최적화기법인 DEAS를 이용한 비용함수의 형상정보 추출

Extraction of Shape Information of Cost Function Using Dynamic Encoding Algorithm for Searches(DEAS)

김 종 욱, 박 영 수, 김 태 규, 김 상 우*
(Jong-Wook Kim, Youngsu Park, Taegyu Kim, and Sang Woo Kim)

Abstract : This paper proposes a new measure of cost function ruggedness in local optimization with DEAS. DEAS is a computational optimization method developed since 2002 and has been applied to various engineering fields with success. Since DEAS is a recent optimization method which is rarely introduced in Korean, this paper first provides a brief overview and description of DEAS. In minimizing cost function with this non-gradient method, information on function shape measured automatically will enhance search capability. Considering the search strategies of DEAS are well designed with binary matrix structures, analysis of search behaviors will produce beneficial shape information. This paper deals with a simple quadratic function contained with various magnitudes of noise, and DEAS finds local minimum yielding ruggedness measure of given cost function. The proposed shape information will be directly used in improving DEAS performance in future work.

Keywords : computational optimization, Dynamic Encoding Algorithm for Searches(DEAS), fitness landscape, cost function

I. 서론

Dynamic Encoding Algorithm for Searches(DEAS, 동부호화 최적화기법)는 비용함수(cost function)의 미분 정보를 사용하지 않고, 잘 계획된 컴퓨터 연산만으로 비용함수의 전역 최적해를 탐색하는 최적화 기법이다[1]. DEAS는 2002년에 발표된 이래로 PID 제어기 설계[2], 가열로의 최적 온도패턴 설계[3], 유도전동기의 파라미터 식별[4], SVM(State Vector Machine)의 파라미터 최적화[5], 변압기 코아형상 최적설계[6] 등 여러 분야에서 만족할 만한 최적화 성능을 보였다.

DEAS의 장점은 0과 1의 이진수로 해를 표현함에서 비롯되는데, 컴퓨터에서 폴더나 파일을 검색할 때 사용되는 binary search나 비선형 최적화 시 많이 사용되는 interval halving method 등이 기본적으로 이진 탐색을 함으로써 탐색 속도가 상당히 빠른 사실이 이를 뒷받침한다. DEAS 는 이진 스트링이나 이진 행렬로 표현되는 현재 해의 가장 오른쪽 비트(Least Significant Bit, LSB)나 열(column)에 0과 1 혹은 이진 벡터를 붙임으로써 탐색 방향을 찾고, 찾아진 방향에 대해 비용함수가 최소가 될 때까지 증가(increment addition) 혹은 감소(decrement subtraction)를 수행함으로써 주변 지역으로 탐색을 확장한다. 이러한 '방향 탐색-확장 탐색'이 session으로 결합되어서 탐색이 진행되므로 유전 알고리듬[7]에서 거론되는 심화탐색(exploitation)과 확장탐색(exploration)이 DEAS에서는 매 session마다 조화롭게 작용한다고 할 수 있다.

DEAS의 또 다른 장점은 해를 이진수로 표현하기 때문에 현재의 탐색점이 이전에 탐색되었는지의 여부와 현재 스트링 길이에 대한 한계 스트링(비트가 모두 0인 스트링 혹은 모두 1인 스트링)에 도달했는지를 손쉽게 판단할 수 있다. 아울러 현재 해의 스트링 길이를 측정해 보면 탐색의 수렴도를 직접적으로 알 수 있고, 지역/전역 탐색 종료에 관한 판단을 쉽게 결정할 수 있다[8].

초기의 DEAS는 현재 해 주변의 모든 가능한 이진 행렬을 생성하고 평가함으로써, 해의 차수가 클수록 이웃 해의 개수가 지수적으로 증가하는 문제점이 있었다. 이러한 문제를 해결하기 위해 변수 별로 차례로 '방향 탐색-확장 탐색'을 수행하는 탐색 전략을 개발했으며 이를 함수 최적화(function optimization)와 유도 전동기의 파라미터 식별에 적용한 결과 작은 계산량으로도 우수한 최적화 결과를 얻을 수 있었다[9]. 초기 DEAS는 주변의 해를 샅샅이 탐색한다는 의미에서 exhaustive DEAS(eDEAS)라고 명명했으며, 변수별 탐색을 수행하는 방법의 DEAS는 univariate method[10]의 이름을 따서 univariate DEAS(uDEAS)라고 명명했다. 대체적으로 10차 이상의 경우 eDEAS보다 uDEAS를 사용하는 것이 계산시간 면에서 유리하다.

DEAS는 전역탐색을 위해 비용함수의 형상 특성을 개략적으로 파악할 수 있는 예비탐색(preliminary search)을 수행한다[8]. 예비 탐색은 일정하게 길이가 증가하는 임의의 이진 스트링들을 초기 탐색점으로 선택한 후 그로부터 지역탐색(local search)을 수행하고, 그 결과로부터 얻어진 비용함수 값의 감소 추이들을 분석함으로써 비용함수의 개략적인 형상정보를 유추하는 탐색법이다. 예비탐색으로부터 얻어진 전역탐색 지수들을 이용하면 본 탐색(main search)에서 효과적으로 전역 최적해를 찾아낼 수 있다. 하지만, 파라미터 식별이나 제어기 이득 조절을 온라인 상태에서 연속적으로 수행할 경우 비용함수의 형상은 조금씩 변하게 되므로 예비탐색에서 설정된 전역탐색 지수 만으로는 탐색 효율이 점차 저하된다. 그리고 비용함수에 잡음이 더해져서 형상이 부드럽지 못한 경우, DEAS가 스트링 길이를 필요 이상으로 늘리며 탐색을 수행하는 것은 비효율적이다. 그러므로 비용함수의 형상을

* 책임저자(Corresponding Author)
논문접수 : 2007. 2. 20., 채택확정 : 2007. 6. 21.
김종욱, 김태규 : 동아대학교 전자공학과
(kjwook@dau.ac.kr/taegyuya@hotmail.com)
박영수, 김상우 : 포항공과대학교 전자전기공학과
(youngsu@postech.ac.kr/swkim@postech.ac.kr)
※ 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2006-(C1090-0602-0013)).

탐색 시 계속적으로 감지할 수 있는 방법이 필요하며 이를 DEAS가 이용할 경우 탐색 효율을 높일 수 있을 것이다.

본 논문에서는 탐색 과정 중에서 비용함수의 형상정보 중 잡음의 존재 여부를 감지할 수 있는 방법을 제안한다. 이를 위해 부드러운 함수와 거친 함수에 대한 DEAS의 탐색패턴을 분석하고, 잡음에 의해 발생한 불규칙적인 탐색패턴을 추출하여 발생횟수를 거칠기 지수(ruggedness measure)로 정량화한다. 제안한 매끄러움도의 검증을 위해 다양한 크기의 잡음이 섞인 2차 함수(quadratic function)에 대해 DEAS를 수행하고 비교한다. 1차원 문제의 경우 eDEAS와 uDEAS가 동일하며, 시각적으로 명확하게 이해할 수 있으므로 본 논문에서는 1차원 문제를 다루도록 한다.

2장에서는 DEAS의 탐색원리를 간략히 설명하고, 3장에서는 비용함수의 거칠기 지수를 탐색패턴의 분석을 통해 정의한다. 4장에서는 잡음의 크기를 변화시키며 DEAS를 수행함으로써 제안된 매끄러움도의 적절성을 검증하며, 5장에서는 결론을 맺는다.

II. DEAS 탐색 원리

이 장에서는 DEAS의 기본 탐색전략인 지역탐색 전략과 전역탐색 전략에 대해 설명한다.

1. 지역탐색 전략

DEAS는 현재 이진 스트링의 LSB에 0이나 1을 붙임으로써 적절한 탐색 방향을 찾고, 찾아진 방향에 대해 비용함수가 최소가 될 때까지 이진 스트링을 증가(increment addition) 혹은 감소(decrement subtraction)시킨다. 전자를 탐색양상을 따라서 bisectional search(BSS), 후자를 unidirectional search(UDS)라 한다. 그리고, 이러한 'BSS와 UDS 짝'을 session이라 한다.

1.1 BSS

다음의 정리는 BSS의 탐색 특성을 설명한다.

정리 1: m 비트 길이의 스트링 $s_p=[a_m a_{m-1}\cdots a_i\cdots a_1]$, $a_i\in\{0,1\}$, $i=1,\ldots,m$ 을 부모 스트링이라 하고, s_p 의 LSB에 0을 붙인 것을 좌측 자녀 스트링 s_c^l, 1을 붙인 것을 우측 자녀 스트링 s_c^r 이라 하며 다음과 같이 표현된다.

$$s_c^l=[a_m a_{m-1}\cdots a_1 0],\quad s_c^r=[a_m a_{m-1}\cdots a_1 1]$$

이 이진 스트링들을 다음과 같은 복호화(decoding) 함수를 이용해서 $[l,u]$ 범위 내의 실수값으로 변환한다.

$$g([b_m b_{m-1}\cdots b_1])=l+\frac{u-l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+1\right) \tag{1}$$

이 때 부모 스트링과 생성된 자녀 스트링의 변환된 실수값들은 다음과 같은 관계가 항상 성립된다.

$$g([b_m b_{m-1}\cdots b_1 0])<g([b_m b_{m-1}\cdots b_1])<g([b_m b_{m-1}\cdots b_1 1]) \tag{2}$$

증명:

$$g([b_m b_{m-1}\cdots b_1 0])=l+\frac{u-l}{2^{m+2}}\left(\sum_{j=1}^{m}b_j 2^{j+1}+1\right)$$

$$=l+\frac{u-l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+\frac{1}{2}\right)$$

$$=l+\frac{u-l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+1\right)-\frac{1}{2}\frac{u-l}{2^{m+1}}$$

$$=g([b_m b_{m-1}\cdots b_1])-\frac{u-l}{2^{m+2}}$$

$$g([b_m b_{m-1}\cdots b_1 1])=l+\frac{u-l}{2^{m+2}}\left(\sum_{j=1}^{m}b_j 2^{j+1}+2+1\right)$$

$$=l+\frac{u-l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+\frac{3}{2}\right)$$

$$=l+\frac{u-l}{2^{m+1}}\left(\sum_{j=1}^{m}b_j 2^j+1\right)+\frac{1}{2}\frac{u-l}{2^{m+1}}$$

$$=g([b_m b_{m-1}\cdots b_1])+\frac{u-l}{2^{m+2}}$$

그러므로

$$g([b_m b_{m-1}\cdots b_1])-g([b_m b_{m-1}\cdots b_1 0])$$

$$=g([b_m b_{m-1}\cdots b_1])-g([b_m b_{m-1}\cdots b_1 0])=\frac{u-l}{2^{m+2}}, \tag{3}$$

$u>l$ 이면 (2)는 항상 성립된다. ■

DEAS는 일반적인 n차원의 엔지니어링 문제에 적용하기 위해 설계되었으며, 이 경우 다루어야 할 스트링은 n개가 된다. 각 행의 LSB에 대해 0이나 1을 삽입함으로써 발생되는 모든 이웃점의 수는 eDEAS의 경우 2^n가지가 발생한다. 이를 설명하기 위해 3차원 문제에서 각 변수와 스트링이 다음과 같은 변환관계를 갖는다고 하자.

$$x_1\leftrightarrow\left[a_m^{(1)}a_{m-1}^{(1)}\cdots a_1^{(1)}\right],$$

$$x_2\leftrightarrow\left[a_m^{(2)}a_{m-1}^{(2)}\cdots a_1^{(2)}\right],$$

$$x_3\leftrightarrow\left[a_m^{(3)}a_{m-1}^{(3)}\cdots a_1^{(3)}\right].$$

이 스트링들을 다음과 같이 쌓아서 행렬로 만들고 임시 부모 행렬을 $\mathbf{B}_p$라고 하자.

$$\mathbf{X}_p=\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}\leftrightarrow\begin{bmatrix}a_m^{(1)}a_{m-1}^{(1)}\cdots a_1^{(1)}\\a_m^{(2)}a_{m-1}^{(2)}\cdots a_1^{(2)}\\a_m^{(3)}a_{m-1}^{(3)}\cdots a_1^{(3)}\end{bmatrix}=\mathbf{B}_p$$

이 때 BSS에 의해 발생되는 자녀 행렬은 다음과 같이 총 8개가 된다.

$$\begin{bmatrix}0\\\mathbf{B}_p\,0\\0\end{bmatrix},\begin{bmatrix}1\\\mathbf{B}_p\,0\\0\end{bmatrix},\begin{bmatrix}0\\\mathbf{B}_p\,1\\0\end{bmatrix},\begin{bmatrix}1\\\mathbf{B}_p\,1\\0\end{bmatrix},\begin{bmatrix}0\\\mathbf{B}_p\,0\\1\end{bmatrix},\begin{bmatrix}1\\\mathbf{B}_p\,0\\1\end{bmatrix},\begin{bmatrix}0\\\mathbf{B}_p\,1\\1\end{bmatrix},\begin{bmatrix}1\\\mathbf{B}_p\,1\\1\end{bmatrix}. \tag{4}$$

그러므로 n차원 문제의 경우 발생할 수 있는 모든 자녀 행렬의 수는 BSS에서 2^n 개가 된다.

그림 1은 eDEAS에서의 BSS 탐색 원리를 의사코드(pseudo-code)로 표현한 것이다. $\mathbf{B}_{n\times m}^{*}$ 는 바로 직전 session에서 얻은 최적의 이진 행렬을 나타내며, BSS는 이 행렬에 방향벡터 $\mathbf{d}_{n\times 1}$ 를 가장 오른쪽 열에 삽입함으로써 2^n개의 이웃행렬

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

Initialization:

Set a direction vector as a zero column;

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

Set $J_{\min} \leftarrow M >> 1$, and $i \leftarrow 1$.

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

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

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

Decode the temporary matrix into a real vector as

$\mathbf{C}_{n\times(m+1)} \xrightarrow{g} \mathbf{X}_{n\times 1}$.

Evaluate the cost;

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

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

Save the current best optima;

$\mathbf{B}^*_{n\times(m+1)} \leftarrow \mathbf{C}_{n\times(m+1)}$, $\mathbf{d}_{opt} \leftarrow \mathbf{d}$,

$\mathbf{X}_{\min} \leftarrow \mathbf{X}$, $J_{\min} \leftarrow J$.

end if

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

$i \leftarrow i + 1$

end while

그림 1. BSS의 의사코드.

Fig. 1. Pseudocode of BSS.

$\mathbf{C}_{n\times(m+1)}$ 을 생성한다. 그리고 이 행렬의 각 행을 (1)에 의해서 복호화 함으로써 실수 벡터 $\mathbf{X}$를 얻고, 비용함수 $f(\mathbf{X})$를 계산한다. 계산된 비용함수 중 최소 값에 해당되는 이진 행렬 $\mathbf{B}^*_{n\times(m+1)}$, 방향 벡터 d_{opt}, 비용함수 값 $J_{\min}$ 을 UDS에게 넘겨준다.

DEAS의 BSS는 현재의 탐색점을 기준으로 각기 상반되는 방향으로 이웃점들을 생성하고, 모든 이웃점 중에서 최소의 비용값을 가진 점을 새로운 지역해로 선택한다. 그러나, 일반적인 직접탐색법인 univariate method[10]나 generating set search (GSS)[11]에서는 이웃점들의 비용값 중 최소값이 현 지역해보다 감소한 경우에만 성공적인(successful) 탐색으로 간주하고, 그렇지 않은 경우 실패한(unsuccessful) 탐색으로 간주한다. 실패한 탐색으로 판명된 경우 univariate method는 탐색을 중단하고, GSS는 step length를 반감하고 다시 탐색한다.

이러한 BSS의 상대적인 선택은 이진 스트링 길이를 강제적으로 늘임으로써 해공간의 분해도를 순차적으로 높이기 위해 설계되었으며, 비용함수가 부드럽지 않은 경우에도 DEAS가 강인한 탐색성능을 가지게 한다. 그림 2는 비용함수에 잡음이 있음에도 불구하고 BSS가 적절한 방향으로 이웃점을 선택할 수 있음을 예시한다. 현재의 탐색점을 부모점 p라 하고 생성된 이웃점을 자식점 s_l, s_r 이라고 한다. 비용함수에 잡음이 없는 경우 그림 2(a)에서 보는 것처럼 univariate method, GSS, DEAS 모두 s_l 의 비용값이 p의 비용값보다 더 작으므로 s_l 을 지역해로 갱신한다. 하지만 2(b)와 같이 비용함수에 잡음이 포함되어 있는 경우 $f(p) < f(s_l) < f(s_r)$ 의 관계가 되어 DEAS만 s_l 을 지역해로 갱신한다. 그림 2(b)에

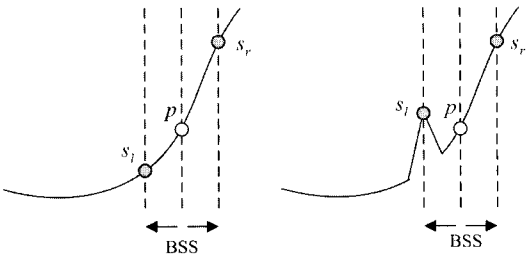


그림 2. 1차원에서의 BSS시 생성되는 이웃점.

Fig. 2. Neighborhood points generated at BSS in one dimension.

Unidirectional Search ($\mathbf{B}^*_{n\times(m+1)}, \mathbf{d}_{opt}, J_{\min}$)

Initialization: Load $\mathbf{B}^*_{n\times(m+1)}, \mathbf{d}_{opt}, J_{\min}$ of BSS

while A better solution is attained **do**

do LIMIT CHECK

Eliminate h limit rows([0 ⋯ 0] or [1 ⋯ 1]) from current UDS with an index $l = n - h$.

Enroll the indices of unlimited strings on $\mathbf{z}_{l\times 1}$.

Set an extension vector; $\mathbf{e}_{l\times 1} = [0\ 0 \cdots 0\ 1]^T$.

for $i = 1 : 2^l - 1$

do REDUNDANCY CHECK

if $\mathbf{e}$ is a redundant search direction with $\mathbf{e}_{opt}$ **then** CONTINUE

Load the current best matrix into a temporary matrix; $\mathbf{U}_{n\times(m+1)} \leftarrow \mathbf{B}^*_{n\times(m+1)}$

for $j = 1 : l$

$p = \mathbf{z}(j)$.

Select the p-th row of $\mathbf{U}$;

$\mathbf{r}_{1\times(m+1)} = \mathbf{U}(p, 1 : m+1)$.

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

$\mathbf{U}(p, 1 : m+1) = \mathbf{r}'_{1\times(m+1)}$.

end for

Decode the extended matrix into a real vector and evaluate its cost function;

$\mathbf{U}_{n\times(m+1)} \xrightarrow{g} \mathbf{X}_{n\times 1}$, $J = f(\mathbf{X})$.

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

Save the current best optima;

$\mathbf{B}^*_{n\times(m+1)} \leftarrow \mathbf{U}_{n\times(m+1)}$, $\mathbf{e}_{opt} \leftarrow \mathbf{e}$,

$\mathbf{X}_{\min} \leftarrow \mathbf{X}$, $J_{\min} \leftarrow J$.

end if

$\mathbf{e} \leftarrow \mathbf{e} + 1$.

end for

end while

그림 3. UDS의 의사코드.

Fig. 3. Pseudocode of UDS.

서처럼 s_1 에서 단순한 잡음이 발생한 경우 뒤따르는 UDS에 의해 다시 비용값이 감소하지만, p점이 지역해라면 UDS를 통해서도 더 이상 비용값이 감소하지 않는다. 그러므로 DEAS가 잡음에 강인함을 알 수 있다.

1.2 UDS

UDS는 BSS에서 찾은 지역해의 방향을 따라 확장된 탐색을 수행한다. UDS가 BSS와 다른 점은 2가지인데, BSS와 달리 UDS는 스트링의 길이를 1 증가시키지 않고 그대로 유지한다는 점이며, BSS는 session당 한 번만 수행되지만 UDS는 비용값이 감소될 때까지 계속해서 수행된다는 점이다.

그림 3은 eDEAS의 UDS를 의사코드로 표현한 것으로서, 직전의 BSS에서 구해진 $\mathbf{B}^*_{n\times(m+1)}$, $\mathbf{d}_{opt}$ 와 $J_{\min}$ 을 이용해서 탐색을 시작함을 보인다. UDS에는 두 가지의 check 루틴이 있는데 limit check와 redundancy check이다. 먼저 limit check은 $\mathbf{B}^*_{n\times(m+1)}$ 에서 행 전체가 모두 0이거나 모두 1인 행(extreme rows)을 찾아서 UDS에서 제외시킨다. 그리고 남은 l 개의 행에 대해서 BSS와 마찬가지로 2^l 개의이웃행렬을 생성한다. 의사코드에서 확장벡터(extension vector) $\mathbf{e}_{l\times 1}$ 는 0과 1로 구성된 이진벡터로서 1인 경우에만 $\mathbf{d}_{opt}$ 의 방향으로 확장 탐색을 수행한다. 예를 들어 2차원 문제에서 extreme row가 없을 경우 확장벡터는 $3(=2^2-1)$ 개가 생성된다. 만약 $\mathbf{d}_{opt}=[0\quad 1]^T$ 이면 확장 벡터들이 생성하는 UDS 탐색방향은 표 1과 같다. 즉, $\mathbf{d}_{opt}=[0\quad 1]^T$ 이고 $\mathbf{e}_1=[0\quad 1]^T$ 인 경우, $\mathbf{e}_1(2)=1$ 이므로 x_2축에서 $\mathbf{d}_{opt}(2)$ 방향(1이므로 increment addition)으로 UDS를 한다.

$\mathbf{e}_2$와 $\mathbf{e}_3$도 같은 방법으로 UDS의 이웃점을 생성하며, 이를 도식화한 것이 그림 4이다. 그림에서 점선으로 표현된 것은 현재의 이진행렬의 행 길이(row length)에 의해서 결정되는 격자이며, 행길이가 길수록 격자 간의 폭이 좁아지게 된다.

UDS 코드의 두번째 check 루틴인 redundancy check은 UDS가 2회 이상 연속적으로 수행될 때 발생되는 반복탐색점(revisited search point)을 탐색에서 제외하게 하는루틴이다. 그림 5는 그림 4에서 $\mathbf{e}_1^{(1)}$ 이 최적방향($\mathbf{e}_{opt}^{(1)}$)이 되고 두번째 UDS가 실행되었을 경우를 예시한 것이다.

두번째 UDS에서 $\mathbf{e}_2^{(2)}$ 의 방향으로 탐색할 경우 직전 UDS에서 $\mathbf{e}_3^{(1)}$ 에 의해 탐색된 이웃점을 다시 만나게 된다. 그러므로 비용함수를 다시 계산하는 것을 방지하기 위해 $\mathbf{e}_2^{(2)}$ 방향은 UDS에서 제외되어야 하는데 간단한 masking 기법으로써 쉽게 구현될 수 있다[8].

2. 전역탐색 전략

DEAS는 전역탐색 전략으로서 multistart 기법을 채용한다. multistart 기법이란 임의의 시작점으로부터 지역탐색을 시작해서 지역해를 찾은 후, 또 다른 임의의 시작점으로부터 지역탐색을 수행하는 전역탐색 기법을 의미한다. 이렇게 지역탐색을 주어진 횟수만큼 수행한 후 그 중에서 가장 좋은 지역해를 전역해로 간주하는 탐색기법이다.

표 1. 확장벡터와 UDS 탐색방향 연관성.

Table 1. Relation of extension vectors and UDS directions.

	$\mathbf{e}_1$	$\mathbf{e}_2$	$\mathbf{e}_3$
확장벡터	$\begin{bmatrix}0\\1\end{bmatrix}$	$\begin{bmatrix}1\\0\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$
UDS 방향 벡터	$\begin{bmatrix}-\\\mathbf{d}_{opt}(2)\end{bmatrix}$	$\begin{bmatrix}\mathbf{d}_{opt}(1)\\-\end{bmatrix}$	$\begin{bmatrix}\mathbf{d}_{opt}(1)\\\mathbf{d}_{opt}(2)\end{bmatrix}$
해공간에서의 탐색방향	$\begin{bmatrix}-\\\text{INC}\end{bmatrix}$	$\begin{bmatrix}\text{DEC}\\-\end{bmatrix}$	$\begin{bmatrix}\text{DEC}\\\text{INC}\end{bmatrix}$

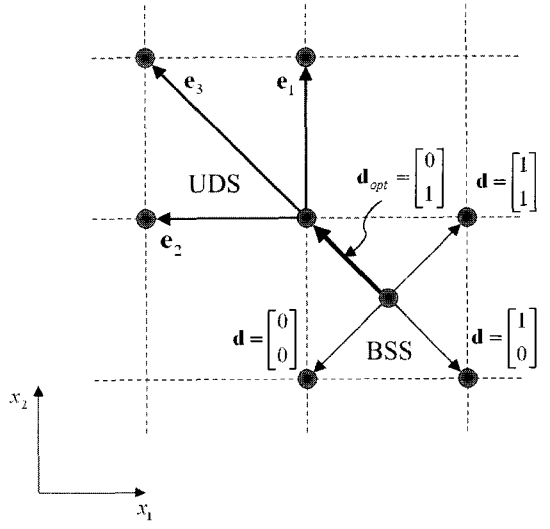


그림 4. 2차원에서의 BSS와 UDS 탐색 예.

Fig. 4. Search example of BSS and UDS in two-dimensional space.

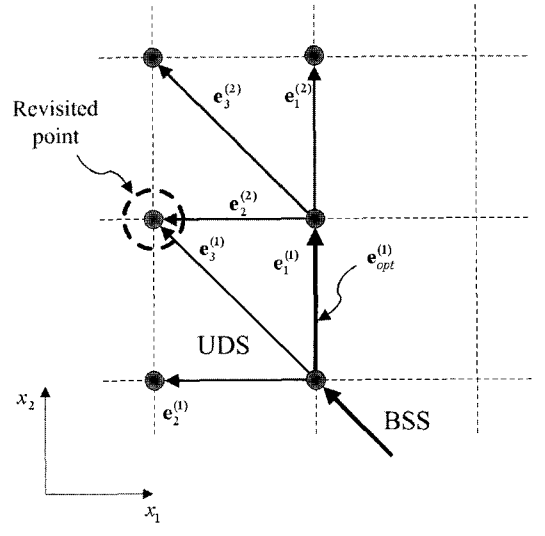


그림 5. 연속적인 UDS에서 발생하는 반복점의 존재.

Fig. 5. Redundant point existing in continuous UDS.

본 논문은 지역탐색 전략을 통해 비용함수의 매끄러움도를 측정하는 것에 주안점을 두고 있기 때문에 전역탐색 전략에 대한 설명은 생략하기로 한다. 전역탐색에 관한 자세한 내용은 [8]에 자세히 소개되어 있다.

III. 비용함수의 형상 정보

진화연산(evolutionary computation)의 경우 적합도 지형(fitness landscape)은 부드러움도(smoothness), 거칠기(ruggedness), 그리고 중성도(neutrality)로 불리는 세 가지 지형 특성에 의해 완전히 설명된다[12]. 어떤 정점(vertex) $v \in V$ 는 v의 적합도 값이 모든 이웃점 $w \in N(v)$ 의 적합도 값보다 큰 경우 지역 최대값(local maximum)이라고 한다[13]. 적합도 지형의 거칠기는 지역 최대값의 개수에 의해 결정되므로 지역해가 많을수록 적합도 지형이 거친(rugged) 형태라고 할 수 있다. 한편 적합도 지형의 부드러움도는 basin of attraction(BOA)의 면적에 비례하므로, BOA가 넓을수록 부드러운(smooth) 지형을 가졌다고 할 수 있다. Jones는 최대화 문제에 대해 BOA를 다음과 같이 정의했다[14].

$$B(v_n) = \{v_0 \in V \mid \exists v_1, \cdots, v_n \in V \text{ with } v_{i+1} \in N(v_i) \\ \text{and } f(v_{i+1}) > f(v_i) \text{ for each } i, 0 \le i < n\} \quad (5)$$

마지막으로 중성도는 평평한 지형을 가리키는 용어이다.

적합도 지형의 특성을 측정하기 위해 Weinberger는 간단한 random walk로부터 구한 시계열 적합도 값에 대해 autocorrelation을 계산했다[15]. 이 방법은 random walk의 특성에 적합하고, 계산량이 많이 소요되는 단점이 있다. 한편, Zinchenko 등은 BOA에서 basin의 깊이와 탐색 난이도를 이용해서 적합도 지형의 특성을 측정하였다[16]. 이 방법은 진화연산을 수행하기 전에, 비용함수 중 벌칙함수(penalty function)를 구성하는 계수 별로 개략적으로 비용함수의 거칠기 정도를 계산하기 때문에 탐색 도중에 사용할 수 없다.

DEAS는 진화연산과는 달리 탐색 경로가 계획적이고 일정한 규칙을 따른다. 그러므로 어떤 비용함수에 대해 BSS와 UDS의 거동을 분석하면 그 함수의 형상에 대한 특성을 간접적으로 알 수 있다. 그림 6은 세가지 대표적인 함수 형상을 나타낸 것으로 부드럽거나 거친 경우와 단봉(unimodal)이거나 다봉(multimodal)인 경우의 조합을 보였다. 본 논문에서는 분석의 용이성을 위해 그림 6(b)의 거친 단봉함수를 대상으로 분석을 시행하고 제안된 거칠기 지수를 측정한다.

그림 7은 DEAS의 지역탐색 시 구축되는 이진 트리 구조를 기반으로 지형이 거친 비용함수에 대해 두 번의 session을 수행한 것을 보인다. 탐색 양상을 각 session의 BSS와 UDS를 기준으로 설명하면 다음과 같다.

i) 최상위인 A 스트링에서 BSS를 수행함으로써 오른쪽 방향이 $\mathbf{d}_{opt}$로 판정된다.
ii) 이 방향으로 UDS가 수행되지만 비용함수를 더 감소시키지 못해 B 스트링에서 탐색을 멈춘다.
iii) B 스트링에서 다시 BSS가 수행되었지만 거친 지형에 의해 이번에는 반대 방향인 왼쪽 방향이 $\mathbf{d}_{opt}$로 선택된다.
iv) C 스트링에서 왼쪽 방향으로 UDS를 수행했을 때 첫번째 UDS에서 비용함수가 감소했고, 더 이상 감소하지 않아 D스트링에서 탐색을 멈춘다.
v) D 스트링에서 다시 BSS를 시작한다.

DEAS에서 BSS만 존재한다면 Branch and Bound Method (BBM)와 유사하다. BBM은 전체 해공간을 겹치지 않는 부분집합(subset)으로 계속 분할하며 탐색을 수행해서 upper bound와 lower bound의 간격이 정해진 tolerance 이하가 되면 탐색을 멈춘다. 그림 7에서 회색 사각형은 BBM에 의하면 A 스트링에서 분할된 후 다시 탐색되지 않는 부분집합을 의미한다. DEAS의 BSS는 BBM의 원리와 유사하고, UDS는 BSS가 도달할 수 없는 탐색영역을 확장시키기 위해 첨가되었음을 고려할 때 부드러운 단봉(unimodal) 비용함수라면 상위 스트링에서 선택되지 않은 회색 사각형 영역으로 다시 탐색을 수행하지 않아야 한다.

그러나 표면이 거친 비용함수의 경우 BSS에서 잘못 된 탐색 방향을 선택할 수 있다. 어떤 스트링에서 그림 7과 같이 탐색이 진행되었다면 이는 스트링 A에서 BSS에 오류가 있음을 의미한다. 즉, 스트링 A에서 왼쪽 방향을 $\mathbf{d}_{opt}$로 결정하는 것이 옳지만 잡음에 의해 오른쪽 방향이 선택되었고, 그로 인해 하위 레벨의 스트링에서 반대방향의 BSS와 2번 이상의 UDS가 수행된 것이다. 그러므로 탐색 중에 이러한 불규칙적인 탐색 패턴이 존재한다면 그 비용함수에는 잡음이 존재한다고 판단할 수 있다.

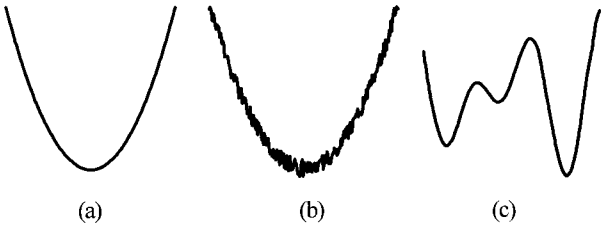


그림 6. 다양한 함수의 형상들: (a)부드러운 단봉함수, (b)거친 단봉함수, (c)부드러운 다봉함수.
Fig. 6. Various function shapes: (a)smooth unimodal function, (b)rugged unimodal function, (c)smooth multimodal function.

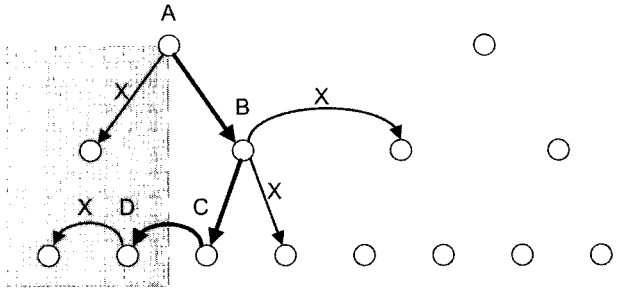


그림 7. 표면이 거친 비용 함수에 대한 탐색 예 -1.
Fig. 7. Search example 1 of DEAS for rugged cost function.

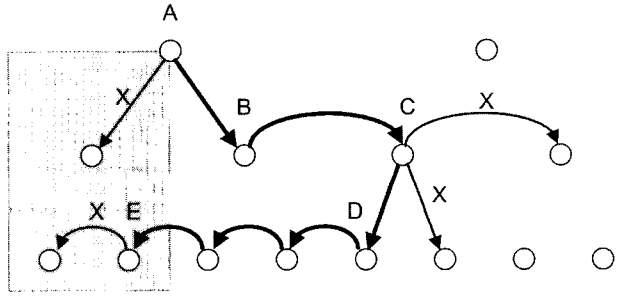


그림 8. 표면이 거친 비용 함수에 대한 탐색 예 -2.
Fig. 8. Search example 2 of DEAS for rugged cost function.

그림 8은 그림 7보다 좀 더 드문 경우로서, 스트링 A에서 BSS가 잘못 수행되었음에도 불구하고 1회 이상의 성공적인 UDS가 존재하고, 그 다음 레벨의 스트링에서 이를 만회하기 위해 적어도 4회 이상의 UDS가 수행되는 것을 보여준다. 이 경우에도 중간(스트링 B) 레벨에서 잡음이 존재한다고 할 수 있다.

전술한 현상을 일반화하면 다음과 같은 명제(propostion)로 표현할 수있다.

명제 1: 현재 레벨의 스트링에서 p번의 UDS가 수행되고 하위 레벨의 스트링에서 상위 레벨과는 반대 방향의 BSS와 $2p$번 이상의 UDS가 수행되면 현재 레벨의 비용함수에 잡음이 존재한다.

이 명제에서 중요한 것은 각 스트링은 행의 길이에 따라 이웃 스트링과의 간격이 달라진다는 사실이다. DEAS의 초기 탐색에서는 스트링의 길이가 짧아서 BSS와 UDS의 step length가 긴 편인데, 이 경우 그림 7, 8과 같이 불규칙적인 탐색 양상이 발견된다면 비용함수의 형상은 그림 6(c)의 다봉함수에 가까울 것이다. 반면에 스트링이 길어지면 step length가 짧아지므로, 그림 7, 8과 같은 탐색양상이 발생하면 비용함수는 그림 6(b)처럼 잡음을 포함하고 있다고 할 수 있다. 그러므로 제안된 명제를 이용하면 부드러움도와 거칠기를 동시에 파악할 수 있음을 알 수 있다.

본 논문에서는 DEAS 지역탐색 시 명제 1의 발생횟수를 새로운 형상정보인 거칠기 지수로 정의한다. 이를 위해 단봉이면서 아래로 볼록한(convex) 함수인 2차 함수에 다양한 크기의 잡음을 섞은 것을 비용함수로 선정하고 DEAS를 수행함으로써, 제안된 거칠기 지수의 신뢰성을 검증하도록 한다.

IV. 실험 결과

본 논문에서 테스트 함수로 선정한 비용함수의 수식은 다음과 같다.

$$y=(x-c)^2+bN,\quad x\in[-1,1], \tag{6}$$

식에서 c는 중점, b는 잡음의 이득, N은 uniformly distributed pseudo-random number를 각각 의미한다. 2차 함수는 중점에 대해 대칭이므로 초기값은 이진 스트링 0으로 고정하고 b와 c를 다음과 같이 변화시키며 DEAS를 수행했다.

- b: 0, 0.05, 0.1, 0.4
- c: -0.9, -0.6, -0.3, 0, 0.3, 0.6, 0.9.

DEAS는 모든 경우에 대해 1비트 스트링인 0을 초기 스트링으로 고정하고 탐색을 시작해서 스트링 길이가 10이 될 때까지 수행했다.

표 2는 $b=0$, 즉 비용함수에 잡음이 전혀 없는 경우에 DEAS의 탐색 양상을 보인다. 전체적으로 스트링 레벨간의 탐색 거동의 특이성이 발견되지 않음과 지역최적해인 c를 정확하게 찾아냄을 알 수 있다. 이 때 발생된 오차는 스트링 길이가 10으로 한정되었음에 기인한다. 표에서 'x'는 UDS에서 limit string이 발생했음을 나타내며, 주목할 점은 c가 0보다 클 경우(지역해가 초기 스트링으로부터 멀어질 경우) UDS가 최초 레벨에서만 2회가 되고 그 다음 레벨에서는 1회 이내가 된다. 즉, 부드러운 단봉함수의 경우 초기 스트링의 UDS가

표 2. b=0일 때의 DEAS의 탐색 양상.
Table 2. Search aspects of DEAS when b=0.

c		-0.9		-0.6		-0.3		0		0.3		0.6		0.9	
탐색 방법		BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS
row length	2	L	x	L	x	R	1	R	1	R	2	R	2	R	2
	3	L	x	R	1	L	1	R	1	R	1	L	1	R	x
	4	L	x	R	1	R	1	R	1	L	1	L	1	R	x
	5	R	1	L	1	R	1	R	1	L	1	R	1	L	1
	6	R	1	L	1	L	1	R	1	R	1	R	1	L	1
	7	L	1	R	1	L	1	R	1	R	1	L	1	R	1
	8	L	1	R	1	R	1	R	1	L	1	L	1	R	1
	9	R	1	L	1	R	1	R	1	L	1	R	1	L	1
	10	R	1	L	1	L	1	R	1	R	1	R	1	L	1
		-0.9004		-0.5996		-0.2998		-0.0010		0.2998		0.5996		0.9004	

표 3. b=0.05일 때의 DEAS의 탐색 양상.
Table 3. Search aspects of DEAS when b=0.05.

c		-0.9		-0.6		-0.3		0		0.3		0.6		0.9	
탐색 방법		BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS
row length	2	L	x	L	x	R	1	R	1	R	2	R	2	R	2
	3	L	x	R	1	L	1	R	1	R	1	L	1	R	x
	4	L	x	R	1	R	1	R	2	L	1	L	1	R	x
	5	L	x	L	1	R	2	L	1	L	2	R	1	R	x
	6	R	1	L	2	L	1	L	2	L	2	R	1	L	1
	7	L	1	R	1	R	1	R	1	R	1	R	1	L	**3**
	8	R	1	R	**3**	L	**3**	L	1	R	1	R	2	L	1
	9	L	**3**	L	2	R	2	R	1	L	1	R	1	R	1
	10	R	2	R	2	R	2	R	1	R	1	L	2	R	2
		-0.9668		-0.6191		-0.2373		-0.0078		0.2188		0.6172		0.9121	

표 4. b=0.1일 때의 DEAS의 탐색 양상.
Table 4. Search aspects of DEAS when b=0.1.

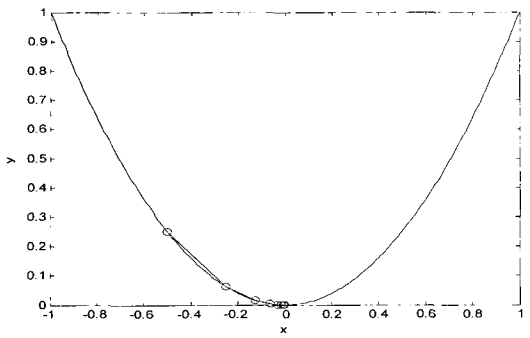
c		-0.9		-0.6		-0.3		0		0.3		0.6		0.9	
탐색 방법		BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS
row length	2	L	x	L	x	R	1	R	1	R	2	R	2	R	2
	3	L	x	R	1	L	1	R	1	R	1	L	1	R	x
	4	R	1	R	1	R	**2**	R	2	L	1	L	1	R	x
	5	R	1	R	2	L	2	L	2	L	1	L	1	L	**2**
	6	L	**2**	R	2	L	1	L	1	L	2	L	2	R	2
	7	L	1	L	1	L	**3**	R	1	R	2	R	1	L	1
	8	R	1	L	1	R	1	R	1	R	2	R	2	L	1
	9	L	1	R	1	R	1	R	1	L	1	L	2	R	1
	10	R	1	L	1	L	1	L	**4**	L	1	R	1	R	2
		-0.8330		-0.4219		-0.3359		-0.0400		0.2666		0.625		0.8906	

표 5. b=0.4일 때의 DEAS의 탐색 양상.
Table 5. Search aspects of DEAS when b=0.4.

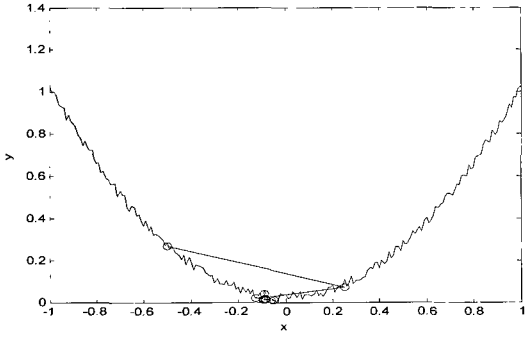
c		-0.9		-0.6		-0.3		0		0.3		0.6		0.9	
탐색 방법		BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS	BSS	UDS
row length	2	L	x	L	x	L	x	R	2	R	2	R	2	R	2
	3	L	x	L	x	R	**3**	L	1	R	1	L	1	L	1
	4	L	x	R	**2**	R	1	L	1	R	1	R	1	R	1
	5	L	x	R	1	L	**2**	L	1	R	2	R	1	L	**3**
	6	L	x	L	1	R	2	L	1	L	1	R	2	L	1
	7	L	x	L	1	R	2	L	2	L	1	R	1	R	1
	8	R	**3**	R	1	L	2	L	1	L	2	R	2	R	2
	9	R	1	R	1	R	**3**	L	1	L	1	L	1	L	2
	10	L	**2**	R	2	R	1	R	1	L	2	R	1	L	1
		-0.8750		-0.6709		-0.0869		-0.0078		0.4375		0.6875		0.6875	

표 6. 잡음 크기와 거칠기 지수와의 비교.
Table 6. Relation of noise magnitude and the proposed ruggedness measure.

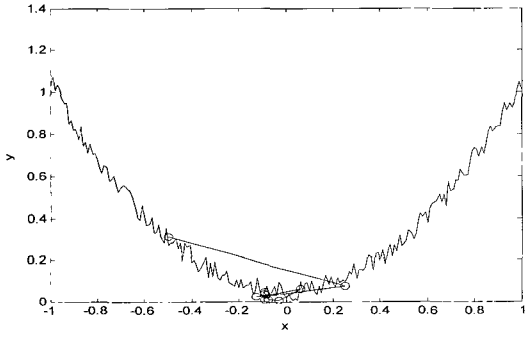
b	0	0.05	0.1	0.4
ruggedness	0	3	4	6



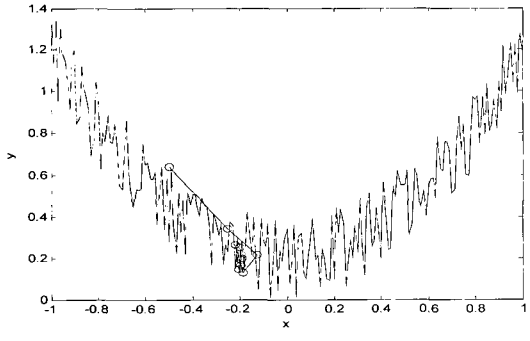
(a) $b=0$



(b) $b=0.05$



(c) $b=0.1$



(d) $b=0.4$

그림 9. 잡음의 크기 변화에 대한 DEAS 탐색 성능 비교.

Fig. 9. Comparison of DEAS search performance under various noise.

지배적으로 탐색을 수행함을 알 수 있다.

표 3-5는 b = 0.05, 0.1, 0.4로 증가할 때 DEAS의 탐색 양상을 보인다. b가 커질수록 인가된 잡음의 크기도 커지기 때문에 명제 1의 불규칙적 탐색 횟수(표에서 색칠한 부분)도 비례해서 증가함을 알 수 있다.

표 6은 표 2-5의 결과를 정리한 것으로서 잡음의 크기와 본 논문에서 제안된 거칠기 지수(ruggedness measure)가 대체적으로 증가함을 보인다.

표 3-5에서 잡음의 크기와 DEAS가 찾아낸 지역해의 정확도와는 대략적으로 반비례하지만 c가 -0.6, 0, 0.3인 경우는 b가 증가한다고 해서 탐색된 지역해가 반드시 알려진 참값으로부터 멀어지는 것은 아님을 알 수 있다. 이것은 임의성을 가진 잡음의 특성에 기인한 것이며, b가 크다고 해서 DEAS의 모든 탐색점에서의 거칠기(ruggedness)도 동일하게 증가하는 것이 아님을 의미한다. 그러므로 최적화 결과로써 얻어진 지역해의 quality는 비용함수의 거칠기 지수로 선정되기에는 곤란하다고 할 수 있다.

결론적으로 위의 최적화 결과로부터 본 논문에서 제안한 거칠기 지수가 비용함수의 형상정보 중 잡음 성분에 관한 유용한 정보를 제시함을 알 수 있다. 이 거칠기 지수를 DEAS에 이용할 경우, 잡음 성분의 크기가 큰 비용함수에 대해서는 이진 스트링의 길이를 비교적 짧게 제한함으로써 탐색효율을 향상시킬 수 있다.

그림 9는 표 2-5의 잡음의 크기를 비교하고, 잡음 존재시 DEAS의 탐색 성능에 어떤 변화가 있는지를 도시한 것이다. 그림에서 보면 b가 0.1 이하인 경우 지역해의 근사값을 잘 찾아낼 수 있음을 알 수 있으며, 이는 DEAS가 잡음에 강한 강인한(robust) 최적화 기법임을 증명한다.

VI. 결론

본 논문에서는 최근에 저자에 의해 개발된 DEAS의 탐색 원리에 대해 설명하고, 1차원 문제에 대해 비용함수의 형상정보를 추출하는 방법을 제안했다. 제안된 거칠기 지수는 최적화 실험결과 비용함수에 섞인 잡음 성분의 크기에 비례해서 증가하므로 비용함수의 형상정보로서 적절하다고 할 수 있다. 그리고 얻어진 형상정보는 향후 DEAS의 탐색 성능을 적응적으로 강화시키는데 사용될 수 있다. 그리고, 일반화를 위해 다차원 문제에서 형상정보를 효율적으로 추출하는 방법에 대한 연구가 이루어질 계획이다.

참고문헌

[1] J. -W. Kim and S. W. Kim, "A novel parametric identification method using a dynamic encoding algorithm for searches (DEAS)," *International Conference on Control, Automation and Systems*, pp. 406-411, Muju, Korea, October 2002.

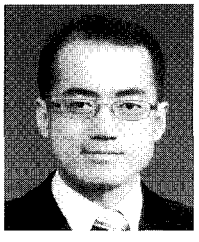
[2] J. -W. Kim and S. W. Kim, "Gain tuning of PID controllers with the dynamic encoding algorithm for searches (DEAS) based on the constrained optimization technique," *International Conference on Control, Automation and Systems*, pp. 871-876, Gyeongju, Korea, October 2003.

[3] Y. J. Jang and S. W. Kim, "An Estimation of a Billet Temperature during Reheating Furnace Operation", *International Journal of Control, Automation, and Systems*, vol. 5, no. 1, pp. 43-50, Feb. 2007

[4] J. -W. Kim and S. W. Kim, "Parameter identification of induction motors using dynamic encoding algorithm for searches (DEAS)," *IEEE Trans. Energy Conversion*, vol. 20, no. 1, pp. 16-24, March 2005.

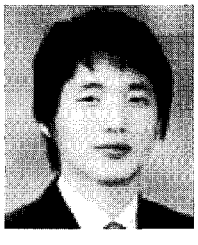
[5] Y. Park, Y. Lee, J. -W. Kim, and S. W. Kim, "Parameter

optimization for SVM using dynamic encoding algorithm," *International Conference on Control, Automation, and Systems*, pp. 2542-2547, KINTEX, Korea, June, 2005.
[6] 김태규, 김종욱, "DEAS를 이용한 변압기 코아의 최적설계," 대한전기학회 논문지, 56권, 6호, pp. 1055-1063, June 2007.
[7] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Lerning*, Addison Wesley, 1989.
[8] J. -W. Kim and S. W. Kim, "Numerical method for global optimization: dynamic encoding algorithm for searches (DEAS)," *IEE Proc.-Control Theory and Appl.*, vol. 151, no. 5, pp. 661-668. Sept. 2004.
[9] J. -W. Kim, N. G. Kim, and S. W. Kim, "On-load parameter identification of an induction motor using univariate dynamic encoding algorithm for searches", *International Conference on Control, Automation and Systems*, pp. 852-856, Bangkok, Thailand, August 2004.
[10] S. S. Rao, *Engineering Optimization*, John Wiley & Sons Inc., 1996.
[11] T. G. Kolda, R. M. Lewis, and V. Torczon, "Optimization by direct search: new perspectives on some classical and modern methods," *SIAM Review*, vol. 45, no. 3, pp. 385-482, 2003.
[12] A. Ghosh, and S. Tsutsui (Eds.), *Advances in Evolutionary Computing: Theory and Application*, Springer-Verlag, 2003.
[13] R. Palmer, *Optimization on rugged landscape,*. vol. IX of SFI Studies in the Science of Complexity, Addison-Wesley, MA, pp. 3-25, 1991.
[14] T. Jones, *Evolutionary Algorithms, Fitness Landscape and Search*, PhD thesis, University of New Mexico, Albuquergue, NM, 1995.
[15] E. D. Weinberger, "Correlated and uncorrelated fitness landscape and how to tell the difference," *Biological Cybernetics*, vol. 63, pp. 325-336, 1990.
[16] L. Zinchenko and S. Sorokin, "Fitness estimations for evolutionary antenna design," NASA/DoD Conference on Evolvable Hardware, pp. 155-164, July 2003.



김 종 욱

1970년 10월 24일생. 1998년 포항공과대학교 전기전자공학과(공학사). 2000년 동 대학원 전기전자공학과(공학석사). 2004년 동 대학원 전기전자공학과(공학박사). 2004년~2006년 포스코 기술연구소 전기강판연구그룹 연구원2. 2006년~현재 동아대학교 전자공학과 전임강사. 2007년~현재 Journal of Mathematical Control Science and Applications 편집위원. 관심분야는 최적화 기법, 파라미터 식별, 지능제어, 임베디드 시스템, 고장 진단.

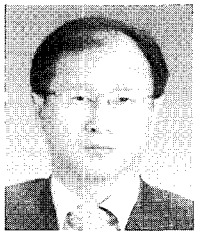


김 태 규

2007년 동아대 전자공학과 졸업. 2007년~현재 동 대학원 전자공학과 석사과정 재학중. 관심분야는 임베디드 시스템, 로봇 제어.

박 영 수

2004년 포항공과대학교 전자전기공학과(공학사). 2006년 동 대학원 전자전기공학과(공학석사). 2006년~현재 동 대학원 전자전기공학과 박사과정 재학중. 관심연구 분야는 무인차량, 항만 자동화, 최적화 알고리즘, 임베디드 시스템.



김 상 우

1962년 8월 14일생. 1983년 서울대학교 제어계측공학과(공학사). 1985년 동 대학원 제어계측공학과(공학석사). 1990년 동 대학원 제어계측공학과(공학박사). 1991년~현재 포항공과대학교 전자전기공학과 부교수. 관심분야는 최적제어, 최적화 알고리즘, 지능제어, 무선통신, 공정 자동화.