A Study for Global Optimization Using Dynamic Encoding Algorithm for Searches

Nam Geun Kim, Jong-Wook Kim and Sang Woo Kim

Division of Electrical and Computer Engineering, POSTECH, Pohang, Korea
(Tel: +82-54-279-5018; Fax: +82-54-279-2903; Email:{kimng, kjwooks, swkim}@postech.ac.kr)

Abstract: This paper analyzes properties of the recently developed nonlinear optimization method, Dynamic Encoding Algorithm for Searches (DEAS) [1]. DEAS locates local minima with binary strings (or binary matrices for multi-dimensional problems) by iterating the two operators; *bisectional search* (BSS) and *unidirectional search* (UDS). BSS increases binary strings by one digit (i.e., zero or one), while UDS performs increment or decrement to binary strings with no change of string length. Owing to these search routines, DEAS retains the optimization capability that combines the special features of several conventional optimization methods. In this paper, a special feature of BSS and UDS in DEAS is analyzed. In addition, a effective global search strategy is established by using information of DEAS. Effectiveness of the proposed global search strategy is validated through the well-known benchmark functions.

Keywords: DEAS, optimization, global search.

1. Introduction

Many practical engineering problem can be formulated as optimization problems using an objective function whose domain, *D*, represents the space of feasible solutions and whose range represents the relative utility of each solution. Solving the optimization problem is to find the solution in *D* for which the objective function obtains its smallest value, the global minimum. Global optimization thus aims at determining not just a local minimum but the smallest local minimum in the region *D*. Global optimization is an inherently difficult problem since no general criterion exists for determining whether the global optimum has been reached. Therefor, most global optimization algorithms are required infinitely many function evaluations [2].

In general, global optimization can be categorized into two different approaches; deterministic and stochastic methods, as they were termed in [3] and [4]. Deterministic methods can guarantee absolute success, but only by making certain restrictive assumptions, the cost function should be sufficiently smooth with twice differentiability, which is quite a rigorous condition to real-world problems. Stochastic methods evaluate the objective function at randomly generated points. The convergence results for these methods are not absolute. However, the probability of success approaches one as the sample size tends to infinity under very mild assumptions about the objective function. Not using the gradient information, these algorithms are more straightforward to apply but less efficient than deterministic methods in terms of their running time and solution qualities.

To make a more effective global optimization strategy, the two basic approaches are combined even if the basic philosophies behind the two approaches are quite different. In the sequel, a nonlinear optimization algorithm named dynamic encoding algorithm for searches (DEAS) is developed in the framework of discrete structures. DEAS is originated from the need of the compromise between deterministic and stochastic search methods. In other words, an iterative algorithm which directs a current minimum to successively approach to the optimum without derivative information. DEAS employs the peculiar property of a binary string that if a binary digit, 0 or 1, is concatenated to a binary string as a new least significant bit (LSB), the decoded real value of the new string is slightly decreased for 0 or increased for 1. In addition, if a binary string undergoes increment addition or decrement subtraction, the real-valued difference is equidistant under the same string length. In DEAS, these familiar characteristics of a binary string is applied to the determination of a search direction and step length, we can get the gradient information and varying rate of step length easily without another calculation process. GA is comparable to DEAS in that it requires no derivative information of a cost function and uses the same decoding function, but differs by its concatenated binary structure. However, since GA simulates the natural selection and evolution on a digital computer, the basic philosophies of GA and DEAS are quite different. Generally, most algorithms do not have well defined stopping conditions. In practice, some methods are halted after a fixed number of iterations, or when the step size becomes smaller than a given threshold. In DEAS, we can make reasonably stopping condition that stops the search when the length of increasing binary strings reaches the desired grid achieving an accuracy on *D*. In order to address general global optimization problems, however, it is necessary to develop an appropriate global optimization scheme for DEAS whose performance is verified by benchmark tests. In this paper, a multistart approach is embedded into DEAS with various revisit check functions for efficient global search.

The paper is organized as follows: Section 2 describes the local and global search strategies of the proposed method with illustrative examples. Section 3 gives the optimization results of the benchmark and parameter identification. Finally, section 4 gives conclusions.

2. Dynamic Encoding Algorithm for Searches

In this section, an optimization problem is defined and The basic search strategies of DEAS are introduced. In addition, several global search strategy are proposed.

2.1. Optimization Problem

Given a objective function $F(\mathbf{x})$ as a minimum problem without constraints, the aim of global optimization is to find $\mathbf{x}^* = [x_1^* \; x_2^* \; \cdots \; x_n^*]^T$ such that

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \{ \, F(\mathbf{x}) \mid \mathbf{x} \in \mathbb{R}^{\mathbf{n}} \}.$$

and the associated extreme value $F^* = F(\mathbf{x}^*)$ of the objective function $F(\mathbf{x})$, in this case the minimum. The expression $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$ means that the variables are allowed to take all real values; $\mathbf{x}$ can thus be represented by any point in an *n*-dimensional Euclidean space $\mathbb{R}^n$.

For a local minimum the following relationship holds:

$$F(\mathbf{x}^*) \leq \mathbf{F}(\mathbf{x}^*)$$

$$\text{for} \quad 0 \leq \|\mathbf{x} - \mathbf{x}^*\| = \sqrt{\sum_{\mathbf{i=1}}^{\mathbf{n}} (\mathbf{x_i} - \mathbf{x_i^*})^{\mathbf{2}}} \quad \leq \varepsilon, \quad \mathbf{x} \in \mathbb{R}^{\mathbf{n}}$$

This means that in the neighborhood of $\mathbf{x}^*$ defined by the size of ε there is no vector $\mathbf{x}$ for which $F(\mathbf{x})$ is smaller than $F(\mathbf{x}^*)$.
In DEAS, we account for the fact that DEAS procedure can only produce approximate answers. Hence we consider the problem solved if for some $\varepsilon > 0$ we find a solution in the *level set* L_d, where $d = f(\mathbf{x}) + \varepsilon$ and

$$L_d = \{\, \mathbf{x} \mid \mathbf{f}(\mathbf{x}) \leq \mathbf{d} \}.$$

This is said that a solution $\mathbf{x} \in \mathbf{L_d}$ is ε-accurate.

2.2. Basic Behavior of DEAS

The basic behavior of DEAS is described by two terminologies: *bisectional search* (BSS) and *unidirectional search* (UDS).

2.2.1 Bisectional Search

If a binary digit, 0 or 1, is appended to an existing binary string as a least significant bit (LSB), the decoded real number of a new binary string decreases for 0, and increases for 1 compared with that of the original binary string. This property is utilized in search algorithm, DEAS, to generate the neighborhood, the set of solution that can be reached from the current solution in a single iteration. The name 'bisectional search' comes from this dichotomous search aspect.
From the viewpoint of local search techniques, the BSS can be comprehended as the selection of the best neighborhood matrices among the 2^n neighborhood ones. In Fig. 1, the neighborhood strings generated by the BSS are depicted with points. Among those neighborhood matrices, the best one whose cost is lowest is saved as an optimal matrix.
The pseudocode for an *n*-dimensional problem of BSS is stated as follows,where BSS is undertaken for an $n \times m$ binary matrix. Note that n equals the number of search parameters, and bit number of string, m, can vary from 1 to any finite number.

Bisectional Search ($\mathbf{B}_{n\times m}$)

Initialization:

Set a direction vector as an all-zero binary string;

$\mathbf{d} = \mathbf{0}_{n\times 1}$.

Set $J_{min} \leftarrow$ temporary variable, $M \gg 1$ and $i \leftarrow 1$.

while $i \leq 2^n$ **do**

Add $\mathbf{d}$ as a least significant column;

$\mathbf{B}_{n\times(m+1)} = [\mathbf{B}_{n\times m} \;\; \mathbf{d}]$.

Decode a temporary matrix into a real vector as

$\mathbf{B}_{n\times(m+1)} \overset{f_d}{\rightarrow} \mathbf{X}_{\mathbf{n}\times\mathbf{1}}$.

Evaluate the cost;

$J = f(\mathbf{X})$.

if $J < J_{min}$ **then**

Save the current best optima;

$\mathbf{B}^*_{n\times(m+1)} \leftarrow \mathbf{B}_{n\times(m+1)}$,
$\mathbf{d}_{opt} \leftarrow \mathbf{d}$,
$\mathbf{X}_{min} \leftarrow \mathbf{X}$,
$J_{min} \leftarrow J$.

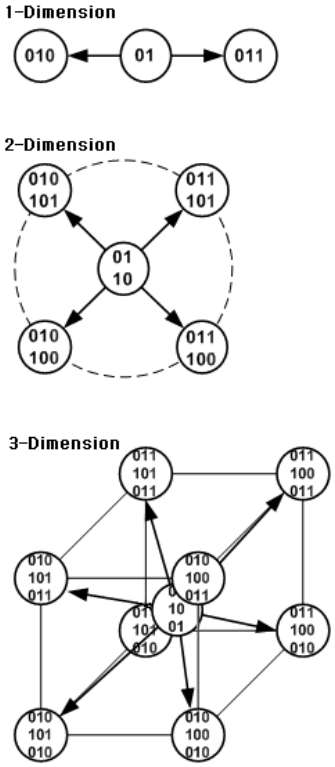

Fig. 1. One, two, and three dimensional diagrams of neighborhood points generated by BSS

end if

$\mathbf{d} \leftarrow \mathbf{d} + 1$

$i \leftarrow i + 1$

end while

Let an *i*-th dimensional *m*-bit-long binary string,

$$\mathbf{B}_{i,m} = b_m \cdots b_j \cdots b_1, \;\; b_j \in \{0, 1\}, \; j = 1, \cdots, m,$$

be termed a *parent string*. If 0 is added to the LSB of $\mathbf{B}_{i,m}$, the new string is termed a *'left-hand child string'*, $\mathbf{B}^l_{i,m+1}$. On the other hand, if 1 is added, it is termed a *'right-hand child string'*, $\mathbf{B}^r_{i,m+1}$.
In the process of DEAS, the mechanism of decoding from the binary vector space $\{0, 1\}^m$ to restricted continuous space $\mathbb{R}^n$ on finite intervals $[u_i, v_i]$ for each variable $\mathbf{x^i} \in \mathbb{R}$ is required. According to the standard binary decoding function $f^i_d : \{0, 1\}^m \rightarrow [u_i, v_i]$, where [5], the real value of the parent string is

$$x_i = f_d(\mathbf{B}_{i,m}) = u_i + \frac{v_i - u_i}{2^m - 1} \sum_{j=1}^{m} b_j 2^{j-1}, \qquad (1)$$

and the child strings are respectively

$$\begin{aligned} x^l_{i+1} &= f_d(\mathbf{B}^l_{i,m+1}) = u_i + \frac{v_i - u_i}{2^{m+1} - 1} \sum_{j=1}^{m} b_j 2^j \\ &= \frac{2^{m+1} - 2}{2^{m+1} - 1} \; x_i, \\ x^r_{i+1} &= f_d(\mathbf{B}^r_{i,m+1}) = u_i + \frac{v_i - u_i}{2^{m+1} - 1} \sum_{j=1}^{m} b_j 2^j + \frac{v_i - u_i}{2^{m+1} - 1} \\ &= \frac{1}{2^{m+1} - 1} + \frac{2^{m+1} - 2}{2^{m+1} - 1} \; x_i. \end{aligned}$$

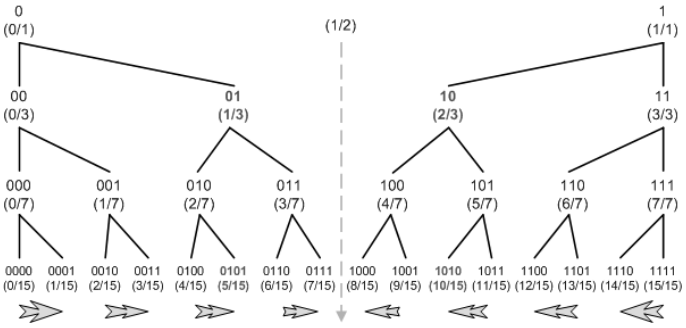

Fig. 2. Biased binary tree structures together with corresponding decoded values in parenthesis

Difference between the parent string and the child strings is the search step size that is important factor to determine the convergence velocity of DEAS and is given by

$$\begin{aligned} \mid x_{i+1}^{l} - x_i \mid &= \frac{1}{2^{m+1}-1}\ x_i, \\ \mid x_{i+1}^{r} - x_i \mid &= \frac{1}{2^{m+1}-1}\ (1 - x_i). \end{aligned}$$

And the comparison of step size between $\mid x_{i+1}^{l} - x_i \mid$ and $\mid x_{i+1}^{r} - x_i \mid$ is given by

$$\mid x_{i+1}^{r} - x_i \mid - \mid x_{i+1}^{l} - x_i \mid = \frac{1}{2^{m+1}-1}\ (1 - 2x_i), \qquad (2)$$

$$\begin{cases} \mid x_{i+1}^{l} - x_i \mid < \mid x_{i+1}^{r} - x_i \mid, & \text{for } x_i < 0.5, \\ \mid x_{i+1}^{l} - x_i \mid = \mid x_{i+1}^{r} - x_i \mid, & \text{for } x_i = 0.5, \\ \mid x_{i+1}^{l} - x_i \mid > \mid x_{i+1}^{r} - x_i \mid, & \text{for } x_i > 0.5. \end{cases}$$

From (2), the differences between a parent string and its two child strings, i.e. $\mid x_{i+1}^{l} - x_i \mid$ and $\mid x_{i+1}^{r} - x_i \mid$, vary according to the position of the parent string in the tree, and the magnitude of the differences decrease exponentially as the length of a parent string increases. In Fig. 2, the property of (2) is clearly visualized in the form of binary trees whose nodes are represented with binary strings and corresponding real numbers. In this way, the standard binary decoding function, (1) has biased search tendency to incline its steps toward the middle point of the finite intervals $[u_i, v_i]$ and to depend on the location of the optimal point between the finite intervals $[u_i, v_i]$ with respect to efficiency. From the viewpoint of global search techniques, the neighborhood operator is possible to guarantee asymptotic global convergence when it meets an symmetry condition more than a bias condition [6]. Therefor, The unbiased binary decoding function is designed like this

$$x_i = f_d(\mathbf{B}_{i,m}) = u_i + \frac{v_i - u_i}{2^{m+1}}\left(\sum_{j=1}^{m} b_j 2^j + 1\right), \qquad (3)$$

and the child strings are respectively

$$\begin{aligned} x_{i+1}^{l} &= f_d(\mathbf{B}_{i,m+1}^{l}) = u_i + \frac{v_i - u_i}{2^{m+2}}\left(\sum_{j=1}^{m} b_j 2^{j+1} + 1\right) \\ &= x_i - \frac{1}{2^{m+1}}, \\ x_{i+1}^{r} &= f_d(\mathbf{B}_{i,m+1}^{r}) = u_i + \frac{v_i - u_i}{2^{m+2}}\left(\sum_{j=1}^{m} b_j 2^{j+1} + 3\right) \\ &= x_i + \frac{1}{2^{m+1}}. \end{aligned}$$

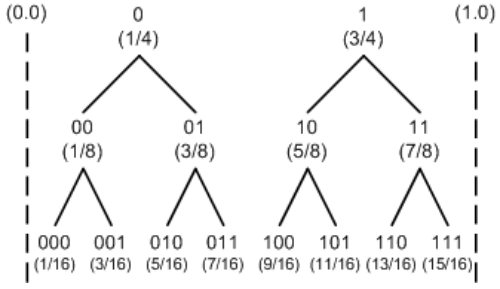

Fig. 3. Unbiased binary tree structures together with corresponding decoded values in parenthesis

Fig. 3 shows that the unbiased binary decoding function guarantees symmetric search property. The differences between a parent string and its two child strings, $\mid x_{i+1}^{l} - x_i \mid$ and $\mid x_{i+1}^{r} - x_i \mid$ are the same length.

The unbiased binary decoding function has good character that the real values of the decoded binary strings can not overlap each other, however, this advantage decreases with increasing dimensionality n.

2.2.2 Unidirectional Search

BSS alone, however, has a serious drawback of regional limitation. The reachable search area by pure BSS is $1/2$ of intervals $[u_i, v_i]$. However, the simple operations of increment addition (INC) and decrement subtraction (DEC) for a binary string can readily remove the barrier between any binary trees. In a unidirectional search

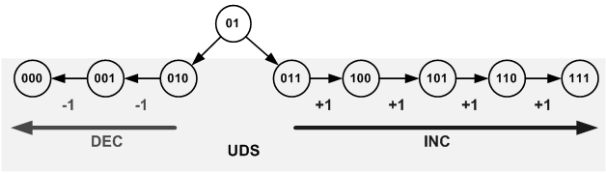

Fig. 4. Process of INC and DEC in a one dimension

(UDS) of DEAS, these operations are iterated while a better optimum is sought in a guided search direction obtained by the previous BSS. Fig. 4 illustrates the process of INC and DEC in UDS.

In Fig. 5, a session of DEAS in a two-dimensional problem, whose local minimum exists in a lower right part, is depicted with binary matrices. For an optimal direction to be discovered, BSS evaluates the cost function four, or 2^2, times, and $[1\ 0]^T$ is selected as an optimal direction vector (DV) of this session, which is handed over to UDS for further exploration. Then, UDS continues to search in three, or $2^2 - 1$, directions per transition by extension vectors (EV) while better solutions are being sought. For notational convenience, a binary digit 0 of each EV means that an extension has not occurred in a given direction, and vice versa. An optimal EV after the first transition in UDS is described as $[0\ 1]^T$, since a current best solution is obtained by moving along the direction of x_2.

Note that, among the second neighborhood in UDS, the matrix A is included again, and is thus evaluated redundantly. This situation also recurs at a fourth neighborhood, related with the previous optimal EV's that are depicted with thick solid lines in Fig. 5. After a transition in UDS, an extension coordinate is shifted in the selected

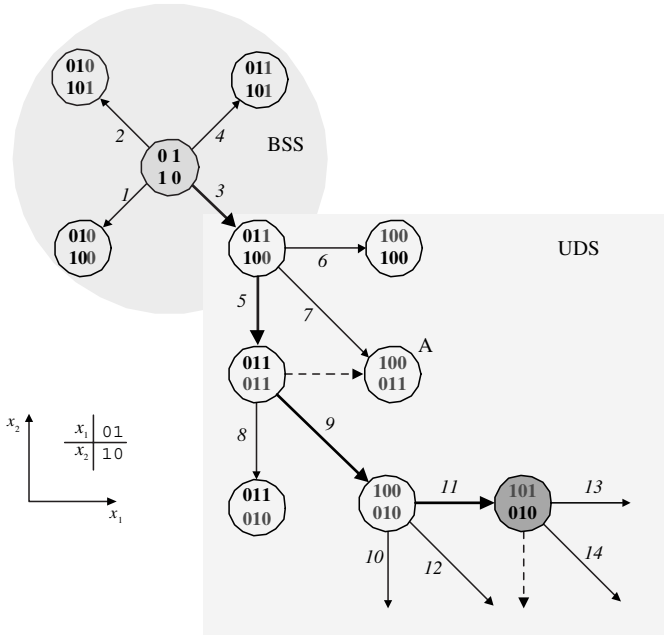

Fig. 5. Search aspect of DEAS in a two-dimensional problem (solid: basic direction, thick and solid: selected direction, dash-dotted: redundant direction)

directions. Thus, those extensions whose previously-selected directions remain unextended will revisit the points already sought. It is unexpectedly simple to implement the redundancy check in UDS by referring to a masking technique in microprocessors as [7]

1	0	0	Previous optimal EV ($\mathbf{e'_{opt}}$)
X	X	X	Current EV ($\mathbf{e}$)
X	0	0	Masked result

Let the previous optimal EV be $\mathbf{e'_{opt}}$ and the current EV be $\mathbf{e}$. If all the bits represented as 'X' in the masked result are simultaneously zero, the corresponding EV is regarded as a redundant search direction. Therefore, for example, revisiting EV's for $[1\ 0\ 0]^T$ as a previous optimal EV are $[0\ 0\ 1]^T$, $[0\ 1\ 0]^T$, and $[0\ 1\ 1]^T$ except $[0\ 0\ 0]^T$ which is basically excluded in UDS.

In conjunction with the pseudo-code of BSS, the UDS routine is written in detail as follows.

Unidirectional Search ($\mathbf{B}^*_{n\times k}$, $\mathbf{d}_{opt}$, J_{min}):
  Load $\mathbf{B}^*_{n\times k}$, $\mathbf{d}_{opt}$ and J_{min} of BSS
Initialization: $\mathbf{U}^*_{n\times k} \leftarrow \mathbf{B}^*_{n\times k}$.
while A better solution is attained **do**
  Set an extension vector; $\mathbf{e} = [\underbrace{0\ 0 \cdots\ 0}_{l-1}\ 1]^T$.
  for $i = 1 : 2^l - 1$
    Redundancy check:
      if $\mathbf{e}$ is computed by $\mathbf{e_{opt}}$ as a reevaluating direction **then** CONTINUE.
    Load the current best matrix into a temporary matrix;
      $\mathbf{T}_{n\times k} \leftarrow \mathbf{U}^*_{n\times k}$.
    for $j = 1 : l$
      $p = \mathbf{u}(j)$.
      Select the p-th row of $\mathbf{T}$; $\mathbf{r^T_{k\times 1}} = \mathbf{T}(\mathbf{p}, \mathbf{1} : \mathbf{k})$.
      if $\mathbf{e}(j) = 1$ **then**
        if $\mathbf{d}_{opt}(p) = 0$ **then** $\mathbf{r} = \mathbf{r} - 1$.
        else $\mathbf{r} = \mathbf{r} + 1$.
      end if
      $\mathbf{T}(\mathbf{p}, \mathbf{1} : \mathbf{k}) = \mathbf{r^T}$.
    end for
    Decode the modified binary matrix into a real vector:
      $\mathbf{T}_{n\times k} \xrightarrow{f_d} \mathbf{X}_{n\times 1}$.
    Evaluate the cost; $J = f(\mathbf{X})$.
    if $J \le J_{min}$ **then**
      Save the current best optimum;
        $\mathbf{U}^*_{n\times k} \leftarrow \mathbf{T}$,
        $\mathbf{e'_{opt}} \leftarrow \mathbf{e}$,
        $\mathbf{X}_{min} \leftarrow \mathbf{X}$,
        $J_{min} \leftarrow J$.
    end if
    $\mathbf{e} \leftarrow \mathbf{e} + 1$
  end for
  $\mathbf{e_{opt}} \leftarrow \mathbf{e'_{opt}}$
end while

From the viewpoint of a local optimization strategy, successive reformation of binary strings in BSS and UDS corresponds to the determination of optimal search directions and step lengths, respectively, and a neighborhood of a current solution is mapped to all the vertexes of gradually shrinking hypercubes in multi-dimensional problems. Therefore, DEAS is deemed to retain the unique property of a dynamic-sized neighborhood selection, while most combinatorial optimization algorithms use a fixed-sized neighborhood selection. A combination of BSS and UDS for a given string length, referred to as a *session* hereafter, is repeated with gradually increasing binary strings.

2.3. Global Search Strategies

In this section, global search strategies using information of DEAS are proposed.

2.3.1 History Check

In searching with DEAS, the whole search space of interest is dynamically divided by a fixed number of grids. It implies that the search path of each binary string is automatically determined according to the cost function landscape unless it is time-variant. That is, if DEAS is applied to any two identical strings, the subsequent search process will yield exactly the same search results. Moreover, DEAS enables UDS to search horizontally. Therefore, search paths of different initial matrices can encounter at one of the subordinate vertexes. Since this revisit gives rise to unnecessary computation, it must be detected and prohibited at every instance of session starting. The simplest way to design this history check function is to save a current binary matrix in a lookup table and compare it sequentially with the past data. This bit-by-bit comparison, however, can cause the problems of storage and computational time. As a solution to this problem, an integer- or real-valued representative for a string concatenated from a matrix is saved and compared in the lookup table by the following assignment function:

$$f_a(b_{m-1}b_{m-2}\cdots b_1b_0) = \sum_{j=0}^{m-1} b_j a^j \qquad (4)$$

where $b_i \in \{0, 1\}$, and a is an integer or real base value. As an illustration, a binary matrix is assigned an integer value by a series

of concatenations and an assignment with (4) of base 2 as

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} \Rightarrow 6$$

Then, every newly assigned value is compared with those of previous optimal matrices whose corresponding row lengths are identical. If an history table is constructed in the fashion of Table 1, which is attained for the Branin function [4], a new member is horizontally compared and enrolled in the case of no identical values. Note that a revisit has occurred at the fourth restart with the row length of 2, and subsequent values are identical to those of the third column. Therefore, the revisit is efficiently prohibited by the aid of a history table in DEAS.

Initialize the history table.
For $restart = 1 : maxRestart$
  Randomly generate an initial binary matrix; $\mathbf{T}_{n\times initLen}$
  HISTORY CHECK:
    Check if the initial matrix has already been visited.
  For $m = initLen : finLen$
    $\left[\, \mathbf{B}^*_{n\times(m+1)},\ \mathbf{d}_{opt},\ J_{min} \right]$=**BSS (** $\mathbf{T}_{n\times m}$**)**
    $\mathbf{B}^*_{n\times(m+1)}$=**UDS (** $\mathbf{B}^*_{\mathbf{n}\times(\mathbf{m+1})}$, $\mathbf{d}_{opt}$, J_{min}**)**
    HISTORY CHECK:
      Check if $\mathbf{B}^*_{\mathbf{n}\times(\mathbf{m+1})}$ has already been visited.
    $\mathbf{T}_{n\times(m+1)} = \mathbf{B}^*_{\mathbf{n}\times(\mathbf{m+1})}$
  end for
end for

The abovementioned global search strategy is shown as simplified pseudocodes. In an n-dimensional problem, the cost function is evaluated 2^n and $2^{l(k)} - 1 - r(k)$ times for BSS and UDS, respectively, where $l(k)$ and $r(k)$ represent the number of permitted variables and the number of redundant searches at the k-th transition, respectively.

2.3.2 Preliminary Search

An additional important feature for global optimization is an escaping scheme. As optimal binary matrices are lengthened, their step lengths decrease exponentially, which means that search points continuously converge to one of the local minima. Thus when successive cost values do not drop below a predefined acceptable value, it can be inferred that a current point falls inside the region of attraction (ROA) of an unacceptable local minimum. For this situation, DEAS commands to escape from a current point and restart, which is similar to the multistart method.

However, it is difficult to theoretically determine the optimal indexes of escaping, which is common to all global optimization methods. DEAS resorts to a kind of heuristics, i.e., surveying the landscape of the cost function by a finite number of trials as a preliminary search. Fig. 6 shows the sketch of the preliminary search over the Goldstein-price function [4], where the row lengths of initial matrices range from 1 to 3 with the maximal added row length of 9, and random restarts are conducted three times for each initial row length. This brief survey provides such significant information as *optInitRowLen*, *rowIndRestart*, and *costIndRestart*. The *optInitRowLen* means an optimal initial row length by which a global minimum can be discovered. Since step lengths are larger with a smaller row length, a small *optInitRowLen* means ROAs of global minima spread wide over the whole search space. The *rowIndRestart* and

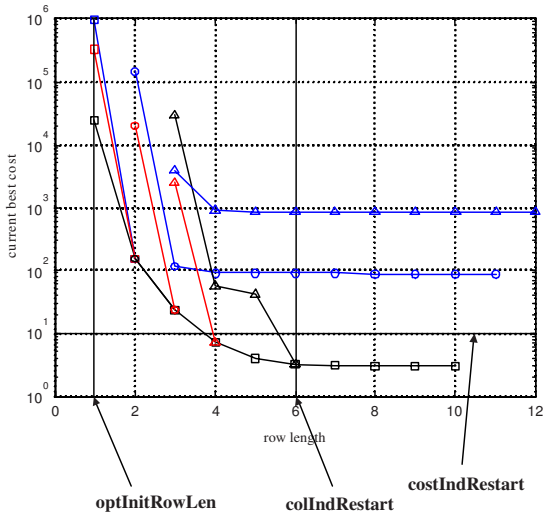

Fig. 6. Global indices of DEAS at a preliminary search with the Goldstein-price function

costIndRestart represent optimal indices of a row length and a cost value for restarts by which it is determined to restart or to continue. The *rowIndRestart* is selected as the minimal row length from which global minima and local minima can be discriminated with naked eyes, and *costIndRestart* is an approximately intermediate value between the best and the second-best local minima. These values are closely related to the smoothness, the ratios of ROAs, and the slopes of the cost function currently handled. Therefore, the preliminary search of DEAS additionally provides approximate information of the cost function.

2.3.3 Hill Climbing Search

When current point falls inside the ROA of an unacceptable local minimum, UDS fails because a better solution is not attained. On this occasion, we make UDS work as many as the success number of previous UDS. Like this, UDS gets extended exploration ability.

3. Benchmark and Analysis of Result

This section presents a performance comparison of DEAS using several standard multi-dimensional test functions, whose functional values and coordinates of each global minima are already reported in [8]-[13].

Table 1. Comparison of DEAS and other algorithms with the number of function evaluations.

	SDE	EA	MLSL	IA	TUN	TS	DEAS
BR	2700	430	206	1354	-	492	94
CA	10822	-	-	326	1469	-	87
GP	5439	460	148	-	-	486	103
RA	-	2048	-	-	-	540	304
SH	241215	-	-	7424	12160	727	137
H3	3416	-	197	-	-	508	189
H6	-	-	-	-	-	2845	1760

Table 1 provides the number of function evaluations computed by conventional global optimization methods and DEAS. Abbreviations for the methods: SDE is the stochastic method of Aluffi-Pentini *et al*. [8]; EA denotes the evolution algorithm of Yong *et al*. [9]; MLSL is the multiple-level single-linkage method of Kan and Timmer [10]; IA is the interval arithmetic technique of Ratschek and Rokne [11]; TUN is the tunneling method of Levy and Montalvo [12]; and TS refers to the tabu search scheme of Cvijovic and Klinowski[13].

The overall termination condition for the whole methods is set as the achievement of the desired accuracy, $\epsilon_1 (= f - f^*) = 10^{-6}$, for fair comparison [4]. The optimization results demonstrate that DEAS is considerably faster than the conventional methods.

This superiority comes from the property that DEAS works with the binary representation. The first advantage is a fast convergence rate owing to BSS whose underlying principle is similar with Branch-and-Bound methods and the dichotomous search and the interval halving method in nonlinear optimization [14]. The binary representation of real numbers in DEAS establishes the second merit which facilitates various revisit checks during local optimization and history check in global optimization. The revisit checks routines are quite easy to implement as is mentioned in Section 2. An additional merit of the binary representation unique to DEAS is that determining the stop condition is remarkably straightforward. Since row lengths of current best matrices are exponentially proportional to search space resolution, maximum row length can be assigned to DEAS considering acceptable parameter resolution. For example, if resolution of 1/100 is considered enough for all parameters, maximum row length is computed to be 7 ($2^7 = 128$). In a similar fashion, another important factors for restart such as *optInitRowLen* and *colIndRestart* are readily determined. In addition to the gradient-free property, this convenience is beneficial to the users of DEAS.

4. Conclusion

This paper analyzes properties of the recently developed nonlinear optimization method, DEAS. To implement global optimization, the multistart approach is adopted as an overall structure, and three revisit check functions are embedded inside DEAS to avoid unnecessary computation.

The performance of DEAS has been verified by several test functions, which confirm that its search speed is faster than the conventional methods for small scaled problems. Furthermore, DEAS is also successfully applied to the identification of an induction motor with experimental data to test its viability to real-world problems.

DEAS corresponds to the direct method, which is not constrained to the twice differentiability of a cost function. Therefore, the proposed method has strong advantages in dealing with practical problems such as the gain tuning of controllers, system identification of electrical apparatuses. For further development, the inherent exponential complexity of DEAS has to be overcome by alternative approaches such as the univariate or modular neighboring scheme to be able to handle large scaled optimization problems.

References

[1] Jong-Wook Kim, *Dynamic Encoding Algorithm for Searches and Its Application to Control*, PhD Thesis, Electrical and Computer Engineering Division - Pohang University of Science and Technology, 2004.

[2] L.C.W. Dixon, *Global optima without convexity*, Numerical Optimization Centre, Hatfeld Polytechnic, Hatfield, England, 1978.

[3] L.C.W. Dixon and G.P. Szegö, *The global optimization: An Introduction*, in: Dixon and Szegö, eds., *Towards global optimization 2*, North-Holland, Amsterdam, 1978,pp. 1-15.

[4] A. Törn and A. Žilinskas, *Global Optimization*, Springer-Verlag, Berlin, 1989.

[5] T. Bäck, *Evolutionary Algorithm in Theory and Practice*, Oxford University Press, New York, 1996.

[6] E. H. L. Aarts and J. H. M. Korst, *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*, John Wiley & Sons, Chichester, 1989.

[7] B. B. Brey, *The Intel 32-Bit Microprocessors*, Prentice-Hall Inc. Englewood Cliffs, 1995.

[8] F. Aluffi-Pentini, V. Paris, and F. Zirilli, "Global optimization and stochastic differential equations," *Journal of Optimization Theory and Applications*, vol. 47, pp. 1–15, 1985.

[9] L. Yong, K. Lishan, and D. J. Evans, "The annealing evolution algorithm as function optimizer," *Parallel Computing*, vol. 21, no. 3, pp. 389–400, 1995.

[10] A. H. G. R. Kan and G. T. Timmer, "A stochastic approach to global optimization," *Numerical Optimization*, Edited by P. T. Boggs, R. H. Byrd, and R. B. Schnabel, SIAM, Philadelphia, Pennsylvania, pp. 245–262, 1985.

[11] H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester, UK, 1988.

[12] A. V. Levy and A. Montalvo, "The tunneling algorithm for the global minimization of functions," *SIAM Journal of Scientific and Statistical Computing*, vol. 6, pp. 15–29, 1985.

[13] D. Cvijovic and J. Klinowski, "Taboo search: an approach to the multiple minima problem," *Science*, vol. 267, pp. 664–666, 1995.

[14] S. S. Rao, *Engineering Optimization*, John Wiley & Sons Inc., 1996.