rosenbrock $$f_2(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$
3. Rastrigin $$f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10cos(2\pi x_i) + 10)$$
4. Greiwank $$f_4(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=0}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1$$
5. Ackley $$f_5(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)}$$

We have solved each test problem in $n = 30$ dimensions. Each experiment has been repeated 30 times. Table 1 shows the mean function value, the standard deviation, and the average execution time till termination for each of the benchmark test functions.

We have set the sizes of the base and diversification population to $Sbase = 100$, and $Divpop = 30$. The termination criterion has been $Maxit = 1000$. For the first iteration, we have set parameters as follows: $MR = mr_3$, $CR = cr_2$, $W = w_2$, $C1 = c1_2$, $C2 = c2_4$.

Table 1: Performance of our approach and others on the five benchmark test functions $f_i(x)$

Method	Benchmark Test Function	Mean Function Value	Std.	Average Execution Time (s)
Pure GA	f_1	6.08	5.90	10.01
	f_2	25.49	13.06	12.73
	f_3	8.3E-02	6.82E-02	12.85
	f_4	6.32E-01	1.18	13.26
	f_5	8.21E-03	2.46E-02	13.38
Pure PSO	f_1	3.21	4.45	11.40
	f_2	37.26	15.21	12.56
	f_3	5.16	1.15	12.93
	f_4	9.84E-01	1.65	13.44
	f_5	8.00E-03	4.57E-04	14.17
$GA \longrightarrow PSO$	f_1	9.36E-01	8,60E-01	16.28
	f_2	2.82	3.25	16.42
	f_3	1.4E-02	5.06E-02	16.50
	f_4	5.09E-01	9.70E-01	17.85
	f_5	3.21E-04	2.19E-02	17.92
$PSO \longrightarrow GA$	f_1	8.78E-01	9.28E-01	16.32
	f_2	5.09	3.86	16.40
	f_3	1.19	8.04E-01	16.53
	f_4	2.28E-03	8.41E-01	17.79
	f_5	1.66E-04	6.93E-03	18.02
GA-PSO [7]	f_1	7.85E-47	1.38E-45	NA
	f_2	6.248	4.211	NA
	f_3	1.78E-16	6.41E-16	NA
	f_4	1.75E-02	1.97E-02	NA
	f_5	5.74E-15	3.50E-15	NA
Our Approach (P_GA-PSO)	f_1	**0**	**0**	**14.46**
	f_2	**6.05E-05**	**3.12E-04**	**14.72**
	f_3	**0**	**0**	**14.68**
	f_4	**1.01E-08**	**5.12E-06**	**14.81**
	f_5	**3.60E-24**	**3.41E-23**	**15.24**

4 CONCLUSION

We observe that our approach (P_GA-PSO) is able to find optimal or near-optimal solutions. In comparison with other methods, our approach outperforms the pure GA, the pure PSO, $GA \longrightarrow PSO$ and $PSO \longrightarrow GA$ in almost all cases as it achieves a lower mean, and is relatively fast. Furthermore, our approach is better than [7] in many cases; in some cases the performance of both approaches is similar; only in very few cases [7] is better than our approach.

For future research we propose to explore further improvements of the proven capability of our approach by further enhancing the system of parameter updating using problem-specific rules.

REFERENCES

[1] W. Chen, M. Nguyen, W. Chiu, T. Chen, and P. Tai. 2016. Optimization of the plastic injection molding process using the Taguchi method, RSM, and hybrid GA-PSO. *The International Journal of Advanced Manufacturing Technology* 83, 9 (2016), 1873–1886.
[2] H. Garg. 2016. A hybrid PSO-GA algorithm for constrained optimization problems. *Appl. Math. Comput.* 274 (2016), 292–305.
[3] M. Ghovvati, G. Khayati, H. Attar, and A. Vaziri. 2016. Kinetic parameters estimation of protease production using penalty function method with hybrid genetic algorithm and particle swarm optimization. *Biotechnology & Biotechnological Equipment* 30, 2 (2016), 404–410.
[4] R. L. Haupt and S. E. Haupt. 2004. *Practical Genetic Algorithms.* Wiley.
[5] J. Kennedy, R. C. Eberhart, and Y. Shi. 2001. *Swarm intelligence.* Morgan Kaufman.
[6] G. R. Raidl. 2006. *A Unified View on Hybrid Metaheuristics.* Springer, 1–12.
[7] M. Settles and T. Soule. 2005. Breeding Swarms: A GA/PSO Hybrid. In *ACM Annual Conference on Genetic and Evolutionary Computation (GECCO).* 161–168.