Proceedings of the 2007 IEEE Symposium on
Foundations of Computational Intelligence (FOCI 2007)

# Eliminating Positional Dependency in Binary Representation via Redundancy

C. Y. Cheong, S. C. Chiam and C. K. Goh

***Abstract* — Recent studies show that evolutionary algorithms are effective optimization tools for their success in solving real-world problem with complex and competing specifications. Although their performances are greatly influenced by the type of representation adopted, this choice often arises from intuition and guesswork due to the absence of proper guidelines and framework. This paper considers binary representation and presents a study on the key factors that affect its algorithmic performance. Subsequently, an encoding scheme is proposed to resolve the problem of positional dependency in binary coding, which is the commonly used genotype-phenotype mapping for this representation. This is achieved by introducing redundancy into the genotype-phenotype mapping, which will better preserve the similarities between the genotype and phenotype search space by resolving the exponential orderings between the alleles. Theoretical analysis and empirical study were conducted to investigate the characteristics of the proposed representation.**

## I. Introduction

Evolutionary algorithm (EA) is a class of stochastic search technique that has been gaining significant attention from the research community in the recent years for its success in solving complex real-world problems with various competing specifications. The EA paradigm is largely inspired by the biological process of evolution, where potential solutions are encoded as chromosomes to epitome the mechanics of DNA blueprint of living organisms. This allows the propagation of information through the operation of recombination and the inheritance of desirable properties to offspring solutions. As such, the representation scheme for the individual chromosome has always been a fundamental design issue in EA and the choice of an appropriate representation is vital for satisfactory algorithmic performance.

Specifically, representation defines "the window" at which the algorithm views the optimization problem. Pseudo-chromosomal (binary) strings, real-number vectors and complex data structures are some classical representation schemes available and the choice of usage is highly dependent on the type of optimization problems involved. Despite the wide range of representation available, binary representation is still widely adopted in many recently proposed EA and EA applications, mainly due to its ease in implementation and compatibility with classical variation operators.

Binary representation stems from the early work of genetic algorithm where potential solutions for the optimization problems are represented by pseudo-chromosomal strings. The genotype exists strictly in the form of binary string, while the phenotype can be in the form of bits, integers, real number, etc., depending on the problem. Although using genotype and phenotype with identical nature seems to be the most direct approach, denoting continuous phenotype with binary genotype, which represents a form of indirect encoding, is still a viable form of representation in EA. This is best explained by Veldhuizen and Lamont [1] in the context of multiobjective optimization:

> *"When the real (continuous) world is modeled (e.g., via objective functions) on a computer (a discrete machine), there is a fidelity loss between the real world and implemented model. However, at a standardized resolution and representation, MOEA (multi-objective evolutionary algorithms) results can be compared against both each other and $PF_{true}$ (global Pareto Front). Thus, whether or not a given MOP's (multi-objective optimization problem) true Pareto front is actually continuous or discrete is then not a major concern, as the computed front is always composed of discrete points at a specified computational resolution."*

Due to the indirect representation, genotype-phenotype mapping are necessary and the most commonly used is the binary code. However, binary code suffers from the problem of positional dependency, where the amplitude of phenotype variation is dependent on the position of altered genotype bit. This is contrary to the classical variation operators, which regards each genotype bit as equal. While there have been attempts to induce positional bias into the variation operation to synchronize itself with the representation [2], this fundamental problem has not been resolved from the perspective of representation.

Hence, to address the issue of positional dependency, this paper proposes a new coding scheme that alleviates this problem by introducing redundancy into the genotype-phenotype mapping. Its characteristics will be investigated via theoretical and empirical studies. The term, binary representation, will conveniently refer to the representation of continuous phenotype with binary genotypes in this paper.

## II. Genotype-Phenotype Mapping

By virtue of the indirect encoding in binary representation, fitness evaluation is decomposed into a genotype-phenotype mapping, $f_g$ and a phenotype-fitness mapping, $f_p$. The former will map the genotype search space, $\Phi_g$ into the phenotype search space range, $\Phi_p$, from which the latter will then map it to the fitness space, $\Re$ [3]. The mathematical formulation is

C. Y. Cheong, S. C. Chiam and C. K. Goh are with the Department of Electrical and Computer Engineering, National University of Singapore, 4, Engineering Drive 3, Singapore 117576. Email: g0500055@nus.edu.sg.

1-4244-0703-6/07/$20.00 ©2007 IEEE

given in (1) and (2). Specifically, $f_g$ represents the decoding of the binary string, $\vec{x}_g$ into its corresponding set of decision variables, $\vec{x}_g$, while $f_p$ calculates the corresponding fitness, depending on the optimization problem at hand.

$$f_g : \Phi_g \to \Phi_p \tag{1}$$

$$f_p : \Phi_p \to \Re \tag{2}$$

The overall difficulty of an optimization problem, $F$ thus depends on the composite function, $f_p \circ f_g : \Phi_g \to \Re$. Since $f_p$ is normally fixed as it depends on the problem at hand, the choice of $f_g$ will have a significant influence on $F$, and is thus one of the fundamental design issues in EA. This section will present a formal definition of $f_g$ and discuss some issues related to it.

### *A. Effect of Redundancy on the Definition of Genotype-Phenotype Mapping*

Biologically, the genotype-phenotype mapping emulates the protein biosynthesis process that dictates how proteins are built in cells by representing variables with pseudo-chromosomal binary stings. Protein biosynthesis is a two stage process where the particular DNA sequence is first copied by an RNA polymerase to produce a complementary RNA; the RNA is then decoded to produce a functional protein according to the rules specified by the genetic code. The two distinct stages are termed as transcription and translation respectively. Essentially, the former extract the relevant genes from the entire DNA strand by filtering away all the redundant genes, while the latter then decodes these genes to obtain the necessary information.

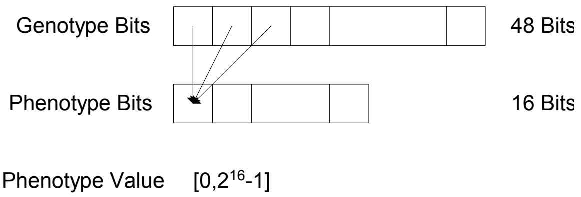


Fig. 1. Illustration of trivial voting mapping

While such definition might be superfluous for the binary coding scheme, the advent of redundancy has complicated $f_g$, where a single phenotype might be represented by several genotypes. Hence, before applying binary code to determine the phenotype value represented by the genotype, some preprocessing is required. This can be explained clearer by using trivial voting mapping [4] as an example. This mapping exercises redundancy by representing each phenotype bit with several genotype bits (three in this case) as shown in Fig. 1. To obtain the phenotype value, the genotype must first be transformed to a phenotypic binary string; after which binary code can then be applied to obtain the phenotype value.

As such, redundancy has extended $f_g$ into a two-stage process which adheres closer to the biological process of transcription and translation. Hence, (1) can be rewritten as,

$$f_{g1} : \Phi_g \to \Phi_{p'} \tag{3}$$

$$f_{g2} : \Phi_{p'} \to \Phi_p \tag{4}$$

where (3) and (4) denote the transcription and translation function respectively and $\Phi_p$ denotes the phenotype search space described in binary bits. The resultant $f_g$ is the composite function $f_{g2} \circ f_{g1} : \Phi_g \to \Phi_p$.

The redefinition of phenotype-genotype mapping is largely motivated by the development in redundancy. Besides the straightforward mapping as in trivial voting mapping, other existing implementations of redundancy on binary representation include defining rules to map genotype into their corresponding phenotype bits like in static random mapping, cellular automaton mapping and random Boolean network mapping [4]. Redundancy allows neutral mutations, where it is possible that a change in a genotype might not change its corresponding phenotype, allowing new properties that can be advantageous in the future to be designed, without interfering with the current phenotype [5].

On the other hand, the development in translation has been less significant. The only alternative available for binary code is gray code which was proposed to alleviate the Hamming cliff problem in binary code.

### *B. Issues of Genotype-Phenotype Mapping*

Binary representation essentially discretized the continuous phenotype search space into distinct points. Ideally, all these points should be evenly distributed to prevent any bias towards any particular region. Redundancy will decrease the total unique number of phenotype points, as some regions in the phenotype space might be overrepresented. Another aspect of distribution is resolution which refers to the distance between each phenotype points. A coarse resolution will result in a model with poor predictive ability as the actual phenotype search space is poorly represented [1]. Conversely, if the resolution is too fine, the model is generally intractable.

Besides distribution, locality, which quantifies the similarities between the genotype and phenotype search space, greatly influences the algorithmic performance of the EA depending on how much the original structure of the phenotype search space is maintained after the transformation. For this purpose, a correlation based analysis [6] and a probabilistic causality model [7] are developed to quantify the impact of genotype-phenotype mapping. Recently, Rothlauf [3] uses locality and distance distortion to measure how well a representation preserves the phenotype search space. The former describes how well neighboring phenotypes correspond to neighboring genotypes and the latter extends this concept to include large changes.

This measure was subsequently extended by Chiam *et al* [8], where the concept of proximity preservation, *PP* was introduced. Essentially, it measures how well the proximity between neighboring phenotypes and genotypes is maintained. Besides considering the mean genotype distance for all neighboring phenotypes as in Rothlauf's locality measure [3], the converse which corresponds to the mean phenotype distance of all neighboring genotypes was accounted as well.

Before these two metrics are introduced, it is necessary to define certain measures to quantify the distance in $\Phi_g$ and

$\Phi_p$ respectively. In this paper, distance in $\Phi_g$ will be measured by the Hamming distance, which is defined as

$$d^g_{\overrightarrow{x_g},\overrightarrow{y_g}} = \sum\nolimits_{i=0}^{\ell-1} | x_{g,i} - y_{g,i} | \tag{5}$$

where $\overrightarrow{x_g}$ and $\overrightarrow{y_g}$ represent two binary strings, $x_{g,i}$ denotes the *i*-th bit of $\overrightarrow{x_g}$ and $\ell$ symbolize the length of the binary strings. As for $\Phi_p$, the distance between two phenotype $\overrightarrow{x_p}$ and $\overrightarrow{y_p}$ is defined as

$$d^p_{\overrightarrow{x_p},\overrightarrow{y_p}} = \sqrt{\sum\nolimits_{i=0}^{n} \left( x_{p,i} - y_{p,i} \right)^2} \tag{6}$$

where *n* represents the dimension of the search space which normally corresponds to the number of objectives in the optimization problem, $x_{p,i}$ and $y_{p,i}$ represents of $i^{\text{th}}$ component of $\overrightarrow{x_p}$ and $\overrightarrow{y_p}$ and the phenotype range in each dimension are normalized to [0,1].

Proximity preservation in the genotype space, $PP_g$ is defined as such,

$$PP_g = \frac{1}{2^\ell} \times \sum_{d^p_{\overrightarrow{x_i},\overrightarrow{x_j}} = d^p_{min}} d^g_{\overrightarrow{x_i},\overrightarrow{x_j}} \tag{7}$$

where $d^p_{x_i,x_j}$ and $d^g_{\overrightarrow{x_i},\overrightarrow{x_j}}$ refer to the phenotype and genotype distance between $\overrightarrow{x_i}$ and $\overrightarrow{x_j}$, and $d^p_{min}$ is the minimum distance between the phenotype points. We see that $PP_g$ only considers neighboring phenotypes ($d^p_{x_i,x_j} = d^p_{min}$) and in the event of tie, the average genotypic distance will be considered. The sum of the genotype distance is divided by $2^\ell$, which denotes the number of phenotype points considered.

Though this means that neighboring phenotypes are situated closely in the genotype space, this is only applicable if the correct bit is altered. Hence, the average phenotype distance moved when altering each of the bits must also be considered and this corresponds to the proximity preservation in the phenotypic space, $PP_p$ which is defined as,

$$PP_p = \frac{1}{2^{\ell-1} \cdot \ell} \times \sum_{d^g_{x_i,x_j}=1} \frac{d^p_{x_i,x_j}}{d^p_{min}} \tag{8}$$

Since each binary string has $\ell$ neighbors, the sum of the phenotype distance is divided by $2^{\ell-1} \cdot \ell$, which is the number of genotype points considered. A low value of $PP_p$ represents the case when locally, the phenotype structure is preserved in the genotype structure, and hence, *F* retains the same level of difficulty in close proximity. These metrics will be used later to measure how well the proximity between neighboring phenotypes and genotypes is maintained for the various translation codes.

## III. Translation Function and Positional Dependency

Although there have been quite a substantial amount of work in transcription, where different type of redundancy schemes had been proposed, there has been limited development in translation after gray coding. In this section, the proposed translation code will be introduced. It introduces redundancy into translation directly, and by doing so, the issue of positional dependency was alleviated as well.

### *A. Classical Translation Functions*

Binary code, BIN is the most extensively used translation function in literature. For a binary string of length $\ell$, BIN is defined as

$$x_p = \frac{1}{2^\ell - 1} \sum_{i=0}^{\ell-1} 2^i \cdot x_{g,i} \tag{9}$$

where $x_{g,i}$ denote the *i*th bit of the genotype, $x_g$.

BIN is a simple one-to-one mapping where the discretized points will be evenly distributed in the phenotype search space without any redundancy. Its main drawback however is the Hamming cliff, where for some neighboring points in the phenotype search space, their distance is amplified significantly in the genotype space. To alleviate this problem, gray code (GRA) was subsequently proposed as an alternative. GRA ensures that genotypes of neighboring phenotype differ by at most a single genotype bit.

Besides these two widely used translation schemes, there is also unary code (UNA) which is simply the unitation, number of ones, of a fixed length binary string. Its mathematical formulation is defined in (10).

$$x_p = \frac{1}{\ell} \cdot \sum_{i=0}^{\ell-1} x_{g,j} \tag{10}$$

UNA is however not considered in practical EA applications, as a binary string of length $\ell$ can only represent $\ell + 1$ distinct phenotype points using UNA, due to the high degree of redundancy.

### *B. Reducing Positional Dependency via Redundancy*

Although, there have been extensive theoretical and empirical comparison studies between these codes [9]-[10], no common consensus has been reached and furthermore, the problem of positional dependency still persists. For BIN, a change in the significant bits will result in larger variation in the phenotype space as opposed to a change in the less significant bits. In fact, the ordering is exponential! Even though GRA ensures that neighboring phenotypes differ by at least one genotype bit, it fails to minimize the phenotype distance between each genotype neighbor. This is a case of low $PP_g$ with high $PP_p$.

Fig. 2 illustrates this concept clearer by showing the genotype and phenotype space of GRA,. For each neighboring phenotypes, their genotype differ by at most a single bit. The high locality between the phenotypes is desirable. However, in the genotype space, $PP_p$ is actually quite high. For example, the nearest genotype neighbors for chromosome 000 are 001, 010, and 100. But, their phenotype distance differs by 1, 3 and

7 respectively. As the evolutionary operators are actually working on the genotype space, locality in the genotype space is more crucial then in the phenotype space.

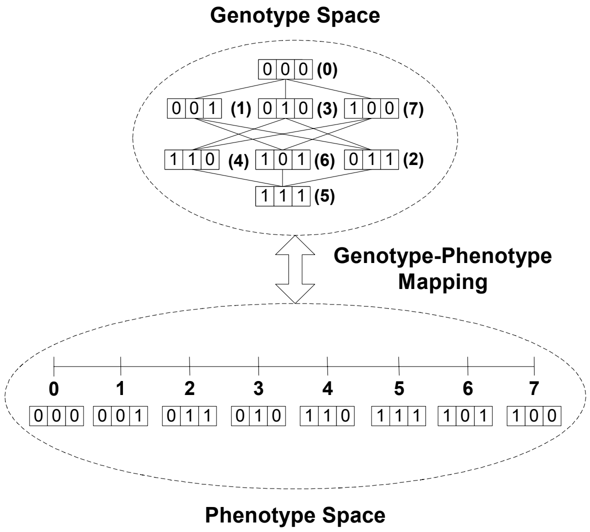


Fig. 2. Genotype and Phenotype space of GRA

On the other hand, UNA, which possess a higher degree of redundancy, has a lower $PP_p$ value. Table I shows all the possible genotype and phenotype value for a binary string of length 3 using UNA. Redundancy exists in the sense that some phenotype value is represented by multiple genotypes. For example, 001, 010 and 100 all represent the phenotype value of 0.33. The number of '0' and '1' directly determine the phenotype value where a higher number of '0' indicates a lower phenotype value and vice versa. The implication of redundancy results in the depreciation of the number of unique points that can be represented by a given string.

TABLE I
GENOTYPE AND THEIR DECODED PHENOTYPE FOR UNA

| **Genotype** | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| **Phenotype** | 0.00 | 0.33 | 0.33 | 0.67 | 033 | 0.67 | 0.67 | 1.00 |

However, by assigning equal weights to each bit, UNA eliminates the orderings between the bits and hence, their inter-dependencies. The transition of phenotype 0.00 to 0.33 can be made by inverting any of the genotype bits to '1'.This positional independency is synchronized to the variation operation which treats each bit as equal.

Closer examination of (9) and (10) reveals that BIN and UNA can actually be generalized as

$$x_p = \sum_{i=0}^{l-1} w_i \cdot x_{g,i} \Big/ \sum_{i=0}^{l-1} w_i \tag{11}$$

where $W = \{w_0, w_1, ...., w_{\ell-1}\}$ is the assigned weight vector that defines the ordering between the bits. If the weight vector consists of equal numbers or defines an exponential ordering of base 2, it will correspond to UNA and BIN respectively

These two coding represent the two extremum where there is a tradeoff between positional dependency and distribution. Instead of predefined weights, the weight vector could be a string of randomized number instead. Such an approach actually represents a compromise between these UNA and BIN and is defined as,

$$x_p = \sum_{i=0}^{l-1} r_i \cdot x_{g,i} \tag{12}$$

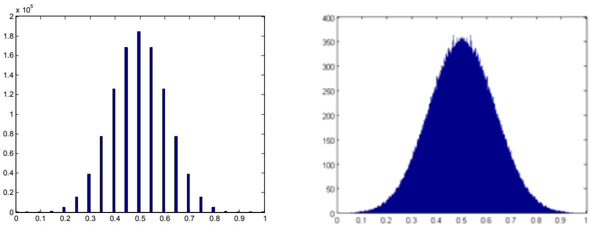
where $R = \{r_0, r_1, ...., r_{\ell-1}\}$ is a normalized weight vector randomly generated at the beginning of every run and remained the same throughout that run. This translation function will be known as random code, RND. Obviously, RND depends heavily on the initial random weight vector generated.

## IV. THEORETICAL ANALYSIS

This section cover the theoretical analysis of RND, examining mainly the two issues highlighted earlier, namely distribution and locality.

### A. Distribution

The distribution of the phenotype points in the search space using UNA for a 20-bit long binary string is shown in Fig. 3(a). An obvious disadvantage of UNA is the coarse resolution due to redundancy, where a binary string of length $\ell$ can only represent $\ell + 1$ different phenotype points using UNA, instead of $2^{\ell} - 1$ phenotype points for BIN. Furthermore, regions in the center of the phenotype search space are overrepresented.



(a) (b)

Fig. 3. Distribution of phenotype points using (a) UNA and (b) RND

The direct consequence of using RND is that better resolution is obtained as seen in Fig. 3(b), which shows its distribution of phenotype points. However, the problem of overrepresentation in the central region still persists, though it is not as severe as UNA. Of course, bias towards specific region in the phenotype search space will not be recommended for solving optimization problems in general. However, if knowledge of the optimal regions could be deduced, for example through some domain knowledge or background experience, overrepresentation in the vicinity of these regions will improve algorithmic performance [5]. But very often, the situation will be such that the optimal region cannot be specified or when it is distributed uniformly in the search space. Some recommendations to resolve this overrepresentation issue for RND will be discussed later in section VII.

### B. Locality

While redundancy worsens the distribution of the translation codes, it alleviates the problem of positional dependency. For this exercise, a binary string of length 3 is considered and the various genotype and their decoded phenotype values under the various codes are tabulated in table II. For RND, a weight vector of [0.2 0.3 0.5] is considered.

TABLE II
GENOTYPE AND THEIR DECODED PHENOTYPE

| | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| **Binary** | 0.00 | 0.14 | 0.29 | 0.43 | 0.57 | 0.71 | 0.86 | 1.00 |
| **Gray** | 0.00 | 0.14 | 0.43 | 0.29 | 1.00 | 0.86 | 0.57 | 0.71 |
| **Unary** | 0.00 | 0.33 | 0.33 | 0.67 | 0.33 | 0.67 | 0.67 | 1.00 |
| **Random** | 0.00 | 0.50 | 0.30 | 0.80 | 0.20 | 0.70 | 0.50 | 1.00 |

The *PP* values for the various codes were calculated and the results are summarized in table III. For $PP_g$ , GRA and UNA has the lowest possible value of 1 which means that for all neighboring phenotypes, their hamming distance differ by only 1 bit. As for $PP_p$, BIN and GRA actually obtained the same value. Conversely UNA and RND attained a smaller value for these measures. In order to further appreciate the significance of this result, different $\ell$ are considered and the results are shown in table IV. As RND depends heavily on the random weight vector generated, 100,000 samples were considered and the median *PP* values were recorded.

TABLE III
*PP* OF THE VARIOUS CODES

| | BIN | GRA | UNA | RND |
|---|---|---|---|---|
| $PP_g$ | 1.63 | 1.00 | 1.00 | 1.00 |
| $PP_p$ | 2.33 | 2.33 | 1.00 | 1.67 |

TABLE IV
*PP* AND *RP* OF THE VARIOUS CODES AGAINST LENGTH

| | **BIN** | | **GRA** | | **UNA** | | **RND** | |
|---|---|---|---|---|---|---|---|---|
| **LEN** | $PP_g$ | $PP_p$ | $PP_g$ | $PP_p$ | $PP_g$ | $PP_p$ | $PP_g$ | $PP_p$ |
| 3 | 1.63 | 2.33 | 1.00 | 2.33 | 1.00 | 1.00 | 1.50 | 2.19 |
| 4 | 1.38 | 3.75 | 1.00 | 3.75 | 1.38 | 1.00 | 1.88 | 2.89 |
| 5 | 1.81 | 6.20 | 1.00 | 6.20 | 1.78 | 1.00 | 2.31 | 3.58 |
| 6 | 2.17 | 10.50 | 1.00 | 10.50 | 2.42 | 1.00 | 2.69 | 4.34 |
| 7 | 1.65 | 18.14 | 1.00 | 18.14 | 2.70 | 1.00 | 3.14 | 5.05 |
| 8 | 1.67 | 31.88 | 1.00 | 31.88 | 3.20 | 1.00 | 3.63 | 5.76 |
| 9 | 2.30 | 56.78 | 1.00 | 56.78 | 3.71 | 1.00 | 4.11 | 6.44 |
| 10 | 1.60 | 102.30 | 1.00 | 102.30 | 4.16 | 1.00 | 4.59 | 7.17 |

By ensuring that each neighboring phenotype differs by at most one bit, $PP_g$ for GRA is restricted to one for all $\ell$ considered. UNA and RND, though started with low values of $PP_g$, has a direct relationship with $\ell$. Lastly for BIN, $PP_g$ fluctuate around 1.7. As for $PP_p$, UNA obtains a value of one for $PP_p$ which signifies that for each neighboring genotype, their phenotype points only differ by one division. RND generally balances well between $PP_g$ and $PP_p$ for the various $\ell$ considered.

In general, the proposed RND alleviate the problem of positional dependency by introducing redundancy into the translation mapping. By doing so, higher locality in the genotype space is achieved. This essentially means that the structure of the phenotype search space is well retained after the transformation. Hence, the effect on the complexity of the problem is minimal.

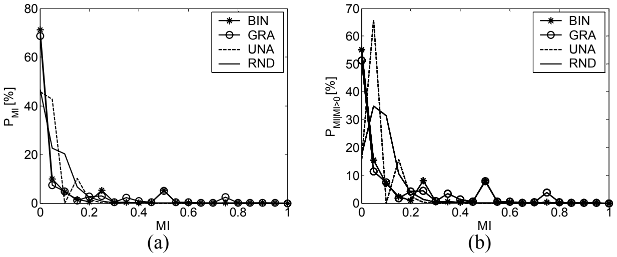
## V. EMPIRICAL STUDY

To gain further insight into the properties of the proposed coding scheme, this section will analyze its relationship between the mutation operators via measures and empirical techniques proposed by Raidl and Gottleib [11]. Mutation innovation (*MI*), which measures the phenotypic distance between solution and the mutated solution, will be considered here. *MI* is a random variable which quantifies the degree of "innovation" being introduced by the mutation operator, and its distribution can reflect several locality properties.

For this exercise, *MI* for the four representation schemes namely, BIN, GRA, UNA and RND, will be investigated empirically by randomly creating 100,000 binary strings of length 15 and applying bit-wise mutation with a mutation probability of 1/15. Fig. 4(a) shows the resulting distribution of *MI*.

In 35.57% of all the runs, *MI* attained a value of zero, reflecting the case where mutation does not affect the phenotype at all. The higher peak attained at *MI=0* by BIN and GRA suggests that there is a significant number of smaller movements due to mutation. Also, there are spikes at *MI* values of 0.5 and 0.75 respectively. This is due to the positional dependency characteristics of BIN and GRA, which causes huge movement, and hence larger values of *MI*, when the significant bits are altered. However, at high values of *MI*, $P_{MI}$ remain at zero for UNA and RND. But due to the limited resolution for UNA, the distribution is rather jerky, and for some low values of *MI*, the probability of occurrence is zero. Conversely, the distribution for RND is more gradual and smooth.

It is also instructive to consider the distribution for *MI* > 0, when mutation actually modifies the phenotype. Fig. 4(b) plots the resulting distribution. Consequently, the spikes for BIN and GRA are more pronounced now.



(a) (b)

Fig. 4. Empirical distributions of (a) MI and (b) (MI|MI>O) for the various coding schemes

TABLE V
*PP* OF THE VARIOUS CODES

| | BIN | GRA | UNA | RND |
|---|---|---|---|---|
| $E(MI)$ | 0.0977 | 0.1404 | 0.0704 | 0.0785 |
| $E(MI \mid MI > 0)$ | 0.0630 | 0.0977 | 0.0452 | 0.0506 |

Table V list the empirically obtained values for the expected values of *MI,* $E(MI)$ and the conditional expected value of *MI* when *MI* > 0, $E(MI \mid MI > 0)$ . In general, a small $E(MI \mid MI > 0)$ denotes high locality, where a single mutation only changes the phenotype slightly, and hence should be encouraged. On the other hand, large values suggest weak locality, which means that highly different solutions are frequently generated, analogous to a random search. From the table, codes with redundancy, namely UNA and RND, achieved lower value of $E(MI \mid MI > 0)$, signalizing high locality that is desirable for local search. While BIN and GRA have larger values, this might be advantageous in certain cases, for example, to escape from local optimum.

From this simple experiment, it further verifies the enhanced locality brought about by redundancy. It will certainly be interesting to consider other measures like crossover innovation and crossover loss and conduct further empirical studies to gain a deeper understanding of the difference between the various translation codes. Also, experimental studies involving actual optimization problems are necessary to affirm the viability and practicality of the proposed translation code.

## VI. Comparative study

In this section, the algorithmic performance for the various translation codes will be compared in some benchmark problems. The different translation codes and their corresponding index and notation are shown in Table VI. A single objective problem (SOP) and a multi objective problem (MOP) will be considered to evaluate the generality of the proposed translation code.

TABLE VI
DIFFERENT ENCODING SCHEME

| Encode scheme | Index | Notation |
|---|---|---|
| Binary code | 1 | BIN |
| Gray code | 2 | GRA |
| Unary code | 3 | UNA |
| Random code | 4 | RND |

The evolutionary platform adopted for the SOP will be a generic elitist EA which maintains a fixed-size population and an archive to store the non-dominated solutions discovered. Elitism is implemented by selecting mating individuals through a binary tournament selection from the combined archive and evolved offspring. The parameter configuration of the EA is summarized in table VII.

TABLE VII
PARAMETER SETTINGS FOR SOP EXPERIMENTS

| | |
|---|---|
| Chromosome | 30 bits per decision variables |
| Population | Population size of 100. Archive size of 100 |
| Selection | Binary Tournament Selection |
| $p_c$ | 0.8 |
| $p_m$ | $1 / \ell$ |
| Generation | 500 |
| Runs | 30 |

Over the years, many single objective problems have been designed for analyzing and comparing different optimization algorithms. In order to evaluate the proposed codes, ACK [12] have been chosen in this paper as it is considered to be more difficult and its degree of difficulty can be adjusted by changing the dimension of the search space, which will vary the number of local minima. Different dimensions were considered to evaluate the scalability of the codes.

The mean, median, minimum and the standard deviations of the various codes are tabulated in table VIII. Ideally, a code should have the lowest value in all four areas which symbolizes proximity and consistency. To supplement the results, the analysis of variance (ANOVA) and a multiple comparison test is used to test the significance of the mean difference. The best codes in each problem were bold. In events where the differences are not as significant, there might be more than one best code.

TABLE VIII
RESULTS FOR ACKLEY FUNCTION

| | | Mean | Median | Minimum | Standard Deviation |
|---|---|---|---|---|---|
| ACK -10 | BIN | 1.036 | 1.0212 | 0.5114 | 0.3586 |
| | **GRA** | 0.772 | 0.7443 | 0.4504 | 0.2273 |
| | UNA | 2.6392 | 2.9828 | 0.2587 | 1.1328 |
| | RND | 4.2981 | 4.2078 | 3.5413 | 0.3687 |
| ACK -20 | BIN | 6.3173 | 6.3367 | 4.9007 | 0.7571 |
| | **GRA** | 5.281 | 5.1848 | 4.3793 | 0.5635 |
| | **UNA** | 5.3095 | 5.178 | 3.9748 | 0.6083 |
| | RND | 6.7839 | 6.7952 | 5.726 | 0.5783 |
| ACK -30 | BIN | 11.788 | 11.8355 | 9.8539 | 0.9849 |
| | GRA | 9.2116 | 9.3812 | 7.4557 | 0.7574 |
| | **UNA** | 6.985 | 6.9355 | 6.0862 | 0.3956 |
| | RND | 8.3644 | 8.4077 | 7.0752 | 0.4565 |

BIN generally has the worst performance amongst the different codes. It is observed that GRA performs better with lower dimensions. But as the dimension increases, UNA and RND prove to be more effective and consistent.

Extending the analysis to a MOP, the evolutionary platform used earlier is extended by incorporating Pareto-based ranking technique and diversity preservation mechanism. The selection criterion is based on Pareto ranking and in the event of a tie, the niche count will be employed. The mechanism of niche sharing is used in the tournament selection as well as diversity maintenance in the archive. The additional parameter configuration of the MOEA adopted is summarized in table IX.

TABLE IX
PARAMETER SETTINGS FOR MOP EXPERIMENTS

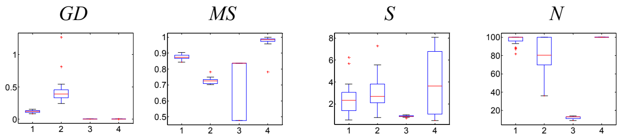
| | |
|---|---|
| Ranking scheme | Pareto ranking |
| Diversity | Niche count with radius 0.01 in the normalized objective space |
| Operator | |
| Generation | 200 |

The MOP considered, ZDT6 [13] has solutions that are non-uniformly distributed along $PF_{true}$ and the density of the solutions is uneven along $PF_{true}$. The difficulty of this problem is to deal with non-uniformities in the non-convex $PF_{true}$ and find a good distribution of points.

Unlike in single-objective optimization, there are several goals in multi-objective optimization [14]. Hence several metrics should be adopted for this matter. In this paper, Generational Distance, *GD* will be used to measure proximity i.e. convergence to the Pareto-optimal set. This metric denotes how "far" the Pareto front found ($PF_{known}$) is from the global Pareto front ($PF_{true}$) and a small *GD* signifies that the $PF_{known}$ is close to the $PF_{true}$. Besides achieving proximity, diversity which refers to how well $PF_{known}$ is defined, should also be achieved. Diversity will be measured by Maximum Spread, *MS* which indicates how well the $PF_{true}$ is covered by $PF_{known}$ and Spacing, *S* which reflects how "evenly" solutions in $PF_{known}$ are distributed. The last metric used will be a simple metric which measures the number of non-dominated solutions found, *N*. For a more complete performance assessment between the various operators, binary quality measures which rate the dominance relationship between pairs of solutions sets must be included [15] to supplement the above mentioned unary metrics. For this purpose, the coverage function, *C* which gives for a pair (A,B) of solutions sets the fraction of solutions in B that are weakly dominated by one or more solutions in A, was chosen.

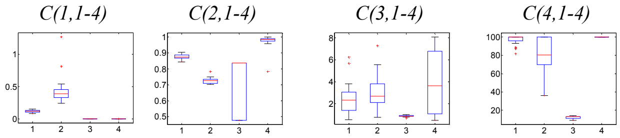
These metrics represent quantitative measures that can describe the quality of the final result of the selected operators which could validate the effectiveness of the proposed code schemes and they are illustrated by box plots to provide the statistical comparison results (Fig. 5-6). Similarly, the ANOVA and multiple comparison tests were performed to evaluate the significance of the mean differences between the various metrics and table X tabulates this relationship.

For ZDT6, RND is able to converge to the global optimum at all runs with an excellent spread. Coverage result shows that UNA and RND dominate the other two codes with the former dominating the latter. The Pareto fronts illustrated in Fig. 7 further affirmed this assertion.



(a) (b) (c) (d)

Fig. 5. Simulations results for ZDT6



(a) (b) (c) (d)

Fig. 6. Simulations results of *C* for ZDT6

TABLE X
RELATIONSHIP OF THE MEAN DIFFERENCES FOR THE METRICS

| ZTT6 | GD | | | | MS | | | | S | | | | N | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 | = | << | >> | >> | = | >> | >> | << | = | < | >> | << | = | >> | >> | < |
| 2 | >> | = | >> | >> | << | = | > | << | > | = | >> | << | << | = | >> | << |
| 3 | << | << | = | >> | << | < | = | << | << | << | = | << | << | << | = | << |
| 4 | << | << | << | = | >> | >> | >> | = | >> | >> | >> | = | > | >> | >> | = |

Note: '<' in row *x* column *y* means that the mean for index x is less then the mean for index y. '<<' signify significantly lesser than.

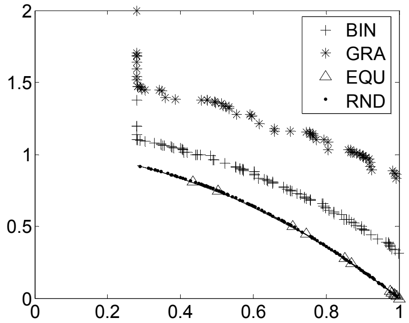


Fig. 7. Pareto fronts of ZDT6

## VII. REDUCING THE EFFECTS OF OVERREPRESENTATION

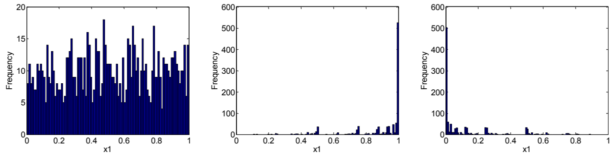
As discussed earlier, the central region of the phenotype space is overrepresented by RND, which is not ideal in situations where the optimal region cannot be specified or when it is distributed uniformly in the search space. This section will look at two approaches that could possibly resolve this issue. The first method will attempt to distribute the solutions uniformly during the generations of the initial population, while the second technique will modify the genotype-phenotype mapping with intermediary function so as to achieve a uniform distribution.

### A. Initialization

The typical algorithm for the generation of the initial population is shown in Fig. 8. Using a value of 0.5 for *M* will ensure that equal number of '0' and '1' are generated which correspond to phenotype points being uniformly distributed throughout the search space using BIN as shown in Fig. 9(a), whereas for random code, all the generated phenotype points will be concentrated in the centre as illustrated in Fig. 10(a). A change in the value of M will affect the balance of '0' and '1' causing bias towards certain region in the genotype space for the various codes. In general, choosing a small (large) M will decrease (increase) the occurrence of '0'. However, the effects of changing M for RND are more intuitive and obvious as compared to BIN, since it essentially just involve a shift in the epicentre of the distribution.

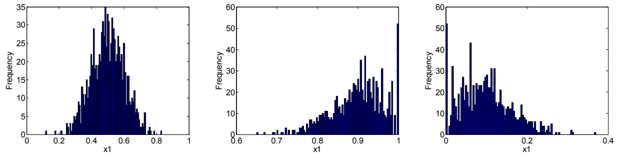
```
Loop (k for 1:chromosome length)
  IF (random number<M)
    chromosome [k]=0
  ELSE
    chromosome [k]=1
  End loop
```

Fig. 8. Pseudo code for generation of initial chromosome



(a) (b) (c)

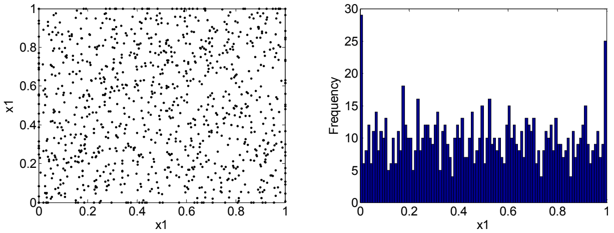
Fig. 9. Distribution of initial population for BIN using a) M=0.5, b) M= 0.1 c) M= 0.9



(a) (b) (c)

Fig. 10. Distribution of initial population for RND using a) M=0.5, b) M= 0.1 c) M= 0.9

Hence, a simple strategy to reduce the effects of overrepresentation for RND is to vary the value of M so that the epicentre of the distribution could be controlled and placed in any desired locations. An interesting example is shown in Fig. 11 where M is simply chosen to be a random number. This results in the initial population being distributed uniformly throughout the search space.



(a) (b)

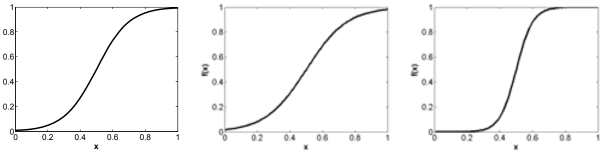
Fig. 11. RND initial population distribution for random M

### B. Intermediary Mapping

Overrepresentation can be resolved by passing the phenotype values through a function that can flatten the central region by mapping phenotype values away from the central region. A modified sigmoid function, S is proposed for this purpose and is defined in (13)

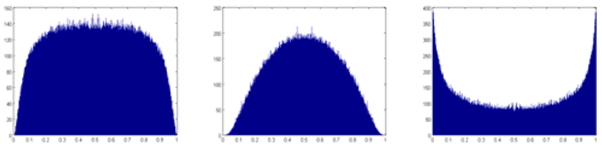
$$S(x_p) = \frac{\dfrac{1}{1+e^{-\alpha\beta\left(x_p-0.5\right)}} - S(-\beta)}{S(\beta)-S(-\beta)} \qquad (13)$$

where $x_p \in [0,1]$, $S(x_p) \in [0,1]$ ,α and β are constant. α affects the slope of the function in the middle region while β is just any arbitrary constant chosen to map the range to [0,1]. Fig. 12(a) and 13(a) illustrate the graphical plot of S and the resultant decision space mapping produced when α =0.5 and β=10. Comparing this to the original decision space mapping in Fig. 3(b), there is a stark drop of phenotype points representing the central region. All these points are shifted to the fringe.



(a) α =0.5 (b) α=0.4 (c) α=1.0

Fig. 12. S for different α



(a) α =0.5 (b) α=0.4 (c) α=1.0

Fig. 13. Phenotype distribution for different α

By varying the values of α, different S and decision space mapping can be obtained. Using lower values of α will linearise the sigmoid function, making overrepresentation in the central region more prevalent. Conversely, higher value of α will shift more phenotype points from the central to the fringe and in extreme cases when α=1.0, this will cause an overrepresentation in the fringe instead.

## VIII. Conclusion

A novel translation code, which improves locality at the expense of distribution for binary representation, was proposed. This was achieved by introducing redundancy into the translation mapping. The theoretical analysis reveals that RND is able to alleviate the problem of positional dependency in BIN and GRA and the empirical studies further affirmed the enhanced locality brought about by redundancy for RND. The simple empirical study conducted demonstrated the practicalities of RND. Possible avenues to improve the poor distribution for RND are also suggested. Future works include subjecting these various translation codes to other empirical studies to further understand the differences between the various translation codes. A comprehensive experimental study involving benchmark and real-life optimization problems is necessary to affirm the viability and practicality of RND. The results and progress will be reported in due course.

## References

[1] D. A. V. Veldhuizen and G. B. Lamont. "Multiobjective Evolutionary Algorithm Test Suites". *1999 {ACM} Symposium on Applied Computing*, pp. 351-357, 1999.

[2] K. C. Tan, S. C. Chiam and C. K. Goh. "Adapting Evolutionary Dynamics of Variation for Multiobjective Optimization". *IEEE Congress on Evolutionary Computation*, pp. 1290-1297, 2005.

[3] F. Rothlauf. "Locality, Distance Distortion, and Binary Representations of Integers". Working paper, University Mannheim, Department of Information Systems, 2003.

[4] M. Shackleton, R. Shipman and M. Ebner. "An investigation of redundant genotype-phenotype mappings and their role in evolutionary search". *IEEE Congress on Evolutionary Computation*, Vol. 1, pp. 493-500, 2000.

[5] F. Rothlauf and D. E, Goldberg. "Redundant Representations in Evolutionary Computation". *Evolutionary Computation*, Vol. 11, No. 4, pp. 381-415, 2003.

[6] T. Jones and S. Forrest. "Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms". *The Sixth International Conference on Genetic Algorithms*, pp. 184-192, 1995.

[7] B. Sendhoff, M. Kreutz and W. Seelen. "A Condition for the Genotype-Phenotype Mapping: Causality". *The Seventh International Conference on Genetic Algorithms*, 1997.

[8] S. C. Chiam, C. K. Goh and K. C.Tan. "Issues of Binary Representation in Evolutionary Algorithms". *The 2nd IEEE International Conference on Cybernetics & Intelligent Systems*, 2006.

[9] R. A. Caruana and J. D. Schaffer. "Representation and hidden bias: Gray v.s. binary coding for genetic algorithms". *Proceedings of the Fifth International Workshop on Machine Learning*, pp. 153-161, 1988.

[10] J. D. Schaffer. "Some Experiments in Machine Learning using Vector Evaluated Genetic Algorithms." *Ph.D. Thesis*, Vanderbilt University, 1980.

[11] G. R. Raidl and J. Gottlieb. "Empirical Analysis of Locality, Heritability and Heuristic Bias in Evolutionary Algorithms: A Case Study for the Multidimensional Knapsack Problem." *Evolutionary Computation*, Vol. 13, No. 4, pp. 441-475, 2005.

[12] D. H. Ackley, Connectionist Machine for Genetic Hillclimbing, Kluwer Academic Publishers, Boston, 1987.

[13] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, Vol. 8, No. 2, pp. 173-195, 2000.

[14] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms, John Wiley & Sons, 2001.

[15] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca and V. G. Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review", *IEEE Transactions on Evolutionary Computation*, Vol. 7, No. 2, pp. 117-132, 2003.