2009 International Symposium on Computing, Communication, and Control (ISCCC 2009)
Proc .of CSIT vol.1 (2011) © (2011) IACSIT Press, Singapore

Speciation with GP Based Hybrid PSO

Muhammad Rashid [1 +] and Abdul Rauf Baig [2]

[1,2] Department of Computer Science, National University of Computer and Emerging Sciences, A.K. Brohi Road, Sector H-11/4, Islamabad, Pakistan

Abstract. In this study we present an extension to the PSOGP algorithm for multimodal optimization problems. PSOGP avoids premature convergence by utilizing a method wherein the swarm is made more diverse by employing a mechanism which allows each particle to use a different equation to update its velocity. This equation is also continuously evolved through the use of genetic programming to ensure adaptability. Enhancements have been proposed which make PSOGP suitable for finding solutions to multimodal optimization problems. We propose a partial random initialization strategy and a generation gap strategy. We also suggest the use of speciation which enables PSOGP to locate multiple solutions. We compare the performance of SPSOGP with SPSO and NichePSO on 5 multimodal test functions.

Keywords: particle swarm optimization, genetic programming, multimodal function optimization, evolution, force generating function, species

1. Introduction

Kennedy and Eberhart [10] proposed the PSO algorithm which uses a swarm of particles to explore the search space. The individual particles move about the search space in an effort to find the optima taking guidance from other particles which have higher fitness as well as from their own experiences. Each particle keeps track of its current position, current velocity and personal best position. In every iteration the velocity of the particles are updated using a force generating function which uses information about that particle's best position, the global best position and random values to determine the change in velocity. The new velocity is then used to update the position of the particle thus enabling it to move through the search space.

In PSOGP [16], enhancements were suggested which allowed both evolution and diversity to be incorporated into PSO. PSOGP utilizes different force generating functions for each particle to ensure diversity in the swarm. So in addition to keeping track of its current position, current velocity, personal best position and global best position, the particle will also keep track of the force generating function used to update its velocity. The force generating function is also evolved throughout the execution of the PSOGP, thus allowing it to adapt to the specific problem. The GP that is used to evolve the force generating function differs from the classical GP in the sense that the population size is limited (i.e. the number of programs in a generation are equal to the number of particles). The function set includes the functions +, -, × and the protected division DIV. The terminal set includes the position of the particle x, the velocity of the particle v, the personal best position of the particle p, the global best of the swarm g and a constant c. In the beginning while initializing the position of the particle the force generating function of each particle is randomly initialized. Within each iteration of PSOGP algorithm, along with updating the position of the particle and the velocity, the force generating function is also evolved using the cross over and mutation operators of genetic programming. One or two parents are selected from the original swarm at random depending upon the operator that is being applied. The particles having better fitness have a greater probability of being

[+] Corresponding author. Tel.: +923335137267; fax: +92514100619.
E-mail address: rashid.nuces@gmail.com.

selected. The force generating functions of all particles are replaced with their children in every iteration. By allowing each particle to have a different force generating function and by evolving those equations through GP, PSOGP can be considered a PSO hybrid which is, not only diverse, but can also adapt to the optimization problem [16].

In this study we further extend PSOGP to enable it to efficiently find solutions to multimodal optimization problems. The proposed SPSOGP algorithm utilizes speciation to locate multiple solutions in a multimodal search space. In the next section we present some of the existing approaches which implement niching and speciation with PSO. In section 3 we present our extensions to the PSOGP algorithm which is followed by a description of the experimentation carried out and the results obtained from those experimentations. In the end we present our conclusion.

2. Niching and Speciation with PSO

Niching algorithms are those algorithms which are capable of locating multiple solutions to a problem. Niches can be defined as partitions of an environment which represent one solution to a problem. Speciation is the process of finding multiple niches or solutions. Species are the partitions of a population competing within an environment. They are the group of particles which converge on a single niche [7]. Kassabalidis et al. [8, 9] implemented Niching by executing the PSO multiple times, starting with a different swarm each time and changing the objective function after a solution is found to penalize the particles which move towards already found solutions. Parsopoulous et al. [13, 14] used a niching technique based on objective function stretching. In this approach, once an optimum is located, it is removed from the search space by using objective function stretching; the swarm is then reinitialized and allowed to search for other optima. This process continues in an iterative manner until a certain threshold has been achieved. Brits et al. proposed the nbest PSO [2, 4]. Brits has suggested the use of a redefined objective function which rewards a particle when it is closer to any of the possible solutions. In addition to this, spatial neighbourhoods are used to create species. The particles move towards the best solution of their nearest topological neighbourhood, which is defined as the centre of mass of the positions of the n closest particles at any given time. Although nbest PSO can efficiently locate multiple optima, it has difficulty in maintaining niches. In the NichePSO presented by Brits et al. [2, 3, 5], the algorithms consists of a single main swarm with all the particles in the beginning, which then continues to spawn smaller sub-swarms as it converges towards potential solutions. Each sub-swarm thus maintains a separate niche and the particles within the sub-swarm continue to evolve and improve the solution. The sub-swarms that are created are completely independent and no information is exchanged between sub-swarms. The species-based particle swarm optimization (SPSO) proposed by Parrott et al. [12] relies on the concept of speciation to maintain niches and find multiple optima for multimodal optimization problems. During each of the iteration, SPSO identifies particles which are to be used as species seeds. The species seed is considered the neighbourhood best for others particles in that species and has the best fitness.

3. Speciation with PSOGP (SPSOGP)

In this study we extend the PSOGP algorithm to enable it to find multiple solutions for multimodal optimization problems. For this purpose we propose some enhancements to the original PSOGP algorithm. Firstly, we have allowed PSOGP to form species in order for it to be able to simultaneously explore multiple niches and converge on multiple optima. We have also used a partial random initialization technique for the GP, wherein some individuals in the GP population are not randomly initialized but are rather initialized with an individual which is known to have a high fitness. Lastly we have employed a generation gap strategy in order to protect the non-randomly initialized individuals in the GP population from becoming extinct.

For the purpose of creating species, we first need to identify species seeds from the swarm. A species seed has the highest fitness within that species and is considered the neighbourhood best for all other particles in the same species. We employed a similar speciation algorithm that has been used in [6, 11, 12, 15]. We first sort the particles in order of their fitness. We then check all particles starting from the one having best fitness and moving towards the worst fitness. If the particle being checked falls within a pre-specified radius of an already identified species seed, then we make this particle a member of the species

represented by that species seed. However if the particle does not fall within the radius of any other species seed found thus far, then we make this particle a species seed itself. Within each iteration new species seeds are determined and the species are recreated around those species seeds.

We also employed a partial non random initialization technique for initializing the force generating functions of the particles. According to this technique, not all of the individuals were randomly initialized. A small percentage of the individuals were initialized with the standard force generating function of PSO. In order to protect these individuals from becoming extinct during the execution of the algorithm, a generation gap was used which only allowed the randomly initialized individuals to be replaced with their children and preserved the non-randomly initialized particle till the end. A point to be noted is that this only applies to the force generating function, the position and velocity of the particles were randomly initialized just as in the standard PSO. The introduction of these extensions to the PSOGP algorithms also introduced two new parameters, the species radius and the percentage of population which uses the standard force generating function. Optimal values for species radius were taken from literature, whereas the percentage of population which uses the standard force generating function was empirically determined.

4. Experimentation

We compared the performance of SPSOGP with SPSO and NichePSO. SPSOGP could not be compared with PSOGP because PSOGP only found single solutions to the problems and could not be fairly compared with SPSOGP which is capable of finding multiple solutions. In order to test the performance of SPSOGP we have made use of 5 test functions. The same test functions have been used before in [1, 3, 12, 17].

The parameters of the experiment are kept the same as in [3] and [12] to allow a fair comparison. The initial positions of the particles in the swarm were uniformly distributed over the entire search space. No restriction was placed on the velocity. However the position of the particles was restricted to be within the limits specified for each function. 10% of the particles were initialized with the standard force generating function of PSO, whereas the force generating function of the other 90% particles were randomly initialized. 10% of the particles which were initialized with the standard force generating function of PSO did not change their force generating function. The parents were selected based on their fitness (i.e. particles with greater fitness had a greater chance of being selected for reproduction). With each iteration the force generating function of 90% of the particles were changed. The algorithm was run 50 times, each time a maximum of 2000 iterations were allowed. The swarm had 30 particles. The force generating function of 10% particles were initialized non-randomly. The crossover operator was used with probability 0.9 and the mutation with probability 0.1. A generation gap of 0.9 was used and the species radius was set to 2.0 for F5 and 0.05 for others.

The acceptable error was set at 0.0001. Any run with an error less than the threshold was considered successful. SPSOGP was terminated when either the optimal solution was found or it had executed 2000 iterations. The accuracy (mean and standard deviation), success rate, and optimum rate were recorded. Accuracy is calculated by taking the average of the fitness differences between all known global optima to their closest species seeds [12]. The success rate was calculated by counting the number of runs in which the error value was less than the threshold i.e. 0.0001 at the end of the run. Similarly the optimum rate was calculated by counting the number of runs in which SPSOGP was able to find the exact optimum, i.e. the error value at the end of the run was zero. It is important to note the difference between success rate and optimum rate. One of the strengths of PSOGP and SPSOGP is that they are able to locate the exact optimum with a higher frequency, which other algorithms cannot achieve. In addition to comparing the performance of SPSOGP with other SPSO and NichePSO algorithms, we were also interested in observing the average number of species created during execution of the algorithm. We recorded the number of species at the end of each run and took the average over 50 runs. We also recorded all of the force generating functions which were used for each particle for a single run of the algorithm. We calculated the frequency of use of each force generating function and analyzed the more frequently occurring ones.

5. Results

Table 1 presents the results obtained for accuracy of SPSOGP. The accuracy was calculated by taking the average of the error of the closest particle to an optima and then averaging the results over 50 runs. The accuracy of SPSO and NichePSO for finding global optima is also presented here. The accuracy of SPSO and NichePSO are taken from [12] and [3] respectively. SPSOGP was able to find the global optima with higher accuracy as compared to NichePSO for all 5 test functions. SPSOGP also showed better or equal performance as compared to SPSO for all test functions.

TABLE 2 ACCURACY OF SPSOGP (MEAN AND STD ERR) AVERAGED OVER 50 RUNS, SUCCESS RATE, OPTIMUM RATE AND AVERAGE NUMBER OF SPECIES. SPSO RESULTS ARE QUOTED FROM [12] AND NICHEPSO RESULTS ARE QUOTED FROM [3].

Function	SPSOGP (Accuracy)	Success rate	Optimum rate	Avg. no. of species	SPSO (Accuracy)	NichePSO (Accuracy)
F1	0.00 ± 0.00	100 %	100 %	5.43	0.00 ± 0.00	7.68E-05 ± 3.11E-05
F2	0.00 ± 0.00	100 %	100 %	5.68	4.00E-17 ± 2.26E-17	9.12E-02 ± 9.09E-03
F3	4.80E-17 ± 1.59E-16	100 %	88 %	5.32	3.20E-14 ± 3.20E-14	5.95E-06 ± 6.87E-06
F4	1.72E-07 ± 0.00	100 %	0 %	4.18	1.72E-07 ± 0.00	8.07E-02 ± 9.45E-03
F5	2.16E-15 ± 2.57E-15	100 %	6 %	8.24	2.19E-09 ± 2.19E-09	4.78E-06 ± 1.46E-06

Table 1 also presents the results obtained for the convergence of SPSOGP. SPSOGP was able to find the global optima for all 5 test functions 100% of the time. A success rate of 100% has been reported for both SPSO [12] and NichePSO [3]. For test functions F1 and F2, SPSOGP was able to locate all the exact optima (having error value of 0) 100 % of the times. For F3 SPSOGP was able to locate the exact optima 88 % of the times. For F4 SPSOGP was not able to locate the exact optima and for F5 the exact optima was located 6 % of the times. Table 2 also shows the average number of species maintained by SPSOGP. For all test functions the average number of species is greater than the number of global optima present in the search space.

TABLE 2 USAGE FREQUENCY OF FORCE GENERATING FUNCTIONS AND THEIR DESCRIPTION

Equation	F1	F2	F3	F4	F5	Description
+*c-px*c-gx	11.00%	10.91%	11.01%	11.09%	13.19%	The standard PSO equation
x	2.75%	3.01%	2.55%	2.68%	2.23%	The current position of the particle
c	2.67%	2.62%	2.49%	2.74%	2.27%	A random number
p	2.03%	1.82%	1.81%	1.73%	1.19%	The current personal best of the particle
g	1.54%	2.00%	1.51%	1.67%	1.12%	The current neighborhood best of the particle
v	1.05%	0.96%	0.79%	0.77%	0.18%	The current velocity of the particle
-px	0.60%	0.67%	0.57%	0.62%	0.91%	The cognition component of the standard PSO equation without the random number
-gx	0.58%	0.74%	0.52%	0.63%	0.87%	The social component of the standard PSO equation without the random number
*c-gx	0.28%	0.45%	0.29%	0.41%	0.69%	The social component of the standard PSO equation
*c-px	0.44%	0.35%	0.44%	0.38%	0.68%	The cognition component of the standard PSO equation

Table 2 lists the 10 most frequently used force generating functions during a single execution of SPSOGP for all of the five test functions. The table lists the percentage of times that a particular force generating functions was used to update the velocity of the particle. The force generating functions are listed in pre-order notation and represent the change in the velocity of the particle. An explanation of the force generating functions is also given in Table 2. For all of the test functions similar force generating functions were evolved more frequently. The standard PSO equation is the most frequently used, this was expected because we had initialized 10% of the particles with the standard PSO equation and had set the generation gap to 0.9 so that these 10% particles did not become extinct. So the frequency of use of the standard PSO equation was expected to be higher than 10%. One of the frequently occurring force generating functions is c, meaning that a random number was used to update the velocity. This is interesting because as a result of this the velocity of the particles was randomly changed with a higher frequency. We can also see the social component of the classical PSO (-gx and *c-gx) being used quite frequently. The cognition component of the classical PSO (-px and *c-px) also occurs with higher frequency. Another interesting equation which is evolved frequently is v, meaning that the current velocity of the particle is added to its velocity causing the velocity to be doubled. Hence we can deduce that quite often the particle is looking to accelerate and double

its velocity. We can conclude that although a large number of random force generating functions were evolved during the execution of SPSOGP, the useful equations were evolved more often than others.

6. Conclusion

PSOGP was able to thoroughly explore the whole search space and find the optimum with great success by using a separate force generating function for each particle which resulted in the introduction of the much need diversity in the swarm. With the addition of evolution of these functions PSOGP was able to achieve significantly faster convergence [16]. In this study we further extended PSOGP by using a partial non-random initializing strategy and generation gap which allowed some of the particles in the swarm to always use the standard PSO equation for updating their velocity. We also incorporated a speciation technique to allow PSOGP to form species. The resulting SPSOGP algorithm has shown promising results when tested on five multimodal test functions. SPSOGP is able to locate multiple optima with high accuracy. In future we would be looking to further extend SPSOGP to allow it to tackle dynamic multimodal optimization problems.

7. References

[1] D. Beasley, D.R. Bull, R.R. Martin. A Sequential Niching Technique for Multimodal Function Optimization. Evol. Comput., vol. 1(2), 1993, pp. 101-125, MIT Press.

[2] R. Brits. *Niching Strategies for Particle Swarm Optimization.* Master's thesis, Department of Computer Science, University of Pretoria 2002.

[3] R. Brits, A.P. Engelbrecht, F. van den Bergh. A Niching Particle Swarm Optimizer. *Proc. of 4th Fourth Asia-Pacific Conf. on Simulated Evolution and Learning,* 2002, pp. 692-696.

[4] R. Brits, A.P. Engelbrecht, F. van den Bergh. Solving Systems of Unconstrained Equations using Particle Swarm Optimization. *Proc. of IEEE Conf. on Systems, Man and Cybernetics,* vol. 3, 2002, pp. 102-107.

[5] R. Brits, A.P. Engelbrecht, F. van den Bergh. *Niche Particle Swarm Optimization.* Technical report, Department of Computer Science, University of Pretoria 2005.

[6] A.D. Cioppa, C. De Stefano, A. Marcelli. Where Are the Niches? Dynamic Fitness Sharing. *IEEE Trans. on Evol. Comput.*, vol. 11(4), 2007, pp. 453-465.

[7] A.P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence.* John Wiley & Sons 2006.

[8] I.N. Kassablidis, M.A. El-Shurkawi, R.J. Marks. Border Identification for Power Systems Security Assesment using Neural Network Inversion: An Overview. *Proc. of IEEE Congress on Evol. Comput.,* vol. 2, 2002, pp. 1075-1079. IEEE Press.

[9] I.N. Kassablidis, M.A. El-Shurkawi, R.J. Marks, L.S. Moulin, A.P. Alves da Silva. Dynamic Security Border Identification using enhanced Particle Swarm Optimization. *IEEE Trans. on Power Systems,* vol. 17(3), 2002, pp. 723-729.

[10] J. Kennedy, R. Eberhart. Particle Swarm Optimization. *Proc. of IEEE Int. Conf. on Neural Networks,* vol.4, 1995, pp. 1942-1948. Perth, Australia.

[11] J.P. Li, M.E. Balazs, G. Parks, P.J. Clarkson. A species conserving genetic algorithm for multimodal function optimization. *Evol. Comput.,* vol. 10(3), 2002, pp. 207-234.

[12] D. Parrott, X. Li. Locating and tracking multiple dynamic optima by a particle swarm model using speciation. *IEEE Trans. Evol. Comput.,* vol. 10(4), 2006, pp. 440-458.

[13] K.E. Parsopoulos, V.P. Plagianankos, G.D. Magoulas, M.N. Vrahatis. Stretching Technique for Obtaining Global Minimizers through Particle Swarm Optimization. *Proc. of IEEE Workshop on Particle Swarm Optimization,* 2001, pp. 22-29.

[14] K.E. Parsopoulos, M.N. Vrahatis. Modification of the Particle Swarm Optimizer for Locating all the Global Minima. *Proc. of International Conference on Artificial Neural Networks and Genetic Algorithms,* 2001, pp. 324-327.

[15] A. Petrowski. A clearing procedure as a niching method for genetic algorithms. *Proc. of IEEE Int. Conf. Evol.*

Comput., 1996, pp. 798-803.

[16] M. Rashid, A.R. Baig. Adaptable Evolutionary Particle Swarm Optimization. *Proc. of 3rd Int. Conf. on Innovative Computing Information and Control, ICICIC2008,* 2008, pp. 602.

[17] W.M. Spears. Simple Subpopulation scheme. *Proc. of Evol. Programming Conf.,* 1994, pp. 296-307.