World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:1, No:7, 2007

Spread Spectrum Code Estimation by Particle Swarm Algorithm

Vahid R. Asghari, and Mehrdad Ardebilipour

Abstract**—**In the context of spectrum surveillance, a new method to recover the code of spread spectrum signal is presented, while the receiver has no knowledge of the transmitter's spreading sequence. In our previous paper, we used Genetic algorithm (GA), to recover spreading code. Although genetic algorithms (GAs) are well known for their robustness in solving complex optimization problems, but nonetheless, by increasing the length of the code, we will often lead to an unacceptable slow convergence speed. To solve this problem we introduce Particle Swarm Optimization (PSO) into code estimation in spread spectrum communication system. In searching process for code estimation, the PSO algorithm has the merits of rapid convergence to the global optimum, without being trapped in local suboptimum, and good robustness to noise. In this paper we describe how to implement PSO as a component of a searching algorithm in code estimation. Swarm intelligence boasts a number of advantages due to the use of mobile agents. Some of them are: Scalability, Fault tolerance, Adaptation, Speed, Modularity, Autonomy, and Parallelism. These properties make swarm intelligence very attractive for spread spectrum code estimation. They also make swarm intelligence suitable for a variety of other kinds of channels. Our results compare between swarm-based algorithms and Genetic algorithms, and also show PSO algorithm performance in code estimation process.

Keywords**—**Code estimation, Particle Swarm Optimization (PSO), Spread spectrum.

I. Introduction

Although spread spectrum communications were initially developed for military applications, they are now widely used for commercial ones, especially for code division multiple access (CDMA), or global positioning systems (GPS) [1]. They are mainly used to transmit at low power without being interfered by jamming, to other users or to multi path propagation. The spread spectrum techniques are useful for secure transmitter, because the receiver has to know the sequence used by the transmitter to recover the transmitter data [2]–[3].

Our purpose is to determine the spreading sequence automatically, whenever the receiver has no knowledge of the transmitter's code. In our previous paper [4], we used Genetic Algorithm (GA), to recover spreading code. GAs have been used to learn complex behaviors characterized by sets of sequential decision rules and we used them for their robustness in solving complex optimization problem, nonetheless, by increasing the length of the code, we will often lead to an unacceptable slow convergence speed. Hence, we have introduced a new method, which is Particle Swarm Optimization (PSO), into code estimation in spread spectrum communication system. In searching process for code estimation, the PSO algorithm has the merits of rapid convergence to the global optimum results, and good robustness to noise. In this paper, we describe how to implement PSO as a component of a searching algorithm in code estimation. Swarm intelligence boasts a number of advantages due to the use of mobile agents. The code estimation performance of the proposed algorithm is examined by computer simulations. The performance measure of interest in this paper is the mean-squared error (MSE) for the code estimation.

The paper is organized as follows. Section II describes the technique of direct sequence spread spectrum (DS-SS). Section III describes the system model used in this paper. Sections IV and V describe the PSO used to implement our proposed code estimator. Our simulation results are presented in section VI. Section VII concludes the paper.

II. DS-SS Technique

In order to spread the signal power over a broadband channel, the direct sequence spread spectrum (DS-SS) technique consists in multiplying the information signal with a periodic pseudo-noise sequence.

Let us consider $b(t)$ the information signal

$$b(t) = \sum_{n=-\infty}^{+\infty} b_n p(t - nT_b) \qquad (1)$$

Where $b_n = \pm 1$ with equal probability and $p(t)$ is a rectangular pulse of duration T_b [4].

Let us note y , the PN sequence of length k ,

$$y = y_0, y_1, \cdots, y_{k-1} \qquad (2)$$

According to the properties of PN sequences, if we assume the receiver does not know this sequence, it can not despread the received signal [4]. So we try to find some results which can despread the received signal, by using PSO algorithm.

Manuscript received July 15, 2005.

Vahid R. Asghari is with the Iran Telecommunication Research Center (ITRC), Tehran, Iran (e-mail: vhasghari@ee.kntu.ac.ir).

Mehrdad Ardebilipour is with the Department of Electrical Engineering, K.N. Toosi University of Technology, Tehran, Iran (e-mail: mehrdad@eetd.kntu.ac.ir).

III. System Description

Typically direct sequence spread spectrum systems use binary or quadrature phase shift keying (BPSK or QPSK) data modulation. Usually the PN sequence is a binary maximal length sequence or a Gold sequence [3].

Although in this method, we can estimate different PN sequences, but here we consider a BPSK data modulation, spread by a Gold sequence. The baseband noise is assumed to be additive, white, Gaussian, and centered.

An interesting method to estimate spreading code is illustrated in [6]. It takes profit of blind identification techniques available for multiple FIR channels. Also In [4], we used Genetic algorithms (GA) to estimate PN sequence. In this method which is based on particle swarm intelligence, we improve the spread of convergence to the global optimum.

IV. PSO Optimization Technique Overview

Particle swarm optimization has its roots in two main component methodologies. Perhaps more obvious are its ties to artificial life (A-life) in general, and to bird flocking, fish schooling, and swarming theory in particular. It is also related, however, to evolutionary computation, and has ties to both genetic algorithms and evolution strategies [7]. Particle swarm optimization comprises a very simple concept, and paradigms are implemented in a few lines of computer code. It requires only primitive mathematical operators, and is computationally inexpensive in terms of both memory requirements and speed [8].

Particle swarm optimization can be used to solve many of the same kind of problems as genetic algorithms (GAs) [8]. This optimization technique does not suffer, however, from some of GA's difficulties; interaction in the group enhances rather than detracts from progress toward the solution. Further, a particle swarm system has a memory, which the genetic algorithm does not have. Change in genetic populations results in destruction of previous knowledge of the problem, except when elitism is employed, in which case usually one or a small number of individuals retain their "identities". In PSO, individuals who fly past optimum are tugged to return toward them; knowledge of good solutions is retained by all particles [9].

Particle swarm intelligence boasts a number of advantages due to the use of mobile agents and stigmergy. These are:

1. *Scalability*: Population of the agents can be adapted according to spreading code size.
2. *Fault tolerance*: Particle swarm intelligence processes do not rely on a centralized control mechanism. Therefore the loss of a few bits or frames does not result in catastrophic failure, but rather leads to graceful, scalable degradation.
3. *Adaptation*: Agents can change, die or reproduce, according to the length of the code changes. But here, we supposed the length of the code is constant.
4. *Speed*: Changes in the systems can be modified very fast.
5. *Modularity*: Agents act independently of other codes of users. It can be used for multiuser systems.
6. *Autonomy*: Little or no human supervision is required.
7. *Parallelism*: Agent operations are inherently parallel.

These properties make particle swarm intelligence very attractive for spread spectrum code estimation.

V. PSO Optimization Technique in Code Estimation

The PSO algorithm, proposed by Kennedy and Eberhart [11], has proved to be very effective in solving global optimization for multidimensional problems in static, noisy, and continuously changing environments [12]. We introduced for the first time the GA technique into spread spectrum code estimation in our previous work [4], and now, we use PSO technique, which has some properties does not exist in GA technique.

In reality, PSO and GA techniques are too similar and by making some changes to GA's algorithm, you have your PSO algorithm. At the beginning, the PSO algorithm randomly initializes a population (called swarm) of individuals (called particles). Each particle represents a single intersection of spreading code. The particles evaluate their position relative to a goal at every iteration. In each iteration, every particle adjusts its trajectory (by its velocity) toward its own previous best position, and toward the previous best position attained by any member of its topological neighborhood. If any particle's position is close enough to the goal function, it is considered as having found the global optimum and the recurrence is ended. Generally, there are two kinds of topological neighborhood structure, corresponding to the global version of PSO (GPSO), and local neighborhood structure, corresponding to the local version of PSO (LPSO). For the global neighborhood structure, the whole swarm is considered as the neighborhood, while for the local neighborhood structure, some smaller number of adjacent members in subswarm is taken as the neighborhood [13].

In the global neighborhood structure, each particle's search is influenced by the best position found by any member of the entire population. In contrast, each particle in the local neighborhood structure is influenced only by parts of the adjacent members. Therefore, the LPSO has fewer opportunities to be trapped in suboptimum than the GPSO.

Generally, the larger the number of particles adopted in PSO, the fewer the opportunities to be trapped in suboptimum, but the greater the time spent searching for the global optimum. In our experiment, 40 particles are used in LPSO, which is a balance between the accuracy required in searching for the global optimum and time consumed. This procedure, whose flowchart is shown in Fig. 1, is iterated a predefined number of consecutive particles.

A. Initialization

Initialization of the PSO is performed at the so-called ($y = 1$)st generation for the first signaling interval, as seen in Fig. 1, by creating p number of candidate solutions, or particles in PSO parlance. For the others iteration, we just use the population of previous iteration. The set of p particles is

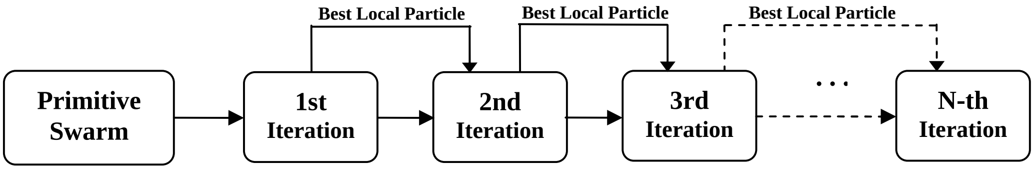

Fig. 2 Flowchart depicting the structure of the iterations

known as a swarm, and p is known as swarm size. These particles represent the unknown variables of interest, which in this case are the estimated PN sequence. Hence, each particle will contain k elements corresponding to the length of the PN sequence.

B. Evaluation and Selection

Associated with the p th combination particle is a so-called figure of merit — more commonly known in PSO as the fitness value — which has to be evaluated, as seen in Fig. 1. The fitness value, denote by $f\left[\hat{y}_n, \tilde{y}_k\right]$ for $k = 1, \cdots, K$ is computed by substituting the elements of both the transmitted string and the k th candidate solution into the objective function or crosscorrelation of them. Let us refer to the elements that constitute the optimal solution as good particles. Any other elements are referred to as bad particles [4]. Intuitively, particles having a high fitness in the sense of crosscorrelation will contain more good elements and hence should be exploited further. At the same time, particles having a low fitness value should be discarded. Then, the particles which are located at the top level of sorted population will be memorized and used for subsequent exploitation and exploration of the solution space.

C. Crossover and Mutation

Crossover and mutation are two different operators which produce one or more new particles. Crossover applies to one or more parents and exchange particle elements (good or bad) with equal probability (p_c) between two different particles, and will constitute the new swarm (population) of the next generation. The mutation operation refers to the alteration of the value of each particle in the offspring with a probability denoted by p_m. In the case of the data string, the mutation process simply inverts the bit value of the element [4].

The PSO algorithm is terminated if there is no improvement in the maximum fitness value of the swarm [4]. In each iteration, as showed in Fig. 2, the PSO uses the previous best particles which were memorized in previous iteration. In this algorithm the rate of convergence and adaptation is increased.

VI. Simulation Results

In this section, our simulation results are presented in order to demonstrate the performance of the proposed code estimator. The channel noise was assumed to be additive, white, Gaussian, centered and real and the data rate (R_b) and the number of chips per bit (p) were assumed to be known by the receiver. The signature sequence was used with a processing gain of $p = 31$.

In order to give an impression of how the PSO manages to estimate the transmitted code over the course of iterations given a population of randomly generated possible solutions at the beginning.

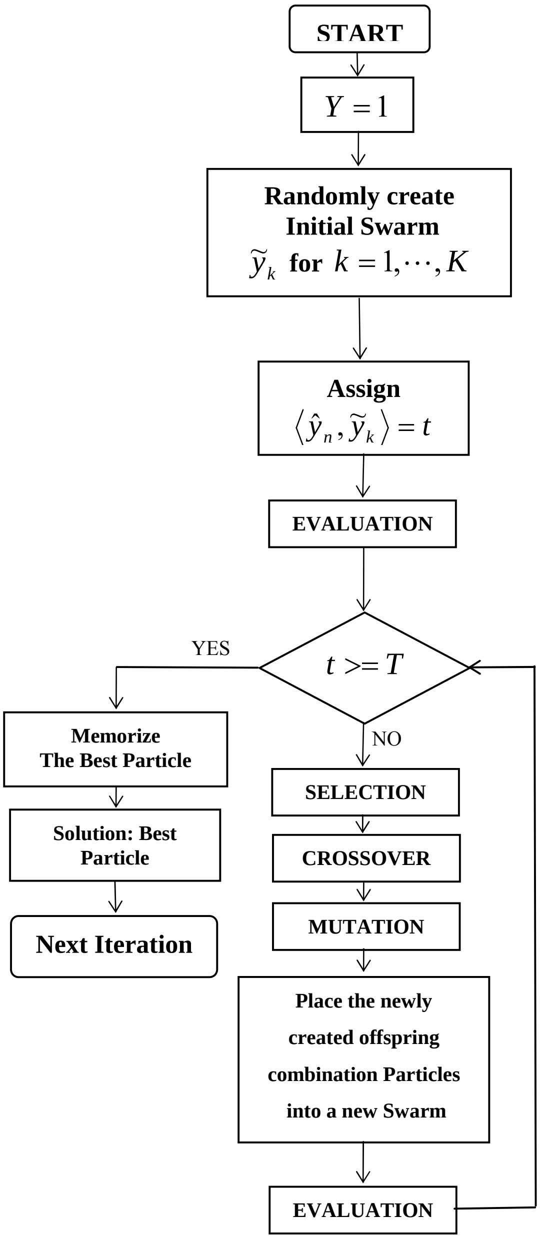
START
$Y = 1$
Randomly create Initial Swarm
$\tilde{y}_k$ for $k = 1, \cdots, K$
Assign
$\langle \hat{y}_n, \tilde{y}_k \rangle = t$
EVALUATION
YES
$t >= T$
NO
Memorize The Best Particle
Solution: Best Particle
Next Iteration
SELECTION
CROSSOVER
MUTATION
Place the newly created offspring combination Particles into a new Swarm
EVALUATION

Fig. 1 Flowchart depicting the structure of the proposed particle swarm optimization (PSO) used to code estimation

The best fitness value of particles in our swarm (population) in some iterations is shown in Fig. 3 at $\xi / N_0 = -5dB$. As we have mentioned in section V, the PSO algorithm will efficiently identify the areas in the solution space, where the optimal solution might be located.

Fig. 3 shows that the entire final searched fitness values in any code estimation process exceed 0.94 for LPSO used as the optimization algorithm. Furthermore, entire the fitness values reach 0.9 within about thirty iterations for LPSO.

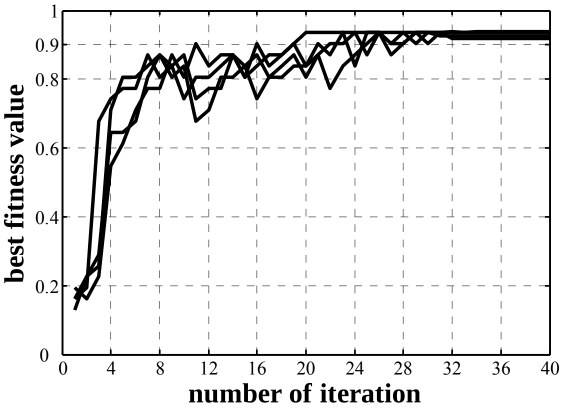

Fig. 3 Best fitness values of the proposed PSO-based estimator for three different try, over forty iterations, $p = 100$, $Y = 10$, and $SNR = -5dB$, $p_C = 0.2$, and $p_m = 0.1$

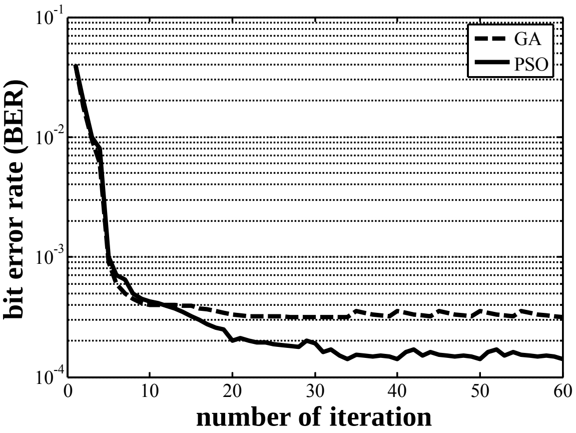

Fig. 4 BER performance of the proposed PSO-based code estimator in compare with GA-based, over 60 iterations where, $p = 100$, $Y = 10$, $p_C = 0.2$, and $p_m = 0.1$

Fig. 4 characterizes the BER performance of proposed estimator in compare with a receiver which uses genetic algorithm (GA) method to estimate the spreading code [4]. It can be seen that the bit error rate (BER) performance of PSO-based code estimator is better than GA-based one. It is because in GA-based code estimator, we try to find the spreading code which has the best fitness value, just in first iteration. Hence, it takes a lot of time [4].

In contrast, in PSO-based code estimator, step by step, we find the spreading code over the course of iterations. In fact, at each iteration, we try to find the code which has the best local fitness value. Hence, it is faster compared with GA-based algorithm.

VII. Conclusion

For the first time, we have introduced the PSO algorithm into spread spectrum code estimator, which showed the desirable features of rapid convergence to the global optimum without being trapped in local suboptimum and robustness to noise. Swarm intelligence however is a new field and much work remains to be done.

Particle swarm optimization is an extremely simple algorithm that seems to be effective for optimizing a wide range of functions. Much further research remains to be conducted on this new concept. The goal in developing it has been to use this system in fading channels.

References

[1] D. Thomas Magill, Francis D. Natali, Gwyn P. Edwards, “Spread Spectrum Technology for Commercial Applications,” *Proceeding of the IEEE*, vol. 82, pp. 572–584, April. 1994.

[2] Raymond. L. Picholtz, Doland L. Schilling, Laurence B. Milstein, “Theory of Spread Spectrum Communications – A Tutorial,” *IEEE Transactions on Communications,* vol. COM-30, pp. 855–884, May. 1982.

[3] John G. Proakis, *Digital communication*, Third Edition, Mac Graw Hill International Editions, 1995.

[4] V. R. Asghari and M. Ardebilipour, “Spread Spectrum Code Estimation by Genetic Algorithms,” *International Journal on Signal Processing*, vol. 1, pp. 301-304, Dec. 2004.

[5] Dilip V. Sarwate, Michael B. Pursley, “Crosscorrelation Properties of Pseudo-random and Related Sequences,” *Proceeding of the IEEE*, vol. 68, pp. 593–619, May. 1980.

[6] Michail K. Tsatsanis, Georgios B. Giannakis, “Blind Estimation of Direct Sequence Spread Spectrum Signals in Multipath,” *IEEE Transactions on Signal Processing,* vol. 45, pp. 1241–1252, May. 1997.

[7] T. Baeck, “Generalized convergence models for tournament and (mu,lambda)-selection,” *Proc. of the Sixth International Conf. on Genetic Algorithms,* pp. 2-7, Morgan Kaufmann Publishers, San Francisco, CA, 1995.

[8] M. Potter, K. De Jong, and J. Grefenstette, “A coevolutionary approach to learning sequential decision rules,” *Proc. of the Sixth International Conf. on Genetic Algorithms,* pp. 366-372, Morgan Kaufmann Publishers, San Francisco, CA, 1995.

[9] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” *Proc. 6th Int. Symp. Micro Machine Human Sci.*, pp. 39–43, 1995.

[10] J. Kennedy and R. C. Eberhart, “Particle Swarm Optimization,” *Proc. IEEE Int. Conf. Neural Networks*, Piscataway, NJ, pp. 1942–1948, 1995.

[11] E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, “Particle swarm optimization for minimax problems,” *Proc. 2002 Congress Evolutionary Computation*, vol. 2, pp. 1576–1581, 2002.

[12] J. Kennedy and R. Mendes, “Population structure and particle swarm performance,” *Proc. 2002 Congress Evolutionary Computation*, vol. 2, pp. 1671–1676, 2002.

Vahid Reza Asghari was born in Iran in 1980. He received B.Eng. degree with honors in electrical and electronics engineering from the Azad University, Iran, in 2002. Recently, he is studying Msc. degree in the electrical engineering in K.N. Toosi University of Technology, Tehran, Iran. His research interests include spread spectrum, Multiuser Detection in CDMA and intelligence methods.

Mehrdad Ardebilipour was born in Iran, on February 1954. He received Bsc. and Msc. degrees in electrical engineering from K.N. Toosi University of Technology, Tehran, Iran, in 1977 and Tarbiat Modarres University, Tehran, Iran, in 1989, respectively. He has also been awarded the degree of Ph.D by the University of Surrey, Guildford, England, in 2001.

Since 2001, he has been assistant professor at K.N. Toosi University of Technology with major interest in the spread- spectrum and multiuser detection.