CHAPTER III

THE IMPROVEMENT OF GA PERFORMANCE

3.1 Background to Improvement of GA Performance

As mentioned in previous chapter, it has been shown that GA can efficiently solve complex optimization problems because of its advantages. However, GA is not a panacea. It usually suffers from the problem of premature convergence and weak exploitation capabilities (Chelouah & Siarry, 2000). Occurrence of premature convergence often makes GA stop at a local optimum, not a global optimum, while weak exploitation capabilities often cause slow convergence of GA. These drawbacks lead GA to fail to obtain the reliable solutions.

In order to enhance the performance of the simple genetic algorithm (SGA), many approaches to developing GA have been suggested in various applications. Each genetic step (as briefed in previous chapter) has been developed. Several authors have developed GA with management at mutation operator. Acevedo and Leboreiro (2004) have proposed mutation shock technique to improve the performance of GA. They have presented, this technique enable GA to speed up the process of finding a better solution. However, the procedure is very effective when only solving small problems. It is ineffective in solving complex optimization problem. Shin, D. J., Kim, J. O., Kim, T. K., Choo, J. B. & Singh, C. (2004) have introduced Tabu search (TS) into mutation step, and this technique is called Genetic-Tabu algorithm (GTA). Because of advantage of TS is that it can find regions lead solution to converge to local optimum, and collect these regions in Tabu list. So, before population is mutated, it is checked by Tabu list. If the population is existed in Tabu list, mutation probability will be increased so that population will jump out from local optimum. Furthermore, GA can be developed at crossover step. Dowsland, K. A., Herbert, E. A., Kendall G. & Burke E. (2004) have presented bound-based crossover. They have introduced a new way of utilizing bound based information to improve the performance of GA.

Moreover, various researchers enhance the GA performance with managing population. Wang, Qian, Yuan, and Yao (1998) have proposed a sub-

population-based distributed evolutionary technique. In this technique, population is divided into a number of sub-populations. Each group is optimized through GA and provides a local optimum. Then, local optimum of each sub-population is communicated through SS-crossover so as to find the global optimum point. Their result have showed that their approach provide fast computational time and stable in converging to global optima. Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C. & Wang, L. M. (2005) have also improved GA performance with managing population. They have proposed a variable population-size (VPGA) technique. This technique not eliminates parents after they provide offspring. The parents have more opportunities to survive if they have higher fitness than offspring. This technique can improve accuracy of solutions. However, it requires much computational time.

In this research, we develop GA with management at the step of generating initial population. The details of this development are discussed in the next topic.

3.2 Strategies for Improvement of GA Performance in This Work

As mentioned in the previous chapter, GA has shown a good performance regarding its ability to search globally. It starts searching with population or multiple points in the search space. Generally, initial population is generated by random sampling. Unfortunately, a random population usually results in the over-sampling in some areas and sparse sampling in others. Consequently, the initial population from random sampling has less uniformity properties—lack of diversity, while the diversity of population is critical to searching efficiency (Haupt, 2004; Wei and Zhao, 2005). Hence, an effective way to improve GA is to enhance the uniformity properties of initial population. One of the approaches is to start generating initial population with uniform distribution, and this approach ensures diversity of the initial population.

Monte Carlo sampling technique is widely used to generate a set of uniformly distribution; however, this approach occasionally brings about large error bounds and variance. In order to cope with the poor precision of the standard Monte

Carlo approach, more efficient sampling methods are being suggested in such application. Two of such approaches are here considered:

- Pseudo-probabilistic simulation, such as Latin hypercube sampling (Iman and Conover, 1982) and Descriptive sampling (Saliby, 1990);

- Deterministic methods, also known as quasi-Monte Carlo methods, based on low discrepancy sequences and generated without any randomness feature (Paskov and Traub, 1995; Traub and Papageorgiou, 1996) such as, Halton sequence sampling (Halton, 1960), Faure sequence sampling (Faure, 1982), and Hammersley sequence sampling (Hammersley, 1960).

In this work, we are interested in Latin hypercube sampling, Faure sequence sampling, and Hammersleay sequence sampling. The detail of these sampling are explained in the next topic.

3.3 Latin Hypercube Sampling (LHS)

Latin hypercube sampling is an alternative to full stratification in high dimensional problems. It generates N of the N^k full-stratification bins in a particularly effective way. Those N points in dimension k are chosen so that all k marginal distributions are perfectly stratified.

The Latin Hypercube Sampling method was first introduced by Mckay, Beckman, and Conover (1979). To generate a Latin hypercube sample of size N in dimension k we can imply the algorithm:

1. Generate a $(N \times k)$ matrix of independent and uniform over $[0,1]$ numbers $U_j^i, j = \overline{1,k}, i = \overline{1,N}$.

2. Compute a vector $\pi_j, j = \overline{1,k}$, independent random permutations of $\{1,2,...,N\}$ sampled uniformly from all $N!$ such permutations.

3. Calculate $V_j^i = \frac{\pi_j(i) - 1 + u_j^i}{N}, j = \overline{1,k}$. (3.1)

4. The N points $V_j^i = (V_1^i, ..., V_k^i), i = \overline{1,N}$, constitute a Latin hypercube sample of size N in dimension k.

Each of these vectors can be transformed into vectors of given distribution. In the Latin Hypercube Sampling method, the range of probable values for each component U_j^i is divided into N segments of equal probability. For example, for the case of dimension k=3 and N=10 segments, the parameter space is divided into $10 \times 10 \times 10 = 10^3$ cells. The next step is to choose 10 cells from the cells. First, the uniform random numbers are generated to calculate the cell number. The cell number indicates the segment number the sample belongs to, with respect to each of the parameters. For example, a cell number (2, 8, 4) indicates that the sample lies in the segment 2 with respect to first parameter, segment 8 with respect to second parameter, and in segment 4 with respect to the third parameter. At each successive step, a random sample is generated, and is accepted only if it does not agree with any previous sample on any of the segment numbers.

As an advantage of LHS method is that it asymptotically eliminates the contribution to the variance due the additive part of the function being integrated. Disadvantage of this method is it does not provide a means of estimating a standard error to be used to form a confidence interval around a point estimate. Because the sample points are so highly correlated in hypercube sampling, proper estimation of error requires careful batching of the sample data. So, to obtain a good estimate of the error we should relax some of the variance reduction properties.

3.4 Low Discrepancy Sequence Sampling

3.4.1 Definition of Discrepancy

Discrepancy measures the departure from the uniformity property (Traub and Papageorgiou, 1996; Morokoff and Caflisch, 1995). The discrepancy of the point set $\{x_i\}_{i=\overline{1,N}} \in [0,1)^s$ is defined as:

$$D_N^{(s)} = \sup_E \left| \frac{A(E;N)}{N} - \lambda(E) \right| \tag{3.2}$$

Where:

$E = [0,t_1) \times ... \times [0,t_s), 0 \le t_j \le 1, j = \overline{1,s}$;

N is the number of points in the sample size;

$\lambda(E)$ is the Lebesgue measure of E, and

$A(E;N)$ is the number of x_i contained in E.

Other words, E is an s-dimensional hyper-rectangle, $\lambda(E)$ is the length, area, volume etc, $\frac{A(E;N)}{N}$ is the percentage of x_i in E, and the discrepancy is the largest difference between $\frac{A(E;N)}{N}$ and $\lambda(E)$.

A low discrepancy sequence is a set of s-dimensional points, filling the sample area efficiently and has a lower discrepancy than straight pseudo-random number set. One simple way to understand the discrepancy concept – and the uniform property – is geometrically. As such, two sets of fifty two-dimensional points are plotted in a unit-square. Figure 3.1 presents a set of fifty points generated by using random and using a low discrepancy sequence for each dimension.

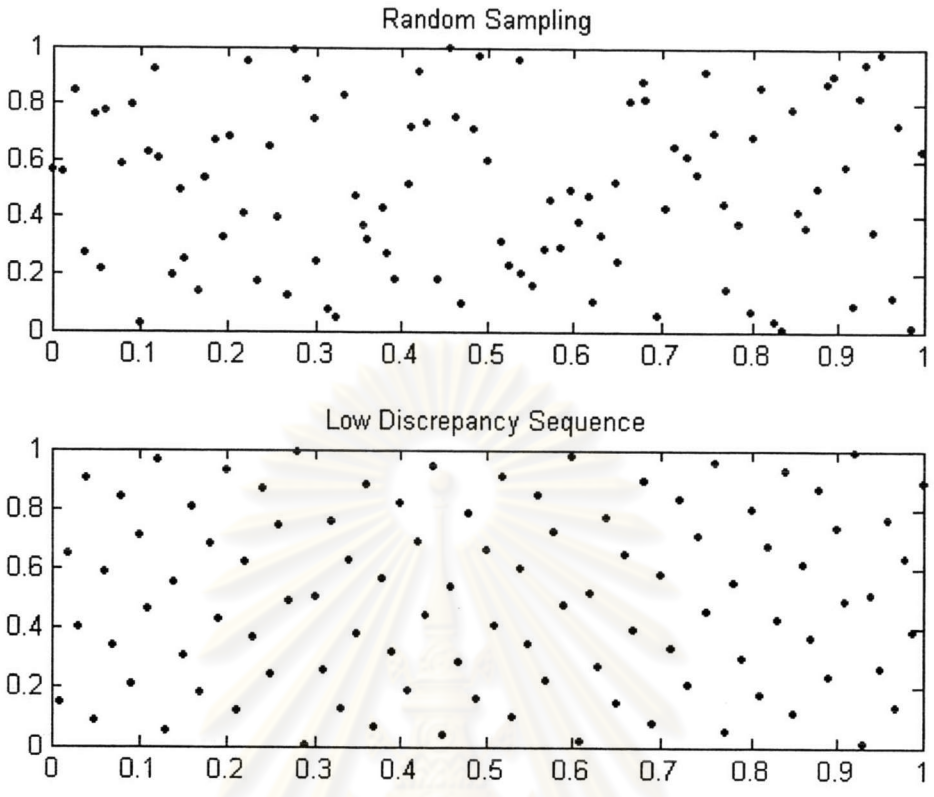

Figure 3.1 Points in the unit square based on random and low discrepancy

Given the importance of the uniformity property in any Monte Carlo application, and that low discrepancy sequences are usually more uniform than random ones, it is reasonable to expect that low discrepancy series will generate better results in simulation experiments than random sampling. The use of low discrepancy sequences is the basis of the Quasi-Monte Carlo sampling (QMC) method.

3.4.2 The Van Der Corput Sequence

The van der Corput sequence is the simplest one dimensional low discrepancy sequence. To obtain nth point x_n of the van der Corput sequence (with a prime base of p), first write the integer *n* in base $P: n = \sum_{i=0}^{l} ai(n) * p^i$, then transpose the digits around the "decimal point" to get the corresponding quasi-random number $x_n = \Phi_p(n) = \sum_{i=0}^{l} \frac{a_i(n)}{p^{i+1}}$. Only a finite number of these $a_i(n)$ will be non-zero. *l* is the

lowest integer that makes $a_i(n) = 0$ for all $i > I$ (I equal to integer part of $\ln(n)/\ln(p), n = \overline{1,N}$.

For example, let p=3 and n=19. We can write 19 in base 3 as $19 = 2\times 3^2 + 0\times 3^1 + 1\times 3^0 = 201$. When reflecting 201 (in base 3) about the "decimal point" we obtain. $x_{19} = \Phi_3(19) = \frac{1}{3} + \frac{0}{9} + \frac{2}{27} = \frac{11}{27}$. This is a number in the interval [0,1]. The sequence with a base of 2 begins:

$$n = 1 : 1 = 1\times 2^0 = 1, x_1 = \Phi_2(1) = \frac{1}{2};$$

$$n = 2 : 2 = 1\times 2^1 + 0\times 2^0 = 2, x_2 = \Phi_2(2) = \frac{0}{2} + \frac{1}{4} = \frac{1}{4};$$

$$n = 3 : 3 = 1\times 2^1 + 1\times 2^0 = 11, x_3 = \Phi_2(3) = \frac{1}{2} + \frac{1}{4} = \frac{3}{4};$$

$$n = 4 : 4 = 1\times 2^2 + 0\times 2^1 + 0\times 2^0 = 100, x_4 = \Phi_2(4) = \frac{0}{2} + \frac{0}{4} + \frac{1}{8} = \frac{1}{8}\ldots$$

After every $N = 2^n - 1$ points, the sequence is "maximally spread out", i.e. the longest interval $(a,b) \subseteq (0,1)$ which does not contain any points from the sequence is a short as possible. The construction process of new low discrepancy sequences involves sub-dividing the unit hypercube into sub-volumes (boxes) of constant volume, which have faces parallel to the hypercube's faces. The idea is to put a number in each of these sub-volumes before going to a finer grid.

Different bases have different cycle length, the quantity of numbers to cover the interval [0, 1) in each cycle. In base two, the pairs (0, 1/2) and the pair (1/4, 3/4) are the two first cycles. For other bases this cycle is larger. For example, in base 3 the sequence has power of 3 denominator with length cycle = 3 (e.g., 0, 1/3, 2/3 is the first cycle). It looks like:

$$0, \frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \frac{1}{27}, \frac{10}{27}, \frac{19}{27}\ldots$$

Two keys to the successful use of QMC methods in high dimensions are the construction of good sequences and the intelligent use or the sequences for path generation. There are several high-dimensional sequences for use in QMC: Halton, Faure, and Sobol sequences. Other important sequences are Hammersley sequences and Niederreiter.

3.4.3 Halton Sequence

Halton sequence is the most basic low discrepancy sequence in multiple dimensions, which can be viewed as the building block of other low discrepancy sequences. The Halton sequence is a general s-dimensional sequence in the unit hypercube [0,1]s. The first dimension of the Halton sequence is the van der Corput sequence base 2 and the second dimension is the van der Corput sequence using base 3. Dimension s of the Halton sequence is the van der Corput sequence using the s-th prime number as the base. As the base of the van der Corput sequence is getting larger as the dimension increases, it takes increasingly longer to fill the unit hypercube (for example, 25th and 26th primes are 97 and 101 correspondingly). The sequence corresponding to the prime p has cycles of length p with numbers monotonically increasing. This characteristic makes the initial terms of two sequences highly correlated, at least to the first cycle of each sequence.

In one dimension for a prime base p, the nth number in the sequence $\{H_n\}, n = \overline{1, N}$, is obtained by the following steps.

For each $n = \overline{1, N}$

1. Write as a number in base np. For example, suppose =3 and n=22, then we can write 22 in base 3 as $22 = 2\times3^2 + 1\times3^1 + 1\times3^0 = 211$.

2. Reverse the digits and put a radix point (i.e. a decimal point base p) in front of the sequence (in the example, we get 0.112 base 3).

3. The result is H_n.

In s-dimension problem, each component a Halton sequence are made with a different prime base p (first n primes are used). Every time the number of digits in n increases by one place, n's digit-reserved fraction becomes a factor of p finer-meshed. So, at each step as n increases points of Halton sequence are better and better filling Cartesian grids.

Table 3.1 Halton sequence for first 3 dimensions

	$Dim=1$ $(Base2)$	$Dim=2$ $(Base3)$	$Dim=3$ $(Base5)$
$n=1$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{5}$
$n=2$	$\frac{1}{4}$	$\frac{2}{3}$	$\frac{2}{5}$
$n=3$	$\frac{3}{4}$	$\frac{1}{9}$	$\frac{3}{5}$
$n=4$	$\frac{1}{8}$	$\frac{4}{9}$	$\frac{4}{5}$
$n=5$	$\frac{5}{8}$	$\frac{7}{9}$	$\frac{1}{25}$
$n=6$	$\frac{3}{8}$	$\frac{2}{9}$	$\frac{6}{25}$
$n=7$	$\frac{7}{8}$	$\frac{5}{9}$	$\frac{11}{25}$
$n=8$	$\frac{1}{16}$	$\frac{8}{9}$	$\frac{16}{25}$

3.4.4 Faure Sequence

The Faure sequence (Faure, 1982) is also a general s-dimensional sequence. Unlike the Halton sequence, all dimensions use the smallest prime p such that $p \geq s$ and $p \geq 2$ as the base. The first dimension of the Faure sequence is the van der Corput sequence in base p. Higher dimensions are permutations of the sequence in the first dimension. For high-dimensional problems the Faure sequence works with van der Corput sequences of long cycle. Long cycles have the problem of higher

computational time (compared with shorter cycle sequences). As occurred with high-dimensional Halton sequence, there is the problem of low speed at which the Faure sequence generates increasing finer grid points to cover the unit hypercube. However, this problem is not as severe as in the case of the Halton sequence. For example, if the dimension of the problem is 55, the last Halton sequence (in dimension 55) uses the 55th prime number that is 257, whereas the Faure sequence uses the first prime number after 55, that is a base 59, which is much smaller than 257. So, the "filling in the gaps" in high-dimensions is faster with Faure sequence when compared with the Halton one. General formula for the length of the n-th cycle in base p is $P^n - 1$. For example, for $P = 4$ the first four cycles are 3, 15, 63, and 255.

To construct the Faure s-dimensional sequence we start by representing any integer n in terms of base p as $n = \sum_{i=0}^{l} a_i^1(n) * p^i$. The first dimension of a Faure point is given by reflecting about the "decimal point", as in the case of van der Corput sequence: $x_n^1 = \Phi_p^1(n) = \sum_{i=0}^{l} \frac{a_i^1(n)}{p^{i+1}}$ We will use recursion to find all remaining dimensions (components) of the sequence point. First assume that all $a_i^{k-1}(n)$ are known. Then $a_i^{k-1}(n)$ can be obtained from the formula: $a_i^k(n) = \sum_{j=i}^{l} \frac{j!}{i!(j-i)!} a_i^{k-1}(n) \bmod b$. (a mod b is the remainder after a number a is divided by divisor b). So, the next level of coefficients is obtained by multiplying by $a_i^{k-1}(n)$ an upper triangular matrix with elements where $\binom{i}{j} = \frac{j!}{i!(j-i)!}$:

$$\begin{bmatrix} \binom{0}{0} & \binom{0}{1} & \binom{0}{2} & \binom{0}{3} & \cdots \\ 0 & \binom{1}{1} & \binom{1}{2} & \binom{1}{3} & \cdots \\ 0 & 0 & \binom{2}{2} & \binom{2}{3} & \cdots \\ 0 & 0 & 0 & \binom{3}{3} & \cdots \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots \\ 0 & 1 & 2 & 3 & \cdots \\ 0 & 0 & 1 & 3 & \cdots \\ 0 & 0 & 0 & 1 & \cdots \end{bmatrix}$$

The successive points in the Faure sequence obtained from

$$x_n^{\ k} = \Phi_p^{\ k}(n) = \sum_{i=0}^{l} \frac{a_i^{\ k}(n)}{p^{i+1}}, 2 \le k \le s, n = \overline{1, N}. \tag{3.3}$$

This recursive procedure permits us to generate S components (dimensions) corresponding to each Faure point n ($n = \overline{1, N}$, N is a number of simulations) in the Faure sequence with P as base ($p \ge s$).

Table 3.2 Faure sequence for first 3 dimensions (Base 3)

	$Dim1$ $(Base3)$	$Dim2$ $(Base3)$	$Dim3$ $(Base3)$
$n=1$	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
$n=2$	$\frac{2}{3}$	$\frac{2}{3}$	$\frac{2}{3}$
$n=3$	$\frac{1}{9}$	$\frac{4}{9}$	$\frac{7}{9}$
$n=4$	$\frac{4}{9}$	$\frac{7}{9}$	$\frac{1}{9}$
$n=5$	$\frac{7}{9}$	$\frac{1}{9}$	$\frac{4}{9}$
$n=6$	$\frac{2}{9}$	$\frac{8}{9}$	$\frac{5}{9}$
$n=7$	$\frac{5}{9}$	$\frac{2}{9}$	$\frac{8}{9}$
$n=8$	$\frac{8}{9}$	$\frac{5}{9}$	$\frac{2}{9}$

3.4.5 Hammersley Sequence

Latin hypercubes are designed for uniformity along a single dimension where subsequent columns are randomly paired for placement on a k-dimensional cube. Hammersley sequence sampling provides a low-discrepancy experimental design for placing n points in a k-dimensional hypercube (Kalagnanam and Diwekar, 1997), providing better uniformity properties over the k dimensional space than Latin hypercubes sampling. A low discrepancy implies a uniform distribution of points in space. The procedure of this sequence is described below.

Each nonnegative integer *k* can be expanded using a prime base *p*:

$$k = a_0 + a_1 p + a_2 p^2 +,...,+a_r p^r \tag{3.4}$$

Where: each a_i is an integer in [0, p-1].

Now, define a function Φ_p of *k* by

$$\Phi_p(k) = \frac{a_1}{p_1} + \frac{a_2}{p_2} + \frac{a_3}{p_3} + ... + \frac{a_r}{p_{r+1}} \tag{3.5}$$

If *p* = 2, the sequence of $\Phi_2(k)$, for *k* = 0,1,2,..., is called Van der Corput sequence as mentioned in sub-section 2.4.2

Let *n* be the dimension of the space to be sampled. Any sequence $p_1, p_2, p_3,..., p_{n-1}$ of prime numbers defines a sequence $\Phi_{p1}, \Phi_{p2}, \Phi_{p3},...\Phi_{pN-1}$ of functions, whose corresponding *k*-th *d*-dimension Hammersley point is

$$\left(\frac{k}{N}, \Phi_{p1}(k), \Phi_{p2}(k),..., \Phi_{pN-1}(k)\right) \quad \text{for } k = 1,2,...N \tag{3.6}$$

Where: p_1, p_2, p_3 = prime number (2,3,5,...)

N = total of number of Hammersley points

Using the Hammersley sampling algorithm with *n*=3 and N =8 produces the data points listed in table 3.3

Table 3.3 Hammersley sequence for first 3 dimension

	$Dim1$	$Dim2$ $(Base2)$	$Dim3$ $(Base3)$
$n=1$	$\frac{1}{8}$	$\frac{1}{2}$	$\frac{1}{3}$
$n=2$	$\frac{2}{8}$	$\frac{1}{4}$	$\frac{2}{3}$
$n=3$	$\frac{3}{8}$	$\frac{3}{4}$	$\frac{1}{9}$
$n=4$	$\frac{4}{8}$	$\frac{1}{8}$	$\frac{4}{9}$
$n=5$	$\frac{5}{8}$	$\frac{5}{8}$	$\frac{7}{9}$
$n=6$	$\frac{6}{8}$	$\frac{3}{8}$	$\frac{2}{9}$
$n=7$	$\frac{7}{8}$	$\frac{7}{8}$	$\frac{5}{9}$
$n=8$	1	$\frac{1}{16}$	$\frac{8}{9}$

3.5 Comparing Different Sampling Techniques

The uniformity characteristics of a sampling technique are clearly important, as discussed above. In this section, the uniformity properties of these sampling techniques: Random sampling, Latin hypercube sampling (LHS), Faure sequence sampling (FSS), and Hammersley sequence sampling (HSS) techniques are compared as shown in Figure 3.2.

Fig 3.2 shows the space filling properties of these techniques. All of the methods sampled 100 points. The points generated by FSS and HSS are spread evenly, almost on a lattice in a unit square. This uniformity on the lattice makes it an attractive method for high dimensional sampling. The Latin hypercube (which is designed to perform well in single dimension) randomly combines with second variable to get a two-dimensional design. As a result, it is not as efficient in two dimensions and this can

reasonably be extrapolated from a 2- to a k-dimensional space. However, all of these sampling techniques have more uniformity properties than random sampling.

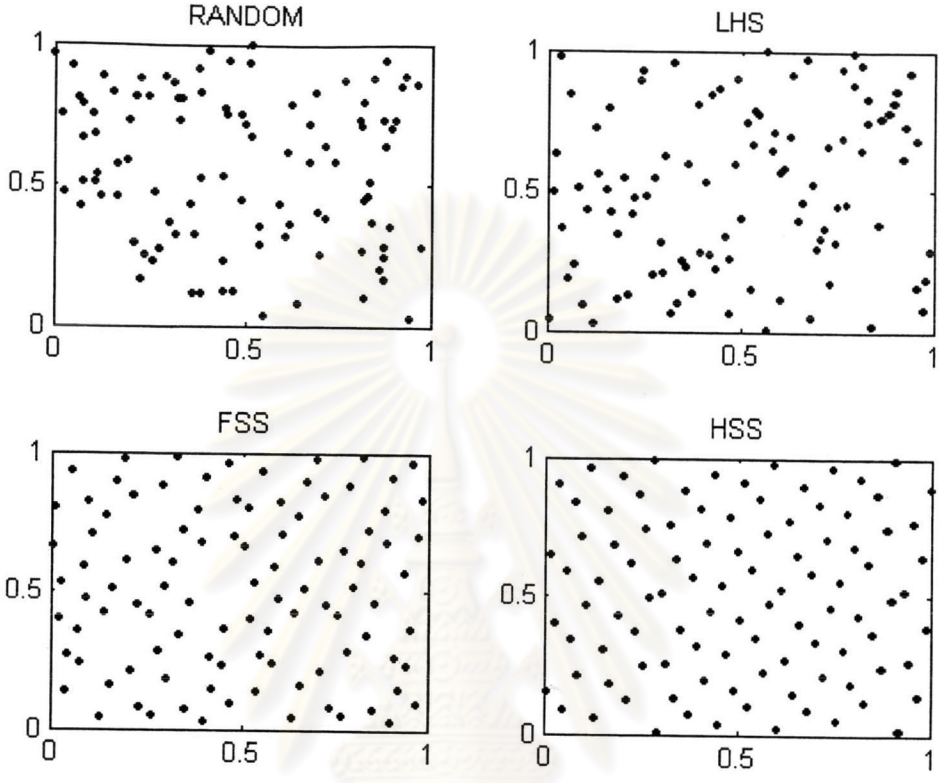

Figure 3.2 Sample Points (100) on a unit square for various sampling

3.6 Summary

Although GA has several advantages, GA not successfully solve on all of problems. Therefore, this chapter highlights the improvement of performance of GA. According to Haupt, 2004; Wei and Zhao, 2005, the diversity of population is critical to the ability of searching the global solution. Hence, an effective way to improve GA is to enhance the uniformity properties of initial population. In this thesis, we improve these properties by sampling technique. Various sampling techniques are also explained in this chapter.