Int. J. Open Problems Compt. Math., Vol. 2, No. 4, December 2009
ISSN 1998-6262; Copyright © ICSRS Publication, 2009
www.i-csrs.org

Hybrid PSO and GA for Global Maximization

K. Premalatha and A.M. Natarajan

Kongu Engineering College, Erode, India
e-mail: kpl_barath@yahoo.co.in
Bannari Amman Institute of Technology, Erode, India

Abstract

This paper present the hybrid approaches of Particle Swarm Optimization (PSO) with Genetic Algorithm (GA). PSO and GA are population based heuristic search technique which can be used to solve the optimization problems modeled on the concept of Evolutionary Approach. In standard PSO, the non-oscillatory route can quickly cause a particle to stagnate and also it may prematurely converge on suboptimal solutions that are not even guaranteed to be local optimum. In this paper the modification strategies are proposed in PSO using GA. Experiment results are examined with benchmark functions and results show that the proposed hybrid models outperform the standard PSO.

Keywords: *Convergence, GA, Optimal solution, PSO, Stagnation.*

1 Introduction

The basic optimization problem is that of minimizing or maximizing an objective function subject to constraints imposed on the variables of that function. The objective function and constraints can be linear or nonlinear; the constraints can be bound constraints, equality or inequality constraints, or integer constraints. Global optimization is the task of finding the absolutely best set of admissible conditions to achieve an objective under given constraints. Global optimization is just a stronger version of local optimization, whose great usefulness in practice is definite. Instead of searching for a locally unemployable feasible point one wants the globally best point in the feasible region.

The global maximization problem can be defined as follows: Given $f: S \rightarrow \Re$ where $S \subseteq \Re^{N_d}$ and N_d is the dimension of the search space S. Find $y \in S$ such that $f(y) \geq f(z), \forall z \in S$. The variable y is called the global maximizer of f and f(y) is called the global maximum. The process of finding the global optimal solution is known as global optimization (Gray et al 1997). A true global optimization algorithm will find y regardless of the selected starting point $z_0 \in S$ (Van den Bergh and Engelbrecht 2002). The variable y_L is called the local maximizer of L because $f(y_L)$ is the largest value within a local neighborhood, L. Mathematically speaking, the variable y_L is a local maximizer of the region L if $f(y_L) \geq f(z), \forall z \in L$ where $L \subset S$.

Every global maximizer is a local maximizer, but a local maximizer is not necessarily a global maximizer y_C of the region C if a starting point z_0 is used with $z_0 \in C$. An optimization algorithm that converges to a local maximizer, regardless of the selected starting point $z_0 \in S$, is called a global convergent algorithm. Generally, a local optimization method is guaranteed to find the local maximizer. In this study, finding global maximum solution using PSO with GAA is proposed. Standard PSO is discussed in Section 2. Section 3 describes an overview of GA approach. Section 4 gives the hybrid approaches of PSO with GA. Section 5 presents the detailed experimental setup and results for comparing the performance of the proposed algorithm with the simple PSO.

2 Particle Swarm Optimization

Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. Particle Swarm Optimization (PSO) incorporates swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior, from which the idea is emerged (Kennedy, 2001) (Clerc , 2002), (Parsopoulos, 2004). PSO is a population-based optimization tool, which could be implemented and applied easily to solve various function optimization problems. As an algorithm, the main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms like Genetic Algorithms (GA) (Goldberg, 1989) Simulated Annealing (SA) (Orosz, 2002), (Triki, 2005) and other global optimization algorithms. For applying PSO successfully, one of the key issues is finding how to map the problem solution into the PSO particle, which directly affects its feasibility and performance.

The original PSO formulae define each particle as potential solution to a problem in D-dimensional space. The position of particle i is represented as

$$X_i = (x_{i1}, x_{i2}, \ldots\ldots\ldots\ldots, x_{iD})$$

Each particle also maintains a memory of its previous best position, represented as

$$P_i = (p_{i1}, p_{i2}, \ldots\ldots\ldots\ldots, p_{iD})$$

A particle in a swarm is moving; hence, it has a velocity, which can be represented as

$$V_i = (v_{i1}, v_{i2}, \ldots\ldots\ldots\ldots, v_{iD})$$

Each particle knows its best value so far (pbest) and its position. Moreover, each particle knows the best value so far in the group (gbest) among pbests. This information is analogy of knowledge of how the other particles around them have performed. Each particle tries to modify its position using the following information:

- the distance between the current position and pbest
- the distance between the current position and gbest

This modification can be represented by the concept of velocity. Velocity of each agent can be modified by the following equation (1) in inertia weight approach (IWA)

$$v_{id} = w * v_{id} + c_1 * r_1 * (P_{id} - X_{id}) + c_2 * r_2 * (P_{gd} - X_{id}) \qquad (1)$$

where, v_{id} - velocity of particle
x_{id} - current position of particle
w - inertia factor,
c_1 - determine the relative influence of the cognitive component
c_2 - determine the relative influence of the social component
p_{id} - *pbest* of particle i,
p_{gd} - *gbest* of the group
r_1, r_2 - random numbers

Where w is called as the inertia factor which controls the influence of previous velocity on the new velocity, r_1 and r_2 are the random numbers, which are used to maintain the diversity of the population, and are uniformly distributed in the interval [0,1]. c_1 is a positive constant, called as coefficient of the self-recognition component, c_2 is a positive constant, called as coefficient of the social component. From equation (1), a particle decides where to move next, considering its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm. In the particle swarm model, the particle searches the solutions in the problem space with a range [−s, s]

The following inertia factor (2) is usually utilized in

$$w = w_{\max} - \frac{w_{\max} - w_{\min}}{iter_{\max}} x\, iter \qquad (2)$$

where, w_{max} - initial weight,
w_{min} - final weight,
$iter_{max}$ - maximum iteration number,
$iter$ - current iteration number.

Using the above equation, diversification characteristic is gradually decreased and a certain velocity, which gradually moves the current searching point close to pbest and gbest can be calculated. The current position (searching point in the solution space) can be modified by means of the equation (3):

$$X_{id} = X_{id} + V_{id} \qquad (3)$$

All swarm particles tend to move towards better positions; hence, the best position (i.e. optimum solution) can eventually be obtained through the combined effort of the whole population.

Maurice Clerc has introduced a constriction factor *k*, (CFA) that improves PSO's ability to constrain and control velocities. *k* is computed as:

$$k = \frac{2}{| 2 - c - \sqrt{c^2 - 4c} |} \qquad (4)$$

Where

$c = c_1 + c_2$ and c > 4

$$V_{id} = k(V_{id} + c_1 * rand() * (P_{id} - X_{id}) + c_2 * rand() * (P_{gd} - X_{id})) \qquad (5)$$

For example, if *c*=4.1, then *k*=0.729. As c increases above 4.0, *k* gets smaller. For example, if *c*=5.0, then *k* =0.38, and the damping effect is even more pronounced.

Figure 1 shows the general flow chart of PSO.

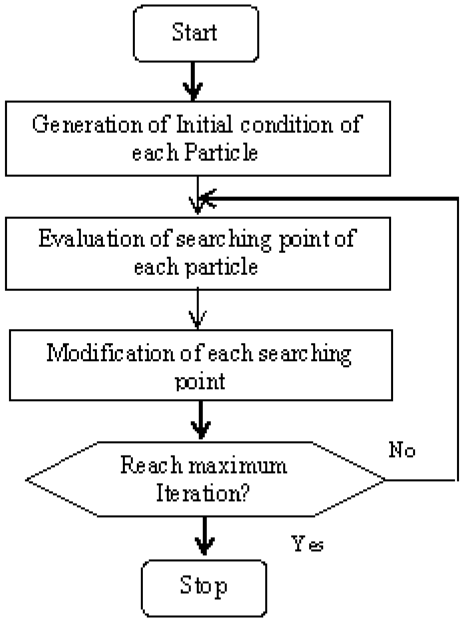

Figure 1 Standard PSO

3 Genetic Algorithm

Genetic Algorithms are a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination and mutation operators to these structures so as to preserve critical information. An implementation of a genetic algorithm begins with a population of (usually random) chromosomes. One then evaluates these structures and allocates reproductive opportunities in such a way that those chromosomes which represent a better solution to the target problem are given more chances to reproduce than those chromosomes which are poorer solutions. The goodness of a solution is typically defined with respect to the current population.

The genetic algorithm can be viewed as two stage process. It starts with the current population. Selection is applied to the current population to create an intermediate population. Then recombination and mutation are applied to the intermediate population to create the next population. The process of going from the current population to the next population constitutes one generation in the execution of a genetic algorithm. (Goldberg 1989) refers to this basic implementation as a Simple Genetic Algorithm (SGA)
In the first generation the current population is also the initial population. There are a number of ways to do selection. After selection has been carried out the

construction of the intermediate population is complete and recombination can occur. This can be viewed as creating the next population from the intermediate population. Crossover is applied to randomly paired strings with a probability denoted *Pc.* A pair of strings is picked with probability Pc for recombination. These strings form two new strings that are inserted into the next population. After recombination, mutation operator is applied. For each bit in the population, is mutated with some low probability *Pm.* Typically the mutation rate is applied with less than 1% probability. In some cases mutation is interpreted as randomly generating a new bit in which case, only 50% of the time will the mutation actually change the bit value. After the process of selection, recombination and mutation, the next population can be evaluated. The process of evaluation, selection, recombination and mutation forms one generation in the execution of a genetic algorithm. Figure 2 shows the Simple Genetic algorithm.

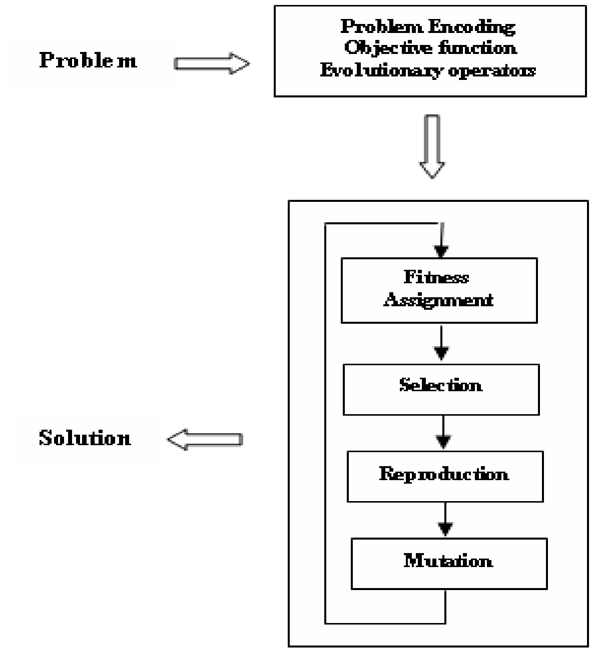

Figure 2 Simple Genetic Algorithm

4 Hybrid PSO with GA

The drawback of PSO is that the swarm may prematurely converge. The underlying principle behind this problem is that, for the global best PSO, particles converge to a single point, which is on the line between the global best and the personal best positions. This point is not guaranteed for a local optimum (Van den Bergh and Engelbrecht 2004). Another reason for this problem is the fast rate of information flow between particles, resulting in the creation of similar particles with a loss in diversity that increases the possibility of being trapped in local optima.

A further drawback is that stochastic approaches have problem-dependent performance. This dependency usually results from the parameter settings in each algorithm. The different parameter settings for a stochastic search algorithm result in high performance variances. In general, no single parameter setting can be applied to all problems. Increasing the inertia weight (w) will increase the speed of the particles resulting in more exploration (global search) and less exploitation (local search) or on the other hand, reducing the inertia weight will decrease the speed of the particles resulting in more exploitation and less exploration. Thus finding the best value for the parameter is not an easy task and it may differ from one problem to another. Therefore, from the above, it can be concluded that the PSO performance is problem-dependent. The problem-dependent performance can be addressed through hybrid mechanism. It combines different approaches to be benefited from the advantages of each approach.

To overcome the limitations of PSO, hybrid algorithms with GA are proposed. The basis behind this is that such a hybrid approach is expected to have merits of PSO with those of GA. One advantage of PSO over GA is its algorithmic simplicity. Another clear difference between PSO and GA is the ability to control convergence. Crossover and mutation rates can subtly affect the convergence of GA, but these can not be analogous to the level of control achieved through manipulating of the inertia weight. In fact, the decrease of inertia weight dramatically increases the swarm's convergence. The main problem with PSO is that it prematurely converges (Van den Bergh and Engelbrecht 2004) to stable point, which is not necessarily maximum. To prevent the occurrence, position update of the global best particles is changed. The position update is done through some hybrid mechanism of GA. The idea behind GA is due to its genetic operators crossover and mutation. By applying crossover operation, information can be swapped between two particles to have the ability to fly to the new search area. The purpose of applying mutation to PSO is to increase the diversity of the population and the ability to have the PSO to avoid the local maxima.

There are three different hybrid approaches are proposed

1 PSO-GA (Type 1) : The *gbest* particle position does not change its position over some designated time steps, the crossover operation is performed on *gbest* particle with chromosome of GA. In this model both PSO and GA are run in parallel.

2 PSO-GA (Type 2) : The stagnated *pbest* particles are change their positions by mutation operator of GA

3 PSO-GA (Type 3): In this model the initial population of PSO is assigned by solution of GA. The total numbers of iterations are equally shared by GA and PSO. First half of the iterations are run by GA and the solutions are given as initial population of PSO. Remaining iterations are run by PSO.

Table 1 shows the benchmark functions taken for experimental analysis. The systems are run on 20 particles with 100 iterations. The inertia weight is w is set as 0.9 and the constants c_1 and c_2 are taken as 2.1. The results obtained from standard PSO and the proposed systems are shown in Figure 3 – Figure 6. For the Shafer and Ackley function PSO-GA (Type 3) outperforms other proposed systems and standard PSO. For the Rosenbrock function, all the three proposed systems are converged on the same value. PSO-GA (Type 1) performs better than PSO-GA (Type 2), PSO-GA (Type 3) and standard PSO for Rastrigin function.

Table 1 Benchmark Functions

Shaffer	$0.5+\frac{\sin\sqrt{x^2+y^2}-0.5}{\left(1.0+0.001\left(x^2+y^2\right)\right)^2}$
Rosenbrock	$\sum_{d=1}^{D}\left(100\left(x_{d+1}-x_d^2\right)^2+\left(x_d-1\right)^2\right)$
Rastrigin	$\sum_{d=1}^{D}\left(x_d^2-10\cos\left(2\prod x_d\right)+10\right)$
Ackley	$-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{d=1}^{D}x_d^2}\right)-\exp\left(\frac{1}{D}\sum_{d=1}^{D}\cos\left(2\prod x_d\right)\right)+20+e$

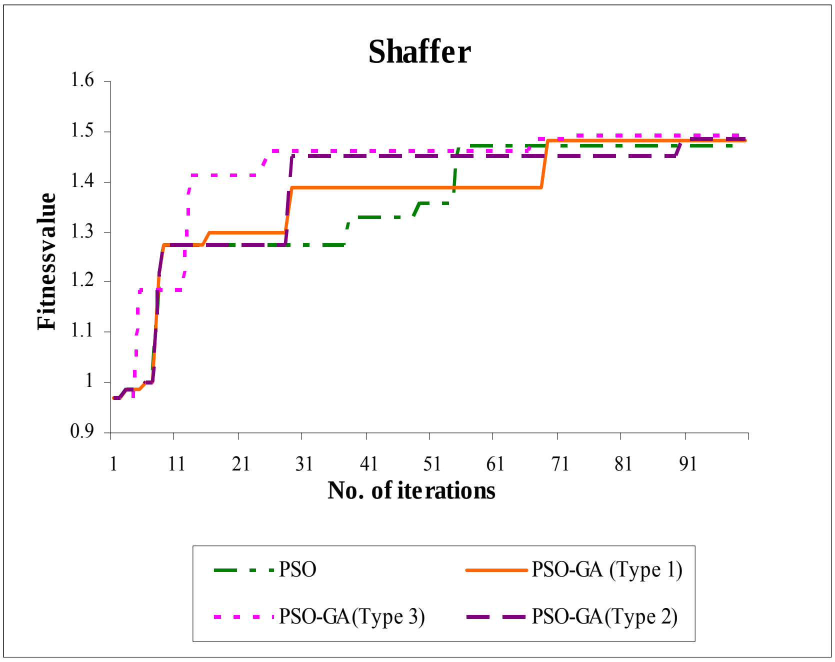

Figure 3 Output for Shaffer function

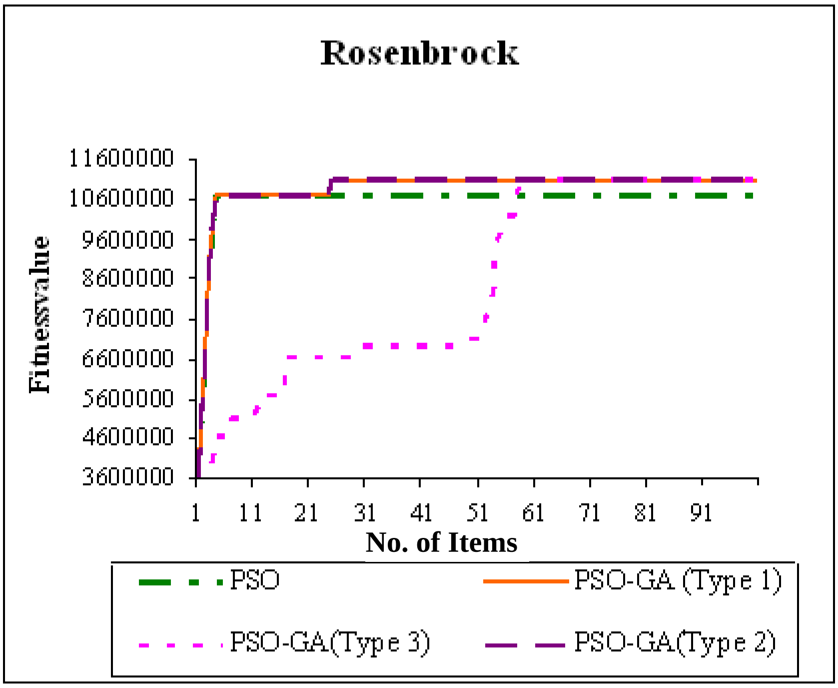

Figure 4 Output of Rosenbrock function

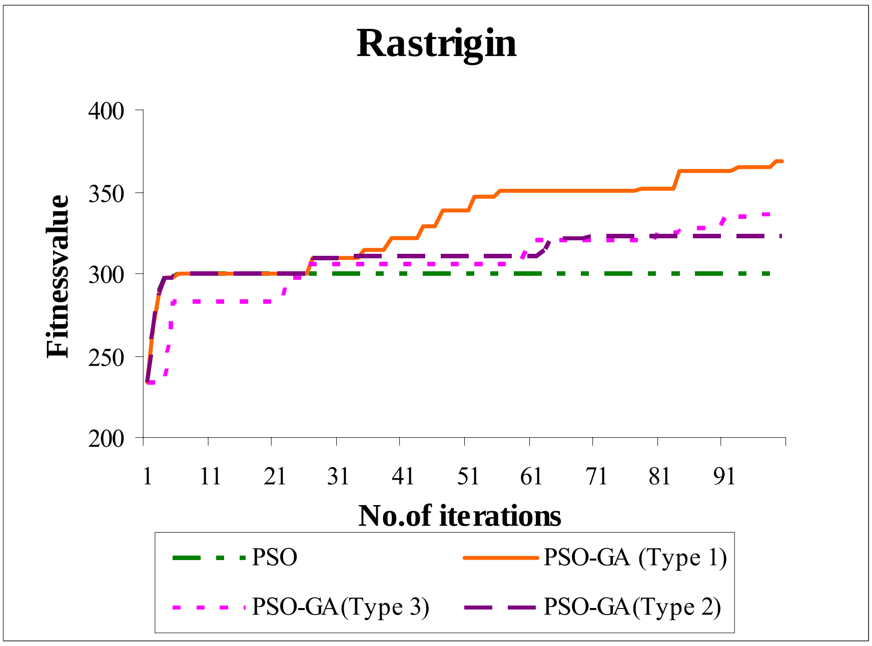

Figure 5 Output for Rastrigin function

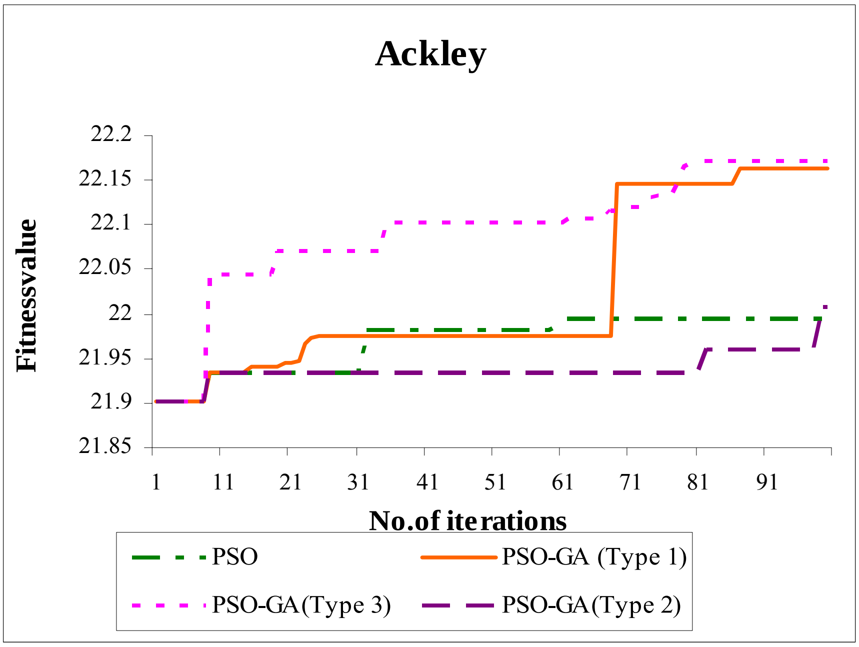

Figure 6 Output for Ackley function

5. Open Problems

In PSO, each particle should be kept in confined space corresponding to the parameter limitations. That decreases the diversity of the particle. If the global best particle does not change its *gbest* position over a period of item then stagnation occurs in the population. Then the solution may be local optimal solution. Due to its stochastic behavior, i.e. it is not possible to find a one way to the global optimum.

6. Conclusion

PSO, which is stochastic in nature and makes use of the memory of each particles as well as the knowledge gained by the swarm as a whole, has been proved to be powerful in solving many optimization problems. The proposed hybrid PSO systems find a better solution without trapping in local maximum, and to achieve faster convergence rate. This is because when the PSO particles stagnate, GA diversifies the particle position even though the solution is worse. In PSO-GA, particle movement uses randomness in its search. Hence, it is a kind of stochastic optimization algorithm that can search a complicated and uncertain area. This makes PSO-GA more flexible and robust. Unlike standard PSO, PSO-GA is more reliable in giving better quality solutions with reasonable computational time, since the hybrid strategy avoids premature convergence of the search process to local optima and provides better exploration of the search process.

References

[1] Gray P., Hart W.E., Painton L., Phillips C., Trahan M. and Wagner J., 'A Survey of Global Optimization Methods', Technical Report, Sandia National Laboratory, 1997.

[2] Van den Bergh F. and Engelbrecht A.P., 'A Cooperative Approach to Particle Swarm Optimization', IEEE Transactions on Evolutionary Computation, 2004, pp. 225-239.

[3] Kennedy J and Eberhart R, Swarm intelligence. Morgan Kaufmann Publishers, Inc., San Francisco, CA,2001.

[4] Clerc M and Kennedy J, The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation,2002, 6(1):58-73.

[5] Goldberg D E, Genetic Algorithms in search, optimization, and machine learning. Addison-Wesley Publishing Corporation, Inc, 1989.

[6] Triki E, Collette Y and Siarry P, A theoretical study on the behavior of simulated annealing leading to a new cooling schedule. European Journal of Operational Research, 2005, 166:pp. 77-92.
[7] Ordonez C, 'Clustering Binary Data Streams with K-means', In Proceedings of DMKD, 2003, pp. 12-19.
[8] Yao, X., Liu, Y., and Lin, G. Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation, 1999, 3(2):82–102.