A Parallel Particle Swarm Optimization Algorithm With Communication Strategies

Shu-Chuan Chu[1] **John F. Roddick**[1] **Jeng-Shyang Pan**[2]

[1] School of Informatics and Engineering,
Flinders University of South Australia,
PO Box 2100, Adelaide 5001, South Australia.
Email: {`shuchuan.chu, roddick`}`@infoeng.flinders.edu.au`

[2] Department of Electronic Engineering,
National Kaohsiung University of Applied Sciences,
415 Chien-Kung Rd, Kaohsiung, Taiwan.
Email: `jspan@cc.kuas.edu.tw`

Abstract

A parallel version of the particle swarm optimization (PPSO) algorithm is presented together with three communication strategies which can be used according to the independence of the data. The first strategy is designed for the parameters of solutions that are independent or are only loosely correlated such as the Rosenbrock and Rastrigrin functions. The second communication strategy can be applied to those parameters that are more strongly correlated such as the Griewank function. In cases where the properties of the parameters are unknown, a third hybrid communication strategy can be used. Experimental results demonstrate the usefulness of the proposed PPSO algorithm.

Keywords: Parallel particle swarm optimization. PPSO. Communication Strategies. Rosenbrock, Rastrigrin and Griewank functions.

1 Introduction

The particle swarm optimization (PSO) algorithm is based on the evolutionary computation technique (Eberhart & Kennedy 1995, Kennedy & Eberhart 1995). PSO is a population based evolutionary algorithm and has similarities to the general evolutionary algorithm. However, PSO is motivated from the simulation of social behavior which differs from the natural selection scheme of genetic algorithms (Goldberg 1989, Davis 1991, Gen & Cheng 1997). The metaphor is that of multiple collections (a swarm) of objects moving in space and thus objects are said to possess position and velocity and are influenced by the others in the swarm.

PSO processes the search scheme using populations of particles which correspond to the use of individuals in genetic algorithms. Each particle is equivalent to a candidate solution of a problem. The particle will move according to the adjusted velocity that is based on the corresponding particles experience and the experience of its companions. For the D-dimensional function $f(.)$, the ith particle for the tth iteration can be represented as $X_i^t = (x_i^t(1), x_i^t(2), \ldots, x_i^t(D))$.

Assume that the best previous position of the ith particle at the tth iteration is represented as $P_i^t = (p_i^t(1), p_i^t(2), \ldots, p_i^t(D))$, then $f(P_i^t) \leq f(P_i^{t-1}) \leq \ldots \leq f(P_i^1)$. The velocity of the ith particle at the tth iteration can be represented as $V_i^t = (v_i^t(1), v_i^t(2), \ldots, v_i^t(D))$. $G^t = (X^t(1), X^t(2), \ldots, X^t(D))$ is defined as the *best* position amongst all particles from the first iteration to the tth iteration, where *best* is defined by some function of the swarm.

The original particle swarm optimization algorithm can be expressed as follows:

$$V_i^{t+1} = V_i^t + C_1.r_1.(P_i^t - X_i^t) + C_2.r_2.(G^t - X_i^t) \quad (1)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}, i = 0, \ldots N-1 \quad (2)$$

where N is particle size and $-V_{max} \leq V_i^{t+1} \leq V_{max}$, ($V_{max}$ is the maximum velocity), r_1 and r_2 are random numbers, $0 \leq r_1, r_2 \leq 1$. A discrete binary version of the particle swarm optimization algorithm was also proposed by Kennedy and Eberhart (1997).

The particle swarm optimization algorithm has been applied *inter alia* to optimize reactive power and voltage control (Fukuyama & Yoshida 2001) and human tremor (Eberhart & Hu 1999). A modified version of the particle swarm optimizer (Shi & Eberhart 1998) and an adaption using the inertia of the modified particle swarm (Shi & Eberhart 1999) have also been presented. The latter version of the modified particle swarm optimizer (Shi and Eberhart 1999) can be expressed as

$$V_i^{t+1} = W^t.V_i^t + C_1.r_1.(P_i^t - X_i^t) + C_2.r_2.(G^t - X_i^t) \quad (3)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}, i = 0, \ldots N-1 \quad (4)$$

where W^t is the inertia at the tth iteration. Shi and Eberhart (2001) have also applied fuzzy theory to adapt the particle swarm optimization algorithm. Finally, the explosion, stability and convergence of the PSO is analyzed by Clerc and Kennedy (2002).

In this paper, the parallel structure for particle swarm optimization is under study and three communication strategies are presented based on different solution space.

2 Parallel Particle Swarm Optimization

Parallel processing is concerned with producing the same results using multiple processors with the goal of reducing the running time. The two basic parallel processing methods are pipeline processing and data parallelism. The principle of pipeline processing is to separate the problem into a cascade of tasks where

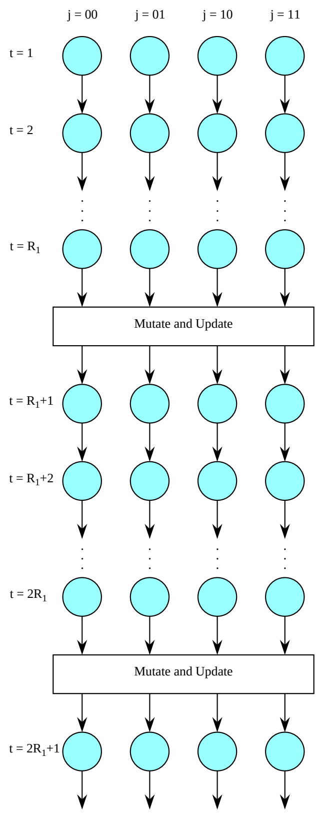

Figure 1: Communication Strategy for loosely correlated parameters.

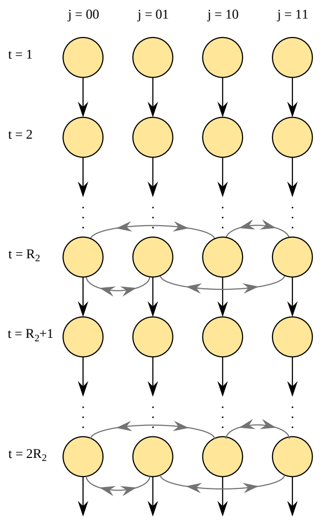

Figure 2: Communication Strategy for strongly correlated parameters.

each of the tasks is executed by an individual processor. Data are transmitted through each processor which executes a different part of the process on each of the data elements. Since the program is distributed over the processors in the pipeline and the data moves from one processor to the next, no processor can proceed until the previous processor has finished its task.

Data parallelism is an alternative approach which involves distributing the data to be processed amongst all processors which then executes the same procedure on each subset of the data. Data parallelism has been applied fairly widely including to genetic algorithms.

The *parallel genetic algorithm* (PGA) works by dividing the population into several groups and running the same algorithm for each group using different processors (Cohoon, Hedge, Martine & Richards 1987). However, the purpose of applying parallel processing to genetic algorithms is more than just as a hardware accelerator. Rather a distributed formulation is developed which gives better solutions with less overall computation. In order to achieve this, a level of communication between the groups is performed every some fixed number of generations. That is, the parallel genetic algorithm periodically selects promising individuals from each subpopulation and migrates them to different subpopulations. With this migration (communication), each subpopulation will receive some new and promising chromosomes to replace the worst chromosomes in a subpopulation. This helps to avoid premature convergence. The parallel genetic algorithm has been applied to vector quantization based communication via noisy channels successfully (Pan, McInnes & Jack 1996).

In this paper, the spirit of the data parallelism method is utilised to create a *parallel particle swarm optimization* (PPSO) algorithm.

It is difficult to find an algorithm which is efficient and effective for all types of problem. As shown in previous work by Shi and Eberhart (2001), the fuzzy adaptive particle swarm optimization algorithm is effective for solutions which are independent or are loosely correlated such as the generalized Rastrigrin or Rosenbrock functions. However, it is not effective when solutions are highly correlated such as for the Griewank function. Our research has indicated that the performance of the PPSO is also highly dependent on the level of correlation between parameters and the nature of the communication strategy. Three communication strategies are thus presented and experiments are carried out which show the utility of each strategy.

The mathematical form of the parallel particle swarm optimization algorithm can be expressed as follows:

$$V_{i,j}^{t+1} = W^t.V_{i,j}^t + C_1.r_1.(P_{i,j}^t - X_{i,j}^t) + C_2.r_2.(G_j^t - X_i^t) \tag{5}$$

$$X_{i,j}^{t+1} = X_{i,j}^t + V_{i,j}^{t+1} \tag{6}$$

$$f(G^t) \le f(G_j^t) \tag{7}$$

where $i = 0, \ldots N_j - 1, j = 0, \ldots S - 1$, S $(= 2^m)$ is the number of groups (and m is a positive integer), N_j is the particle size for the jth group, $X_{i,j}^t$ is the ith particle position in the jth group at the tth iteration, is the velocity of the ith particle in the jth group at the tth iteration, G_j^t is the best position among all particles of the jth group from the first iteration to the tth iteration and G^t is the best position among all particles in all groups from the first iteration to the tth iteration.

As discussed above, three communication strategies have been developed for our parallel particle swarm optimization algorithm. The first communication strategy shown in Figure 1 is designed based on the observation that if the parameters are independent or are only loosely correlated, then the better particles may get good results quite quickly. Thus multiple copies of the best particles in each group G^t are mutated and those mutated particles migrate and replace the worst particles in the other groups every R_1 iterations.

However, if the parameters of the solution are loosely correlated the better particles in each group may not get optimum results particularly quickly. In

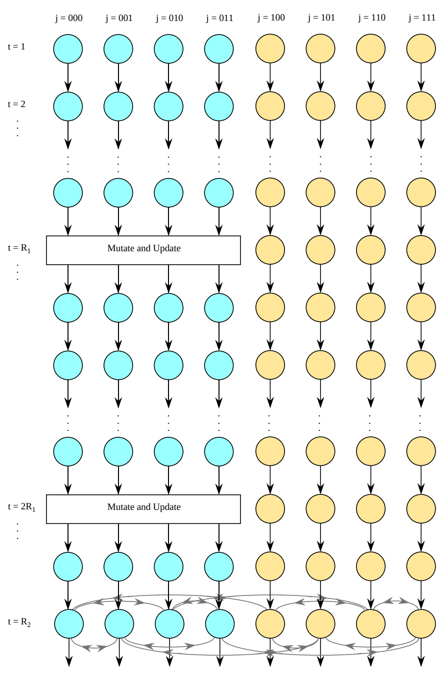

Figure 3: A General Communication Strategy for unknown correlation between parameters.

this case, a second communication strategy may be applied as depicted in Figure 2. This strategy is based on self-adjustment in each group. The best particle in each group G_j^t is migrated to its neighbour groups to replace some of the more poorly performing particles every R_2 iterations.

When the correlation property of the solution space is known the first and second communication strategies work well. However, they can perform poorly if applied in the wrong situation. In the cases where the correlation property is unknown, a hybrid communication strategy can be applied. The hybrid communication strategy separates the groups into two subgroups with the first subgroup applying the first communication strategy every R_1 iterations and all groups applying the second communication strategy every R_2 iterations as depicted in Figure 3.

The complete parallel particle swarm optimization (PPSO) algorithm with its three communication strategies is as follows:

1. **Initialization:** Generate N_j particles $X_{i,j}^t$ for the jth group, $i = 0, \ldots N_j - 1, j = 0, \ldots S - 1$, S is the number of groups, N_j is the particle size for the jth group and t is the iteration number. Set $t = 1$.

2. **Evaluation:** The value of $f(X_{i,j}^t)$ of every particle in each group is evaluated.

3. **Update:** Update the velocity and particle positions using equations (5), (6) and (7).

4. **Communication:** Three possible communication strategies are as follows:

   Strategy 1: Migrate the best particle among all particles G^t to each group and mutate G^t to replace the worse particles in each group and update G_j^t with G^t for each group for every R_1 iterations.

   Strategy 2: Migrate the best particle position G_j^t of the jth group to the qth groups to substitute some worse particles in each receiving group for every R_2 iterations. Here $q = j \oplus 2^m$, $j = 0, \ldots S - 1$, $m = 0, \ldots n - 1$ and $S = 2^n$.

   Strategy 3: Separate the groups into two subgroups. Apply communication strategy 1 to subgroup 1 every R_1 iterations and communication strategy 2 to both subgroup 1 and subgroup 2 for every R_2 iterations.

5. **Termination:** Step 2 to step 5 are repeated until the predefined value of the function or the number of iterations have been reached. Record the best value of the function $f(G^t)$ and the best particle position among all particles G^t.

3 Experiments

Let be an n-dimensional real-value vector. The Rosenbrock function is as following:

$$f_1(X) = \sum_{i=1}^{n}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2). \qquad (8)$$

The second function is the generalized Rastrigrin function which can be expressed as:

$$f_2(X) = \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i)^2 + 10). \qquad (9)$$

The third function is the generalized Griewank function as follows:

$$f_3(X) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \qquad (10)$$

The first experiment was carried out to test the performance of the first PPSO communication strategy which applies when the parameters of the solution are loosely correlated such as for the Rastrigrin and Rosenbrock functions. The second experiment tested the performance for the second strategy for use when the parameters of the solution are more strongly correlated as is the case for the Griewank function. The final experiment tested the performance of the third communication strategy for all the three functions. All three experiments are compared with the linearly decreasing inertia weight of the PSO for 50 runs (Shi & Eberhart 1999).

The parameters of the functions for PSO and PPSO were set as in Table 1. We did not limit the value of X, C_1 and C_2 were set to 2, the maximum number of iterations was 2000, $W_t^0 = 0.9$, $W_t^{2000} = 0.4$ and the number of dimensions was set to 30.

Function	Asymmetric Initialization Range	V_{max}
f_1	$15 \le x_i \le 30$	100
f_2	$2.56 \le x_i \le 5.12$	10
f_3	$300 \le x_i \le 600$	600

Table 1: Asymmetric initialization ranges and V_{max} values.

For fair comparison, the number of groups $\times$ the number of particles per group was kept constant – the particle size for the PSO was 160 and was 8×20, 4×40 and 2×80 for PPSO. For the first experiment, the number of iterations for communication was set to 20 and the best particle position is migrated and mutated to substitute the particles of the receiving group at a rate of 25%, 50%, 75% and 100%. As shown in Table 2 and Table 3, the first communication strategy is effective for the parameters of solution that are independent or loosely correlated.

Migration	PSO	PPSO(2,80)	PPSO(4,40)	PPSO(8,20)
25%	108.7	65.38	98.56	75.95
50%	108.7	75.99	61.10	67.18
75%	108.7	61.19	64.51	59.96
100%	108.7	68.44	60.20	50.88

Table 2: Performance comparison of PSO and PPSO with the first communication strategy for Rosenbrock function.

Migration	PSO	PPSO(2,80)	PPSO(4,40)	PPSO(8,20)
25%	24.54	16.88	17.91	16.06
50%	24.54	15.12	12.88	12.84
75%	24.54	12.88	11.18	11.02
100%	24.54	11.24	10.51	10.03

Table 3: Performance comparison of PSO and PPSO with the first communication strategy for Rastrigin function.

For the second experiment, the number of iterations for communication is set to 100 and the number of worse particles substituted at each receiving group is set to 1 and 2. Experimental results are shown in Table 4, the second communication strategy for 8 groups may improve the performance by up to 66%.

Migration	PSO	PPSO(2,80)	PPSO(4,40)	PPSO(8,20)
1-copy	0.01191	0.01137	0.00822	0.00404
2-copy	0.01191	0.01028	0.01004	0.00601

Table 4: Performance comparison of PSO and PPSO with the second communication strategy for Griewank function.

For the third experiment, the parameters are the same as the first and second experiments. 50% of particles are substituted in the receiving group for the first communication strategy and 2 particles are substituted in the receiving group for the second communication strategy. As shown in Table 5, the hybrid communication strategy can be effective for all three functions.

Function	PSO	PPSO(4,40)	PPSO(8,20)
Rosenbrock	108.739	76.587	82.622
Rastrigin	25.544	18.626	19.144
Griewank	0.011910	0.010527	0.009886

Table 5: Performance comparison of PSO and PPSO with the third communication strategy.

4 Conclusions

In this paper, a parallelised version of the particle swarm optimization scheme is presented. Three communication strategies for PPSO are presented which can be used according to the strength of the correlation of parameters. A third strategy is suggested in cases in which the characteristics of the parameters are unknown. Experimental results demonstrate the usefulness of the parallel particle swarm optimization algorithm with these three communication strategies.

References

Clerc, M. & Kennedy, J. (2002), 'The particle swarm-explosion, stability, and convergence in a multi-dimensional complex space', *IEEE Transactions on Evolutionary Computation* **6**(1), 58–73.

Cohoon, J. P., Hedge, S., Martine, W. N. & Richards, D. (1987), Punctuated equilibria: a parallel genetic algorithm, *in* 'Second International Conference on Genetic Algorithms', pp. 148–154.

Davis, L., ed. (1991), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York.

Eberhart, R. & Hu, X. (1999), Human tremor analysis using particle swarm optimization, *in* 'Congress on Evolutionary Computation', pp. 1927–1930.

Eberhart, R. & Kennedy, J. (1995), A new optimiser using particle swarm theory, *in* 'Sixth International Symposium on Micro Machine and Human Science', pp. 39–43.

Fukuyama, Y. & Yoshida, H. (2001), A particle swarm optimization for reactive power and voltage control in electric power systems, *in* 'Congress on Evolutionary Computation', pp. 87–93.

Gen, M. & Cheng, R. (1997), *Genetic algorithm and engineering design*, John Wiley and Sons, New York.

Goldberg, D. E. (1989), *Genetic algorithm in search, optimization and machine learning*, Addison-Wesley.

Kennedy, J. & Eberhart, R. (1995), Particle swarm optimization, *in* 'IEEE International Conference on Neural Networks', pp. 1942–1948.

Kennedy, J. & Eberhart, R. (1997), A discrete binary version of the particle swarm algorithm, *in* 'IEEE International Conference on Computational Cybernetics and Simulation', pp. 4104–4108.

Pan, J., McInnes, F. & Jack, M. (1996), 'Application of parallel genetic algorithm and property of multiple global optima to vq codevector index assignment for noisy channels', *Electronics Letters* **32**(4), 296–297.

Shi, Y. & Eberhart, R. (1998), A modified particle swarm optimizer, *in* 'IEEE World Congress on Computational Intelligence', pp. 69–73.

Shi, Y. & Eberhart, R. (1999), Empirical study of particle swarm optimization, *in* 'Congress on Evolutionary Computation', pp. 1945–1950.

Shi, Y. & Eberhart, R. (2001), Fuzzy adaptive particle swarm optimization, *in* 'Congress on Evolutionary Computation', pp. 101–106.