The Study of Special Encoding in Genetic Algorithms and a Sufficient Convergence Condition of GAs*

Bo Yin[1], Zhiqiang Wei[1], and Qingchun Meng[1,2]

[1] Computer Department of Ocean University of China,
Postfach 26 60 71, Qingdao, China
{ybfirst, weizhiqiang, mengqc}@ouc.edu.cn
[2] State Key Lab. of Intelligent Technology & Systems Tsinghua University,
Postfach 10 00 84, Beijing, China
Mengqc@ouc.edu.cn

Abstract. In this paper, the encoding techniques of Genetic Algorithms are studied and a sufficient convergence condition on genetic encoding is presented. Some new categories of codes are defined, such as Uniform code, Bias code, Tri-sector code and Symmetric codes etc. Meanwhile, some new definitions on genetic encoding as well as some operations are presented, so that a sufficient convergence condition of GAs is inducted. Based on this study, a new genetic strategy, GASC(Genetic Algorithm with Symmetric Codes), is developed and applied in robot dynamic control and path planning. The experimental results show that the special genetic encoding techniques enhance the performance of Genetic Algorithms. The convergence speed of GASC is much faster than that of some traditional genetic algorithms. That is very significant for finding more application of GAs, as, in many cases, Genetic Algorithms' applications are limited by their convergence speed.

1 Introduction

The study of Genetic Algorithms(GAs) and their applications have fascinated a great number of researchers during last decades. GAs have enjoyed wide recognition in a large range of domains [1],[2]. A number of experimental studies show that GAs exhibit impressive efficiency in practice and consistently outperform both gradient techniques and various forms of random search on many complex problems. But with the increasing of the problem complexity, classical GAs usually can't solve the problem efficiently. Therefore, it is necessary to improve the performance of GAs and bring some good properties to them, so that they can solve complex problems with higher efficiency[3],[4].

In this paper, the problems of encoding and algorithm convergence in GAs will be studied. Some new encoding techniques are developed. Meanwhile, some definitions in genetic encoding as well as some operations are presented, so that a sufficient convergence condition of GAs could be inducted. Following these studies, some new genetic strategies are proposed, such as Genetic Algorithms with Symmetric Codes, etc. These genetic strategies have been successfully applied to solve the problem of robot dynamic control and path planning [5],[6].

* This research is supported by the National Natural Science Foundation of China(60374031).

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3611, pp. 1005 – 1014, 2005.

2 The Influence of Encoding on the Performance of GAs

It is known that encoding is the first step of GAs. GAs encode the parameters of a problem and produce the initial population. From the initial population, GAs begin to search for the problem's solution. If the initial population is an ill distribution due to the encoding deficiency, i.e. it can not reflect the possible distribution of the optimal solution, it will take a long time to get the optimal solution, or even the solution can't be obtained sometimes. Therefore, encoding can affect the GAs' performance, especially their convergence.

However, the research of the encoding problem has not got enough attention so far. Bethke[7] and Holland[8] analyzed the properties of the binary code string from the concept of Building Block. But they didn't systematically analyze the influence of encoding on the performance of GAs from the aspect of the population members' distribution in the solution space.

In the following section, we will focus our study on some special encoding techniques.

3 Some Special Encoding Techniques in GAs

Encoding determines whether GAs can represent a problem effectively. In order to guarantee GAs' performance, some special genetic codes and encoding techniques have been developed. These typical encoding techniques are introduced as follows.

3.1 Some Traditional Special Encoding Techniques

3.1.1 Bias Encoding

Bias encoding is to encode the problem solution space in different member densities. In a sub-spaces, in which there exists bigger possibility of solution, more members should be encoded.

3.1.2 Uniform Encoding

Uniform encoding will distribute the population members in a manner of uniform possibilities.

3.1.3 Tri-sector Encoding

Tri-sector code is a special kind of Bias Code. According to its definition, the population will be encoded in three sectors, and in each sector the values of members are biased to some values.

3.2 Symmetric Encoding

In this section, the Symmetric Encoding is proposed to improve the performance of GAs. Symmetric codes include Horizontal Symmetric Code, Vertical Symmetric Code, General Horizontal Symmetric Code and General Vertical Symmetric Code.

Definition 1. Horizontal Symmetric Code String: HSCS

(1) If a code string S_i satisfies the following constraint:

$$\sum_{k=1}^{K} Sb_{ik} = 0 \cdot \tag{1}$$

Then, it is called a horizontal symmetric code string, denoted by HSCS.

(2) The encoding of the string S_i is called Horizontal Symmetric Encoding (or Horizontal Symmetric Code), denoted by HSC. And $S_i \in \text{HSC}$.

(3) If a sub-code string of S_i satisfies the following constraint:

$$\sum_{j=J1}^{J} Sb_{ij} = 0, J \in [2, K] \quad and \quad J - J1 + 1 \geq 2 . \tag{2}$$

then the sub-code string $(Sb_{iJ1}, Sb_{iJ1+1}, \cdots, Sb_{iJ})$ is called a sub-horizontal symmetric code string, denoted by $(J - J1)d - \text{SHSCS}$.

(4) If $J - J1 = 2$, this sub-horizontal symmetric code string is called 2-distance sub-horizontal symmetric code string, abbreviated as $2d - \text{SHSCS}$. Analogically, $(J - J1)d - \text{SHSCS}$ can be defined. Its corresponding sub-code string is called sub-horizontal symmetric code string.

Definition 2. Vertical Symmetric Code String: VSCS

Suppose there are n horizontal symmetric code strings with the same length K:

$$\begin{gathered} S_1 = (Sb_{11}, Sb_{12}, \cdots\cdots, Sb_{1K}) \\ S_2 = (Sb_{21}, Sb_{22}, \cdots\cdots, Sb_{2K}) \\ \vdots \\ \vdots \\ S_n = (Sb_{n1}, Sb_{n2}, \cdots\cdots, Sb_{nK}) \end{gathered} \tag{3}$$

(1) If there are two code strings l and i, which have the following relation:

$$\sum_{k=1}^{K} (Sb_{ik} - Sb_{lk}) = 0 \quad j \neq l \, and \quad l, i \in n . \tag{4}$$

then the code strings S_l and S_i are called vertical symmetric code strings, denoted by VSCS.

(2) The encoding of VSCS are called Vertical Symmetric Encoding, denoted by VSC, or abbreviated as Vertical Symmetric Codes, which is expressed by $(S_i, S_l) \in \text{VSC}$.

According to definition 1, we can define $(J - J1)$ distance sub-vertical symmetric code strings $(J - J1)d - \text{SVSCS}$. In the operation of GAs, a sub-symmetric code string can generate new symmetric code strings.

Definition 3. General Horizontal Symmetric Code: GHSC
If a symmetric code string S_i satisfies:

$$\sum_{k=0}^{K} Sb_{ik} = C_2 - C_1 . \tag{5}$$

Then the code string S_i is called a General Horizontal Symmetric Code String, denoted by GHSCS, i.e. $S_i \in \text{GHSC}$. C_1 and C_2 are two constants. According to Definition 1, sub-General Horizontal Symmetric Code ($J - J1)d - \text{GHSCS}$) can be defined.

Definition 4. General Vertical Symmetric Code：GVSC
If two general horizontal symmetric code strings satisfy:

$$\sum_{k=1}^{K} (Sb_{ik} - Sb_{jk}) = C_2 - C_1 . \tag{6}$$

Then the code strings S_i and S_j are called General Vertical Symmetric Code Strings, denoted by VGSCS, $(S_i, S_j) \in$ GVSC. C_1 and C_2 are constants. According to Definition 2, sub-General Vertical Symmetric Code (SGVSC) can be defined.

Based on these special codes, some new genetic strategies are developed, such as Genetic Strategy with Gate Change Function and Genetic Algorithm with Symmetric Codes [5],[6].

4 Some Definitions for the Study of Genetic Algorithm Convergence

The former study[5]shows that the influence of genetic encoding on GAs' performance is significant. Based on the encoding techniques, some new and powerful genetic strategies may be developed, so as to deal with more complex engineering problems. In practice, it is obvious that GAs must be modified when they are applied to solve some complex problems. Therefore, it is necessary to study more genetic encoding and GAs' convergence problems.

In this section, some new definitions and member operations are proposed to describe the encoding problem and encoding mechanism.

Supposing that a population *P-space* has m members and each member has n binary bits complying with the requirement of GAs coding, i.e.

$$\begin{aligned} P-space &= (S_1, S_2, \cdots\cdots, S_m)^T = (C_1, C_2, \cdots\cdots, C_n) \\ &= \left[a_{ij}\right]_{i=1,\cdots\cdots,m,\, j=1,\cdots\cdots,n} . \end{aligned} \tag{7}$$

Here, a_{ij} is a binary bit '0' or '1'. $C_1, C_2, \cdots\cdots, C_n$ represent n columns in *P-space*. $S_1, S_2, \cdots\cdots, S_m$ represent m rows.

Definition 1. (Link)
A column C_i of a population *P-space* is called as a Link, so there are *n* Links in the population *P-space*.

Definition 2. (Dead-block)
In a *P-space*, if there exists one or several sub-matrix blocks in which all its elements can not be changed when a cross-over operation is proceeded, it is called as a Dead-block. In another word, in Dead-block, all the elements are of same value.

Definition 3. (Living-block)
In a *P-space*, if there exists one or several sub-matrix blocks in which all its elements can be changed when a cross-over operation is proceeded, it is called as a Living-block. In the same way, Living-population (Definition 4) and Dead-population (Definition 5) can be defined.

Definition 4. (Fix-link)
If all the elements in a Link C_i of a population are of same values '0' or '1', it is called as a Fix-link.

Definition 5. (Unfix-link)
If in the *m* elements of a Link C_i, there is at least one element which has different value from those of the others, this link is called as an Unfix-link.

Definition 6. (Bit-exchange position)
When a cross-over operation is conducted between two members S_l and S_k at the i^{th} bit position, this bit is called as Bit-exchange position P_{ci}.

Definition 7. (Bit-transformation)
A cross-over operation between two members at P_{ci} is called as genetic transformation noted as T_i .

Definition 8. (Bit-transposition)
An operation between two consecutive Bit-transpositions T_i and T_{i+1} is known as a Bit-transposition operation, noted as TP_i .

Definition 9. (Bit-and operation)
If an operation, conducted between two bits b_i of member S_k and b_j of member S_l, satisfies the following law:

$$\begin{array}{ll} \text{if } b_i = b_j & \text{then } b_i \otimes b_j = b_i . \\ \text{if } b_i \neq b_j & \text{then } b_i \otimes b_j = * . \\ \text{if } b_i \; or \; b_j = * & \text{then } b_i \otimes b_j = * . \\ i, j = 1, 2, \cdots\cdots, n, & b_i \in S_k ,\; b_j \in S_l . \end{array} \tag{8}$$

Here, the symbol * represents a Don't Care Bit [12]. This operation is called as Bit-and operation. Based on this operation, Member-and (Definition 12) and Population-and operation (Definition 13) can be defined in the same way.

As it is known, a population, *P-space* = $(C_1, C_2, \ldots\ldots C_n)$, in which a member has *n* elements, will have 2^n possible members. Therefore, in general a Genetic Algorithm should be able to generate any one of the 2^n possible members. According to the above definitions, we can get the sufficient convergence condition of GAs.

5 The Sufficient Convergence Condition on Encoding of GAs

Based on the above definitions on population and genetic operations, some conclusions will be discussed in the following lemmas and theorems.

Lemma 1. In a population with *m* members and *n* Links, if all of its *n* Links are Unfix-links, then an Unfix-member will be generated after a population-and operation.

Lemma 2. In a Living-population with *m* members and each member with *n* elements, any of the *n* elements in a member may be changed by limited times of transposition operations.

Lemma 2 means that, in a Living population, any of the 2^n possible members in solution space may be obtained by limited times of transposition operations.

Theorem 1. In a population with *m* members and each member with *n* bits, only if the result of its Population-and operation is a Don't Care member, it is possible to generate any of the possible 2^n members in the population by limited times of Transposition or Transformation operations. In another word, each Link in the population is required to be Unfixed Link.

Theorem 1 means that only if every link in a population is Unfixed Link, it is possible for Genetic Algorithm to generate any of 2^n possible members of the population by limited numbers of genetic operation. Otherwise GAs will run the risk of being unable to generate some members.

Theorem 2. Supposed that an optimization problem has a solution space that is covered by a Living-population of GAs, and the population has *m* members and each member has *n* bits. If the condition in Theorem 1 is satisfied, then in GAs the problem's solution or solutions may be got by limited genetic Transposition or Transformation operations.

Theorem 2 presents a sufficient condition on the convergence of GAs. This sufficient condition will help us to take a better encoding requirement, so that a GA can get its convergence, i.e. this condition can guarantee a GA to avoid the risk of missing some solutions during the solution procedure.

6 The Application of GASC in Robot Optimal Control and Path Planning

The problem of robot dynamic control and path planning has been studied for decades. This problem has very high algorithm complexity, as there are multiple variables and solutions. In order to solve this problem using GAs, some measures must be taken so as to improve the efficiency of GAs. In this paper, the special genetic strategy-GASC is employed to solve this problem.

From literature[5], it is known that the robot has a linear dynamic model (the relation between controls C_1, C_2 and linear velocity and angular velocity; the final velocities: $v(T)=w(T)=0$, there T is the time in which robot finishes one trajectory). According to the definition of Symmetric Codes, we can know that the population in GASC can automatically satisfy the robot linear speed and angular speed constraints at the final point of a trajectory. This property will make the problem easier to solve. That is why we develop GASC. The following two theorems can express the advantages of the application of Symmetric Codes to the robot problem.

Theorem 1. If and only if two controls C_1 and C_2 (the members of a population) are Horizontal Symmetric Code: HSC, then linear velocity at the final point of trajectory $v(T)$ is zero. Or in the following expression:

$$v(T)=0 \Leftrightarrow (C_1,C_2) \in \text{HSC}. \tag{9}$$

Theorem 2. If and only if two controls C_1 and C_2 are Vertical Symmetric Code: VSC, then angular velocity at the final point of trajectory $w(T)$ is zero. Or in the following expression:

$$w(T)=0 \Leftrightarrow (C_1,C_2) \in \text{VSC}. \tag{10}$$

The above theorems are derived under the condition of linear robot dynamic model and zero final velocity. They can be extended to general final velocity conditions.

From the above two theorems, it is clear that if the population (C_1 and C_2) of GA is encoded in Symmetric Codes, the robot's linear and angular final velocities' constraints will be automatically satisfied.

7 Experimental Results

From the previous section, we can see that the robot optimal dynamic control and path planning problem can be mathematically formulated into an optimization problem. GAs are applied to solve this problem. Since GAs have learning abilities, they can search and improve the solution generation by generation.

In the application of GASC, the robot energy consumed is considered as a criterion. The optimization function is formatted according to the principle of external penalty method for dealing with the constraints. The main algorithm parameters are: Popsize= 50; Pc=1; Pm=0.02. Two kinds of Genetic Algorithms are used in the experiment:(1) Simple Genetic Algorithms; (2) GASC.

7.1 Experimental Results of Simple Genetic Algorithms

In this phase several simple genetic techniques are used. We have practiced two kinds of codes and three "mutation—crossover" methods:

1. Code 1: uniform code. The robot torques C_1 and C_2 are uniformly coded from -0.10 to 0.10.

   Code 2: 3--sector code. In this code, C_1 and C_2 are divided into 3 sectors (T1, T2, T3). T1=15, T2=20, T3=15.

   During T1 stage, roughly 70% of C_1, C_2 have values from .04 to .10, 20% from 0 to .04 and 10% from -0.10 to 0.

   During T2 stage, approximately 60% of C_1, C_2 have values from -0.02 to 0.02, 20% from 0.02 to 0.10 and 20% from -0.10 to -0.02.

During T3 stage, roughly 70% of C_1, C_2 have values from -0.10 to -0.04, 20% from -0.04 to 0 and 10% from 0 to 0.10.

2. Random 10-point cross-over technique. In every member of a generation, 10 points are chosen randomly, and cross-over occurs between two strings chosen randomly.
3. 1-point cross-over technique. In every generation, we choose two members randomly and a cross-over is performed at the point at which robot meet an obstacle.
4. 10-point--1-point cross-over technique. During the operation of the first half number of generations given, 10-point crossover technique is introduced in, and the second half, 1-point method is used.

The results obtained by simple GAs are listed in Table 1. Meanwhile, Fig.1 is the trajectory obtained by the simple GAs. From Fig.1, we can see that the trajectory is not satisfactory. Even if the simulation is allowed to proceed to 100000 generations, there is not any string which arrived at the final point X(20,20). This is comprehensible, since simple GAs often break the hopeful strings and destroy the previous search work unreasonably.

Table 1. The list of the results obtained by simple Gas

Tests	Codes	GN	K_{j1}	K_{j2}	Q_o	BP(x,y)
P1T1	1	2000	0.4	4.0	0.8	8.7/9.8
P1T2	1	2000	0.6	6.0	0.8	10/0.3
P1T3	2	2000	0.6	6.0	0.8	10.8/0.3
P1T4	2	10000	1.2	0.6	1.0	16.1/9.5
P1T5	1	10000	1.2	0.6	1.0	15.3/10.9

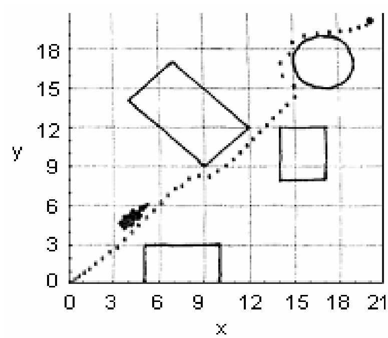

Fig. 1. The trajectory obtained by Simple Genetic Algorithms

Where, GN is the generation number of GAs. K_{j1} and K_{j2} are the weights assigned for the code strings of the control parameters. Q_o is the initial angle of the robot. BP(x,y) represents the best position the robot arrives at when the algorithm is over. P1T1--P1T5 represent the cases with different parameters in the simulation experiment.

7.2 Experimental Results of GASC

In this experiment, GASC is employed to solve the optimization function. Meanwhile, some special genetic tactics are also introduced, such as “Hopeful Member Immigrating”, “Good member Protection”, etc. These techniques also have an important influence on the performance of GASC[5]. The trajectory obtained by GASC is shown in Fig.2. From the result, we can see that GASC outperforms simple GAs, as the quality of the trajectory is much better than the one in Fig.1.

8 Conclusion

Our study shows that genetic encoding techniques have significant influence on GAs' performance in solving problems with high algorithm complexity. In such problems, some special codes must be taken, otherwise the algorithms may not be able to get convergence or the solution obtained is poor. The experiments show that the proposed Genetic Algorithms with the symmetric codes can find solutions with better quality in shorter time than some classical GAs.

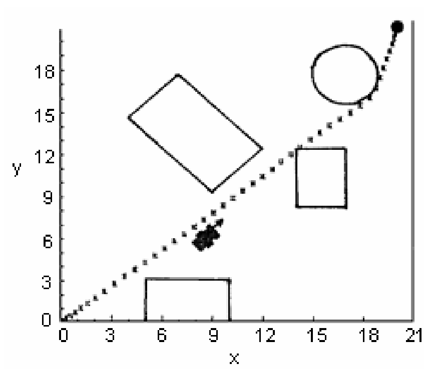

Fig. 2. The trajectory obtained by GASC

In this paper, the genetic encoding problem and a sufficient convergence condition of GAs are studied. Some new categories of genetic codes are defined, and they are applied into robotic problem successfully. Meanwhile, some new definitions on genetic encoding as well as some operations are presented, so that a sufficient convergence condition of GAs is inducted. This condition can help encoding the population so as to ensure the convergence of a GAs. It is proved that this condition is necessary for GAs.

References

1. Gautam Garai, Chaudhuri B.B.: A novel genetic algorithm for automatic clustering. Pattern Recognition Letters. 25(2004) 173-187
2. Barrie M.Baker, M.A. Ayechew: A genetic algorithm for the vehicle routing problem. Computers & Operations Research. 30(2003) 787-800
3. Chi-bin Cheng, C.J.Cheng, E.s.Lee: Neuro-Fuzzy and Genetic Algorithm in Multiple Response Optimization. Computers and Mathematics with Applications. 44(2002) 1503-1514
4. Yas Abbas Alsultanny, Musbah M. Aqel: Pattern recognition using multiplayer neural-gentic algorithm. Neurocomputing. 51(2003) 237-247
5. Qingchun Meng, Hongbo Ji, Hao Dong: Application of a new Genetic strategy to robot control. In Proc. of Intl. Conf.IEEE, ICIPS'97. Beijing (1997) 568-570
6. Qingchun Meng, Changjiu Zhou, et al: Genetic Algorithm with Symmetric Code and Its Application to Dynamic System. ACTA ELECTRONICA SINICA. 27(1999) 59-63
7. Bethke A.D.: Genetic algorithm as function optimizers. Ph. D. Thesis, Dept. Computer and Communication Sciences, University of Michigan(1981)
8. Holland J. H: Adaptation in natural and artificial systems. University of Michigan press. Ann Ardor, Michigan(1975)