한국 퍼지 및 지능시스템 학회 2002, Vol. 12, No. 5, pp. 451-461

적응성 있는 차분 진화에 의한 함수최적화와 이벤트 클러스터링

Function Optimization and Event Clustering by Adaptive Differential Evolution

황희수

Hwang Heesoo

한라대학교 전기전자제어공학부

요 약

차분 진화는 다양한 형태의 목적함수를 최적화 하는데 매우 효율적인 방법임이 입증되었다. 차분 진화의 가장 큰 이점은 개념적 단순성과 사용의 용이성이다. 그러나 차분 진화의 수렴성이 제어 파라미터에 매우 민감한 단점이 있다. 본 논문은 새로운 교배용 벡터 생성법과 제어 파라미터의 적응 메커니즘을 결합한 적응성 있는 차분 진화를 제안한다. 이는 수렴성을 해치지 않으면서 차분 진화를 보다 강인하게 만들며 사용이 쉽도록 해준다. 12가지 최적화 문제에 대해 제안한 방법을 시험하였다. 적응성 있는 차분 진화의 응용 사례로써 이벤트 예측을 위한 교사 클러스터링 방법을 제안한다. 이 방법을 진화에 의한 이벤트 클러스터링이라 부르며 데이터 모델링 검증에 널리 사용되는 4 가지 사례에 대해 그 성능을 시험하였다.

ABSTRACT

Differential evolution(DE) has been proved to be an efficient method for optimizing real-valued multi-modal objective functions. DE's main assets are its conceptual simplicity and ease of use. However, the convergence properties are deeply dependent on the control parameters of DE. This paper proposes an adaptive differential evolution(ADE) method which combines with a variant of DE and an adaptive mechanism of the control parameters. ADE contributes to the robustness and the easy use of the DE without deteriorating the convergence. 12 optimization problems is considered to test ADE. As an application of ADE the paper presents a supervised clustering method for predicting events, what is called, an evolutionary event clustering(EEC). EEC is tested for 4 cases used widely for the validation of data modeling.

Key Words : 차분진화, 함수최적화, 이벤트 클러스터링, 퍼지 목적프로그래밍(differential evolution, function optimization, event clustering, fuzzy goal programming)

1. 서 론

차분 진화는 집단에 속한 개체 벡터의 거리와 방향 정보를 사용하는 알고리즘으로 ICEC96 경연대회에서 진화알고리즘 가운데 10개의 시험 문제를 가장 빨리 해결하였다[1-3]. 이는 차분 진화가 그 구조와 연산의 단순성에도 불구하고 적은 집단 크기로 최적 해를 찾는 효율적인 방법임을 입증하는 것이다. 그러나 탐색결과가 알고리즘 제어 파리미터에 매우 민감한 단점이 있다. 제어 파라미터의 민감성 문제는 간단하게 정리될 수 없는데 이는 문제에 따라 파라미터의 크기에 편차가 많기 때문이다. 유전알고리즘과 같은 경우에는 많은 연구가 있어서 제어 파라미터의 유효 범위가 대략 설정되어 있지만 차분 진화의 경우에는 그렇지 못하다. 따라서 진화과정에서 문제에 따라 제어 파라미터를 적응시킬 메커니즘이 절실하게 요구된다. 본 논문에서는 새로운 교배용 벡터 생성 및 제어 파라미터를 적응시킬 수 있는 차분 진화(ADE: Adaptive Differential Evolution)를 제안한다. 이는 차분 진화를 적용할 때 문제에 따라 적당한 제어 파라미터 값을 선정해야 하는 고민을 없애주기 때문에 그 만큼 차분 진화를 강인하고 사용하기 편리하게 해준다. 적응 알고리즘은 적응 대상 파라미터 집합을 구성하고 그 가운데 진화에 기여도가 많은 파라미터 값이 보다 빈번하게 사용되도록 한다. 파라미터 값은 기본적으로 룰렛-휠과 같은 선택 메커니즘에 의해 결정되지만 진화 초기에 특정한 제어 파라미터가 과도하게 사용되는 것을 막기 위해 선택된 횟수를 조정한다. 제안된 방법을 12가지 함수 최적화 문제에 적용해서 그 결과를 비교함으로써 그 타당성과 효과를 시험한다.

시계열 데이터 분석은 공학 및 과학 연구의 근간이 되고 있는데 기존의 시계열 데이터 분석법은 데이터의 변동이 적고 유수의 독립성과 정규성에 의해 제한을 받는다[6-8]. 그러나 실제 시계열 데이터는 이런 조건을 충족시키지 못한다. 이런 접근법의 가장 큰 단점 가운데 하나는 복잡한 특성을 찾아내지 못하는데 있는데, 그 원인은 모든 시계열 데이터를 예측하고 특성화하려는데 있다. 데이터마이닝은 숨겨진 패턴을 찾을 목적으로 데이터를 분석하는 것으로 데이터가 많고 복잡하며 관계가 감춰져 있는 경우에 유용하다. 기존의 데이터마이닝 접근법은 발견해야 될 패턴의 구조에 대한 사전 지

접수일자 : 2002년 1월 22일
완료일자 : 2002년 8월 12일

식을 필요로 한다[9-11]. 최근에는 하드 클러스터링을 이용해서 위상공간에서 이벤트 예측을 위한 패턴을 찾는 연구가 이루어지고 있다[12]. 본 논문에서는 적응성 있는 차분 진화(ADE)의 응용 사례로 이벤트 예측 모델링을 위한 이벤트 클러스터링(EEC: Evolutionary Event Clustering)을 제안한다. 이벤트 클러스터링은 퍼지 목적 프로그래밍에 의해 결합된 다수의 목적함수에 의해 교사된다. 이벤트 예측 모델은 클러스터들의 중심점, 모양 결정 파라미터, 최소 및 최대 반경만으로 구성된다. 데이터와 클러스터 중심간의 거리와 클러스터 반경에 따른 퍼지 멤버쉽함수 값에 의해 이벤트 예측이 이루어지기 때문에 다차원인 경우에도 계산이 간단하다. EEC 알고리즘의 성능 검증을 위해 데이터 모델링 평가에 자주 사용되는 4가지 사례를 검토한다.

2. 차분 진화에 의한 함수최적화

이 절에서는 비선형이고 미분 불가능한 연속 공간 함수를 최적화하기 위한 차분 진화(DE: Differential Evolution)의 개요와 함수최적화를 위해 본 논문에서 제시한 적응성 있는 차분 진화(ADE: Adaptive Differential Evolution)에 대해 설명한다. 또한 12가지 함수최적화 문제에 대해 제시한 방법의 최적해 탐색능력을 시험하였으며 그 결과를 기존의 다른 방법들과 비교 검토한다.

2.1 차분 진화 개요

차분 진화는 집단에 속한 개체 벡터의 거리와 방향 정보를 사용한다. 이들은 5가지 진화 체계를 발표하였는데[1-3], 이 가운데 두 가지가 ICEC96 경연대회에서 10개의 시험 문제를 해결하는데 사용되었는데 진화알고리즘 가운데 가장 빨리 10개 문제를 풀 수 있었다. 이는 차분 진화가 효율적인 함수최적화 방법임을 입증하는 것이다. 이 알고리즘은 구조와 연산이 대단히 간단하지만 그럼에도 수렴속도가 매우 빠르며 알고리즘 제어 파라미터가 적어서 사용하기 편리하고 병렬처리에도 적합하다.

차분 진화는 집단 크기의 목적변수 벡터를 사용하는 탐색 방법이다. 집단 크기는 진화 과정에서 변하지 않는다. 초기 집단은 균일 분포로 랜덤하게 구성된다. 차분 진화의 핵심은 변화된 목적변수 벡터를 발생하는 체계에 있다. 차분 진화는 두개의 개체 벡터의 차이에 가중치를 곱한 것을 세 번째 개체 벡터에 더해서 교배용 벡터를 발생한다. 이 벡터와 교배 대상 벡터가 교배되어 새로운 벡터가 얻어진다. 이 새로운 벡터의 목적함수 값이 교배 대상 벡터의 것 보다 좋으면 이 벡터로 교배 대상 벡터를 대체한다. 즉, 교배를 통해 새로 만들어진 벡터가 더 우수하면 이 벡터는 살아남게 된다. 이는 유전알고리즘에서 종종 사용되는 엘리트주의와 유사한 효과를 거둘 수 있다. 집단에서 선택된 벡터의 거리와 방향 정보를 사용하여 벡터를 랜덤하게 변화시키는 것이 차분 진화로 하여금 뛰어난 수렴성을 갖도록 해준다.

개체 $x_{i,G}$에 대해 집단에서 랜덤하게 선택된 서로 다른 3개의 벡터로부터 교배용 벡터를 식(1)과 같이 만든다.

$$v_{i,G+1}=x_{r1,G}+F\cdot(x_{r2,G}-x_{r3,G}) \tag{1}$$

$x_{r1,G}$, $x_{r2,G}$와 $x_{r3,G}$는 i번째 새로운 개체 $v_{i,G+1}$를 만들기 위해 사용된 서로 다른 개체들이고 G는 진화 세대를, F는 선택된 교배 대상 벡터들의 차이를 제어하는 파라미터로 0과 2사이의 값을 갖는다. 차분 변화를 통해 얻어질 벡터가 식(2)와 같이 d차원이라 하자. 이 벡터는 식(1)의 교배용 벡터와 교배 대상 벡터인 $x_{i,G}$를 교배하여 만들어진 것이다. 교배 연산은 식(3)과 같이 정의된다.

$$x'_{i,G+1}=(x_{1i,G+1}, x_{2i,G+1}, \ldots, x_{di,G+1}) \tag{2}$$

$$x'_{ji,G+1}=(v_{ji,G+1}*x_{ji,G}) \quad j=1,\ldots,d \tag{3}$$

$*$는 교배 연산자로 균일 교배가 주로 사용된다. 균일 교배는 식(4)와 같이 이루어진다.

$$x'_{ji,G+1}=\begin{bmatrix} v_{ji,G+1} & \text{if}(rand\le Cr) \\ x_{ji,G} & otherwise \end{bmatrix} \tag{4}$$

$rand$는 0과 1 사이의 랜덤 변수, Cr은 교배율이다.

교배 후 만들어진 개체의 생존 여부는 $x'_{i,G+1}$와 $x_{i,G}$의 성능 비교를 통해 이루어진다. $x'_{i,G+1}$의 적합도가 $x_{i,G}$의 것보다 더 우수하면 $x_{i,G+1}=x'_{i,G+1}$이 되고 그렇지 않으면 $x_{i,G+1}=x_{i,G}$로 이전 개체를 그대로 사용한다.

차분 진화에서는 실수를 직접 사용하기 때문에 개체 평가시에 유전자형을 표현형으로 변환할 필요가 없다. 다음 세대를 위한 개체 선정이 적합도 비교에 의해서만 이루어지기 때문에 일반적인 진화알고리즘에서 필요한 선택 메커니즘과 최대 또는 최소화 문제에 따른 적합도 함수로의 변환이 필요없다. 이는 알고리즘의 처리 속도 개선에 유리하다. 차분 진화의 실행 단계를 정리하면 다음과 같다.

단계 1 : 초기 집단 구성(랜덤 값으로 μ개의 개체를 초기화. 각 개체는 n개의 목적변수로 구성, t=0)

$$P(t)=\{a_1(t), a_2(t), \cdots, a_\mu(t)\}, \quad a_k=\{x_{ik}, \cdots, x_{nk}\}$$

단계 2 : 집단내 모든 개체의 목적함수($Func$) 평가

$$\Phi(t)=\{\Phi(a_1(t)), \cdots, \Phi(a_\mu(t))\},$$
$$\Phi(a_k(t))=Func(x_{1k}, \cdots, x_{nk})$$

단계 3 : 모든 개체($i=1,\cdots,\mu$)에 대해 차분 변화를 위한 개체 a_{r1}, a_{r2}와 a_{r3}를 선택하여 교배용 벡터를 만들고 이를 교배 대상 벡터와 교배함

$$v_i(t)=a_{r1}(t)+F\cdot(a_{r2}(t)-a_{r3}(t)) \;//\text{ 교배용 벡터}$$
$$x'_i(t)=(v_i(t)*a_i(t)) \;//\text{ 교배}$$

단계 4 : 모든 개체의 목적함수($Func$) 평가

$$\Phi(t)=\{\Phi(x'_1(t)), \cdots, \Phi(x'_\mu(t))\}$$

최대화 문제이면 $\text{if}(\Phi(x'_i(t))>\Phi(a_i(t)))\quad a_{i(t)}=x'_i(t)$;

최소화 문제이면 $\text{if}(\Phi(x'_i(t))<\Phi(a_i(t)))\quad a_{i(t)}=x'_i(t)$

단계 5 : 종료조건을 확인하고 종료조건이 만족되지 않으면 t=t+1로 하고 단계3으로 복귀.

2.2 적응성 있는 차분 진화에 의한 함수최적화

이 절에서는 차분 진화에서 교배용 벡터를 생성하는 새로운 방법과 차분 진화의 제어 파라미터를 적응시킬 수 있는 방법을 제안해서 알고리즘의 강인성과 사용의 편이성을 개선하고자 한다. 제안한 방법을 다양한 함수최적화 문제에 적용함으로써 그 타당성과 효과를 평가한다.

2.2.1 교배용 벡터 생성(DE3)

기존의 차분 진화에서는 교배용 벡터를 생성하기 위해 5가지 방법을 제시하고 있다[1,2,4]. 그러나 이들 방법은 각기 특정한 문제에 대해 우수한 성능을 보이지만 고려된 모든 문제에서 우수한 성능을 보이지는 않았다. 본 논문에서는 기존의 5가지 방법 가운데 상대적으로 우수한 것을 혼합하여 사용한다. 즉, 진화 과정에서 가장 좋은 개체를 이용해서 알고리즘의 수렴성을 개선하면서 동시에 다양한 개체로부터 차분 변화를 만들어 내기 위해 식(5)와 같은 교배용 벡터 생성 방법을 제안한다. 차분 진화에서 교배용 벡터 생성 방법을 제외한 다른 부분은 동일하다. 이후 식(5)의 교배용 벡터 생성 방법에 의한 차분 진화를 DE3로 표기하며 이것의 성능 평가를 위해 함수최적화 문제를 다룬다.

$$v_{i,G+1}=(x_{best,G}+x_{r1,G})/2+F\cdot(x_{r2,G}+x_{r3,G}-x_{r4,G}-x_{r5,G}) \quad (5)$$

2.2.2 함수최적화

1) 함수최적화 문제

앞서 제안한 DE3 방법의 성능 평가 및 차분 진화의 최적 제어 파라미터(교배율 Cr 및 가중치 F)를 찾기 위해 모의 시험을 수행한다. 연속/불연속, 콘벡스/비콘벡스, 단일모달/멀티모달, 저차원/고차원 및 결정론적/확률적 특성을 고려하여 12개의 함수최적화 문제를 선택하였다(부록 참조). 부록에서 f_4의 경우 목적함수 값이 15 보다 작으면, f_5는 0.99800 보다 작으면 전역 최소 해를 찾은 것으로 한다. f_9는 오차의 제곱에 가중치를 곱해서 하나의 목적함수 값을 계산한다. 전역 최소점이 0인 경우는 목적함수 값이 10^{-6}의 정확도를 갖으면 전역 최소 해를 찾은 것으로 한다.

2) 성능평가 지수

알고리즘의 성능은 보통 온라인 성능, 오프라인 성능과 수렴하는데 걸린 함수평가 횟수 등에 의해 평가된다. 오프라인 응용에서는 다수의 함수평가를 실행하고 종료조건이 만족되면 그 때까지 가장 좋은 것을 저장하고 이를 사용한다. 하지만 온라인 응용의 경우 함수평가는 온라인 실험을 통해 이루어지기 때문에 오프라인과 달리 빠른 시간에 목표 성능에 도달하는 것이 중요하다. 본 논문에서는 알고리즘의 평균적인 수렴성을 평가하기 위해 전역 최소 해를 찾는데 걸린 목적함수 평가횟수를 평가기준으로 사용한다. 동일한 조건에서 10번의 실험을 수행하고 매 실험마다 랜덤한 초기 값을 사용한다. 최대 진화횟수가 maxgen이고 집단 크기가 popsize라면 전체 진화과정에서 평가될 수 있는 최대 평가횟수는 maxgen × popsize이다. 10번의 실험에서 한 번이라도 이 횟수에 도달할 때까지 최적 해를 찾지 못하면 수렴하지 못한 것으로 간주하고, 그 이전에 수렴하면 그 때까지의 실행한 함수평가 횟수를 평가기준으로 사용한다.

3) 시험 결과

차분 진화의 성능은 차분 진화 제어 파라미터(교배율과 가중치)에 영향을 받는다. 본 논문에서는 제안된 방법을 다양한 제어 파라미터 조건에서 시험하였다. 교배율은 0.2에서 1까지 0.1간격으로, 가중치도 0.2부터 1까지 0.1 간격으로 격자 점을 구성하고 각 점에서 f_1에서 f_9까지 10번 실험하여 수렴하는데 걸리는 함수평가 횟수의 평균을 구하였다. 그 결과 가운데 일부를 그림1과 그림2 보였다. 그림에서 nfeval은 10번의 실험에서 수렴하는데 걸리 함수평가 횟수의 평균값을 나타낸다. 시험결과를 보면 본 논문에서 제안한 차분 진화가 다른 진화알고리즘과 유사한 특성을 보임을 알 수 있다. 즉, 집단 크기가 커짐에 따라 수렴 가능성은 증대하지만 수렴속도는 나빠지며 수렴성이 알고리즘 제어 파라미터 F와 Cr에 매우 민감한 것이다. 제어 파라미터의 민감성은 기존의 5가지 차분 진화에도 공통적으로 제기되는 문제이다. 특히 차분 진화의 경우 문제에 따라 제어 파라미터 크기에 편차가 많기 때문에 유전알고리즘에서와 같이 제어 파라미터의 유효 범위를 설정하기 쉽지 않다. 이를 해결하기 위해서는 진화과정에서 문제에 따라 제어 파라미터가 스스로 적응할 수 있는 메커니즘이 요구된다.

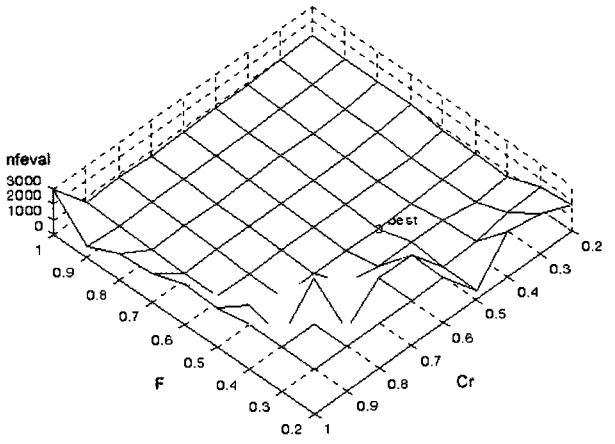

그림 1 F와 Cr에 따른 f_1의 평균 함수평가 횟수 (집단 크기 6)

Fig. 1 Average number of f_1 evaluation depending on F and Cr(population size 6)

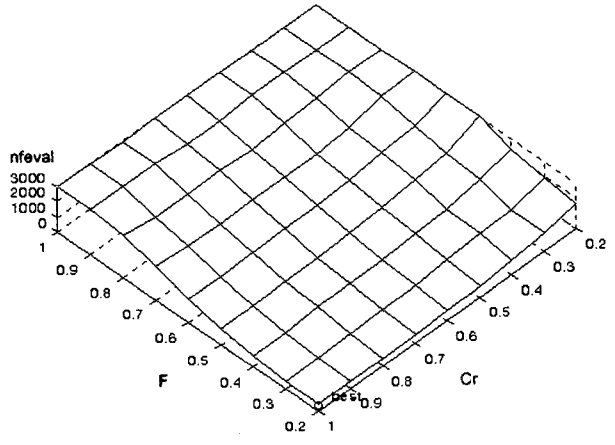

그림 2. F와 Cr에 따른 f_1의 평균 함수평가 횟수 (집단 크기 30)

Fig. 2 Average number of f_1 evaluation depending on F and Cr(population size 30)

2.2.3 적응성이 있는 차분 진화

제어 파라미터의 민감성 문제를 해결하기 위해 F와 Cr이 진화과정에서 스스로 문제에 적응할 수 있는 방법을 제안한다. 이는 차분 진화를 적용할 때 문제에 따라 적당한 제어 파라미터 값을 선정해야 하는 고민을 없애주기 때문에 그

만큼 차분 진화를 강인하고 사용하기 편리하게 해준다.

1) AFDE3

AFDE3에서 AF는 제어 파라미터 F를 적응시킨다는 의미이며 DE3는 앞서 제안한 식(5)에 의한 차분 진화를 표시한다. 제어 파라미터의 적응 알고리즘은 적응 대상 파라미터 집합을 구성하고 그 가운데 진화에 기여도가 많은 파라미터 값이 보다 빈번하게 선택되어 사용되도록 하는 것이다. 파라미터 값은 기본적으로 룰렛-휠과 같은 선택 메커니즘에 의해 결정된다. 진화 초기에 특정한 제어 파라미터가 우세해서 이후의 진화 과정에서 해당 파라미터가 과도하게 사용되는 것을 방지하기 위해 진화세대 단위로 파라미터가 선택된 횟수를 조정한다. AFDE3는 다음과 같은 과정을 거쳐 수행된다.

단계 1 : 초기 집단 구성(랜덤 값으로 μ개의 개체를 초기화. 각 개체는 n개의 목적변수로 구성, t=0)

$$P(t)=\{a_1(t), a_2(t), \cdots, a_\mu(t)\},\quad a_k=\{x_{ik}, \cdots, x_{nk}\}$$

제어 파라미터 F 값으로 사용될 파라미터 집합 F_s를 정의하고, F_s에서 파라미터가 선택되어 사용된 횟수를 기록하는 nF_s 값을 1로 초기화한다. 본 논문에서는 $F_1=0.2$ 이며 0.05씩 증가해서 $F_{17}=1.0$인 F_s를 사용한다.

$$F_s=\{F_1, \cdots, F_k, \cdots, F_{mk}\},\quad nF_s=\{nF_1, \cdots, nF_k, \cdots, nF_{mk}\}$$

단계 2 : 집단내 모든 개체의 목적함수($Func$) 평가

$$\Phi(t)=\{\Phi(a_1(t)), \cdots, \Phi(a_\mu(t))\},$$
$$\Phi(a_k(t))=Func(x_{1k}, \cdots, x_{nk})$$

단계 3 : 모든 개체($i=1, \cdots, \mu$)에 대해 사용할 제어 파라미터 F를 다음과 같이 선택하고

$$p_k=\sum_{j=1}^{k}\frac{nF_j}{\sum_{l=1}^{mk}nF_l},\quad \text{if}\,(p_{k-1}<rand\le p_k)\ F=F_k$$

$rand(\ni[0,1])$는 랜덤하게 발생된 값이다. 차분 변화를 위해 서로 다른 개체 a_{r1}, a_{r2}, a_{r3}, a_{r4}, a_{r5}를 선택하여 교배용 벡터를 만들고 이를 교배 대상 벡터와 교배한다.

$$v_i(t)=(a_{best,G}+a_{r1,G})/2+F\cdot(a_{r2,G}+a_{r3,G}-a_{r4,G}-a_{r5,G})$$
// 교배용 벡터
$$x'_i(t)=(v_i(t)*a_i(t))\ \text{// 교배}$$

단계 4 : 모든 개체의 목적함수($Func$) 평가

$$\Phi(t)=\{\Phi(x'_1(t)), \cdots, \Phi(x'_\mu(t))\}$$
if $(\Phi(x'_i(t))>\Phi(a_i(t)))$ { // 최대화 문제인 경우
$a_{i(t)}=x'_i(t)$
$nF_k=nF_k+1$
} // k는 단계3에서 개체별로 F_s에서 F 값으로 선택된 값의 인덱스

단계 5 : 진화 초기에 특정한 nF_k가 우세해서 전체 진화 과정에서 F_k가 과도하게 사용되는 것을 막기 위해 nF_s 값을 조정한다. k=1에서 mk까지 nF_k를 아래와 같이 조정한다.

$const$=3.0, $nF_{\min}=\min(nF_s)$, $nF_{\max}=\max(nF_s)$,
$nF_{avg}=average(nF_s)$

if $(nF_{\min}>(const\cdot nF_{avg}-nF_{\max}))$ {
$\triangle=nF_{\max}-nF_{avg}$
$a=(const-1)\cdot nF_{avg}/\triangle$
$b=nF_{avg}\cdot(nF_{\max}-const\cdot nF_{avg})/\triangle$ }
else {
$\triangle=nF_{avg}-nF_{\min}$
$a=nF_{avg}/\triangle$
$b=-nF_{\min}\cdot nF_{avg}/\triangle$ }
$nF_k=a\cdot nF_k+b$

단계 6 : 종료조건을 확인하고 종료조건이 만족되지 안으면 t=t+1로 하고 단계3으로 복귀.

F를 0.2에서 1 사이의 값 F_s에서 적응하도록 하고 Cr은 0.1에서 1까지 0.1 간격으로 10개의 격자 점을 구성하고 함수 f_1에서 f_9까지 10번 실험하여 수렴하는데 걸리는 함수평가 횟수의 평균을 구하였다. 그 결과가 그림3이다. 함수 f_7과 f_9를 제외하면 Cr 값의 변화에 크게 민감하지 않기 때문에 0.5~0.9 범위에서 Cr 값을 선정할 수 있다. 그렇지만 f_7과 f_9는 서로 극단적인 Cr 값을 요구하기 때문에 Cr 값도 적응시킬 필요가 있다.

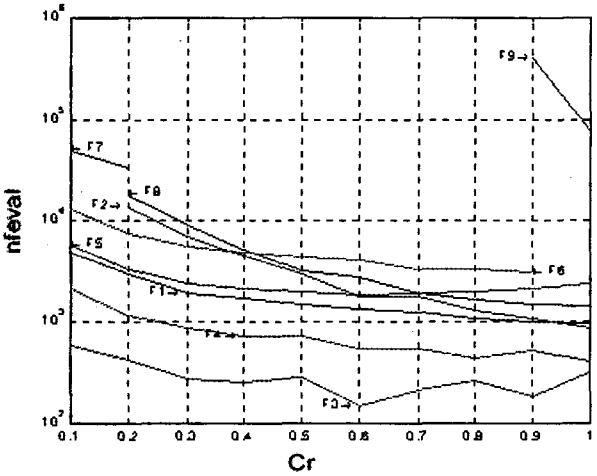

그림 3 AFDE3에 의한 평균 함수평가 횟수
Fig. 3 Average number of function evaluation by AFDE3

2) AFCDE3

AFCDE3에서 AFC는 제어 파라미터 F와 Cr 모두를 0.2~1 범위에서 적응시킨다는 의미이며 DE3는 앞서 제안한 식(5)에 의한 차분 진화를 표시한다. Cr에 대한 적응 메커니즘은 앞서 F를 적응시킨 방식과 동일하며 F_s와 nF_s 대신 C_{rs}와 nC_{rs}를 사용한다. C_{rs}와 nC_{rs}의 초기 값은 F_s와 nF_s의 것과 동일하다. 적용 결과를 정리한 것이 표1이다. 이 표에서 x는 전역 최적 해로 수렴하지 못함을, n.a.는 적용 불가를 표시하며 제시된 결과 중 ANM은 Annealed Nelder & Mead 방법, ASA는 Adaptive Simulated Annealing, DE1과 DE2은 차분 진화 결과이다[4]. AFDE3열은 Cr에 대해

가장 좋은 결과를 보인 것이고 AFCDE3열은 F와 Cr 모두를 적응시킨 것으로 집단 크기 외에는 고려할 제어 파라미터가 없다. 표1에 제시된 결과를 보면 비교적 목적함수가 간단한 f_1과 f_2의 경우에는 ANM 방법이 가장 효과적이지만 함수가 복잡해지면 전역 최적 해를 찾지 못한다. AFDE3의 경우 복잡도가 작거나 매우 큰 문제에서 가장 좋은 결과를 보이고 있다. AFCDE3의 경우에도 복잡성이 큰 문제의 경우에는 AFDE3 다음으로 수렴성이 좋지만 다른 경우에는 DE1과 DE2와 유사하거나 약간 떨어진다. 그렇지만 DE1과 DE2 모두 최적의 결과를 낳는 제어 파라미터를 탐색해서 얻은 결과임을 고려하면 만족스러운 결과로 볼 수 있으며, 이는 제어 파라미터의 적응 효과로 판단된다. AFCDE3에서 f_7인 경우 10번의 실행 가운데 한 두 번은 수렴하지 못하였다. 그림3을 보면 f_7의 경우 상대적으로 작은 Cr 값을 필요로 하기 때문인 것으로 보인다.

표 1 차분 진화의 시험 결과 비교
Table 1 Comparison of test results for function optimization

	ANM	ASA	DE1	DE2	AFDE3	AFCDE3
	T,TF,NV 평균	TRS,TAS 평균	NP,F,Cr 평균	NP,F,Cr 평균	NP,Cr 평균	NP 평균
f1	0,n.a.,195	1E-5,10397	10,0.5,0.3490	6,0.95,0.5392	10,0.9253	6318
f2	0,n.a.,1106	1E-5,1E511275	6,0.95,0.5746	6,0.95,0.5615	10,1275	151098
f3	300,0.99,20 90258	1E-7,100354	10,0.8,0.3915	20,0.95,0.21300	20,0.8882	101048
f4	300,0.98,30 X	1E-5,1004812	10,0.75,0.52378	10,0.95,0.22873	10,0.81945	102651
f5	3000,0.995,50 X	1E-5,1001379	15,0.9,0.3735	20,0.95,0.2828	15,0.61086	151155
f6	5E6,0.995,50 X	1E-5,1003581	10,0.4,0.2834	10,0.9,0.21125	12,0.81499	121763
f7	10,0.99,50 X	1E-5,0.1 X	30,1,0.322167	20,0.99,0.212804	20,0.225085	30 *
f8	5,0.95,52116	1E-6,30011864	10,0.8,0.51559	10,0.9,0.91076	12,0.9786	121018
f9	5E4,0.995,150 X	1E-8,700 X	100,0.65,1165680	80,0.6,1254824	80,151761	80165380

표1의 결과를 보면 차분 진화의 경우 적은 집단 크기로 전역 최적 해를 찾는 능력을 보여주고 있다. 상대적으로 작은 집단 크기는 최적 해를 찾는데 보다 적은 수의 함수평가를 필요로 할 것이고, 이는 알고리즘의 수렴 속도를 개선할 것이다. 따라서 본 논문에서 제안한 차분 진화를 보다 객관적으로 평가하기 위해서 집단 크기가 동일한 조건에서 다른 진화알고리즘들과 성능을 비교한다. 이를 위해 집단 크기를 200으로 하고 함수 f_{10}, f_{11} 및 f_{12}의 최적화를 수행하였다. f_{11}의 경우 함수평가 횟수가 40,000번, f_{10}과 f_{12}의 경우에 100,000번에 도달하면 진화를 종료하고 그때까지 가장 좋은 함수 값을 최종 수렴된 목적함수 값으로 한다. 이 실험을 20번 반복하여 최종 수렴된 목적함수 값의 평균과 표준편차로 알고리즘들을 비교 평가한다.

표 2 차분 진화와 다른 진화알고리즘의 비교
Table 2 Comparison of AFDE3, AFCDE3 and other evolutionary algorithms

		ES1	ES30	EP	GA	AFDE3	AFCDE3
		NP, Nsigma 200, 1	NP, Nsigma 200, 30	NP 200	NP, Cr, Mr 200, 0.6, 0.001	NP 200	NP 200
f11	평균	1.075E-5	6.672E-1	1.998E+2	1.647E+2	1.09E-8(Cr:0.7)	3.43E-6
	표준편차	4.730E-6	2.610E-1	6.564E+1	4.741E+1	2.61E-8	6.89E-6
f12	평균	4.1	0	0	5.39E+1	0(모든 Cr)	0
	표준편차	3.177	0	0	4.741E+1	0	0
f10	평균	1.326	1.618E-3	1.976	5.253	0.62E-13(Cr:0.6)	5.80E-12
	표준편차	1.039	9.290E-4	6.300E-1	5.13E-1	2.03E-13	9.43E-12

비교에 사용된 다른 진화 알고리즘은 진화전략 ES1과 ES30, 진화프로그래밍 EP, 유전알고리즘 GA이며 이들에 사용된 제어 파라미터는 참고문헌[5]에 상세히 기술되어 있다. 실험 결과가 표2에 보여진다. 이 표를 보면 동일한 집단 크기에서도 본 논문에서 제안한 AFDE3와 AFCDE3의 수렴성이 가장 뛰어남을 알 수 있다. 따라서 표1과 표2의 결과를 종합하면 제안된 차분 진화가 단순함에도 불구하고 성능이 우수하며 문제에 따라 제어 파라미터가 적응되기 때문에 알고리즘이 강인하다고 할 수 있다. 또한 집단 크기를 제외하고는 고려해야 할 제어 파라미터가 없기 때문에 사용이 편리하다.

3. 차분 진화에 의한 이벤트 클러스터링

본 절에서는 적응성 있는 차분 진화의 응용 사례로써 시계열 데이터 예측에 데이터마이닝 개념을 적용해서 데이터에 숨겨진 패턴을 찾아내기 위한 방법, 특히 예측 대상인 이벤트를 특성화해서 예측이 가능하도록 숨겨진 패턴을 찾는 방법을 제시한다. 이는 결정론적이고(deterministic) 무질서한(chaotic) 시계열 데이터를 포함해서 비주기적이고 불규칙하며 변동이 심한(non-stationary) 시계열 데이터를 처리할 능력을 갖는다. 이 방법은 앞서 제시한 차분 진화(AFCDE3)를 통해 교사(supervised)에 의한 클러스터링을 수행함으로써 기존의 한계를 극복할 수 있으며 다차원 시계열 데이터에도 쉽게 확장될 수 있다. 또한 다수의 목적함수를 처리할 수 있도록 퍼지 목적프로그래밍의 개념도 도입된다. 최종 이벤트 예측 모델은 클러스터들의 중심점, 모양 결정 파라미터, 최소 및 최대 반경만으로 구성된다. 데이터와 클러스터 중심간의 거리와 클러스터 반경에 의한 퍼지 멤버쉽함수 값에 의해 이벤트 예측이 이루어지기 때문에 예측에 필요한 계산이 간단하다.

3.1 진화기반 이벤트 클러스터링

3.1.1 이벤트 예측 모델

본 연구는 시계열 데이터에서 이벤트를 특성화해서 이벤

트 발생을 예측하는 것이다. 시계열 데이터는 이벤트 예측 모델을 찾는데 사용되는 데이터 X(이후 모델용 데이터로 칭함)와 이를 평가하는데 사용되는 데이터 Z(평가용 데이터로 칭함)로 나뉜다. 예측하고자 하는 이벤트를 정의하는 이벤트 특성화함수 $g(t)$를 규정하고 모델용 데이터 집합에서 특성화 함수를 만족하는 데이터 X_{event}와 그렇지 못한 데이터의 집합 X^c_{event}을 구분한다. 특성화함수는 시계열 데이터를 이벤트와 이벤트가 아닌 것으로 구분할 수 있어야 한다. 이벤트 클러스터링은 X로부터 $g(X_{event})$와 $g(X^c_{event})$를 분리해서 $g(X_{event})$만을 찾아낼 수 있는 특징과 패턴을 내포하는 최적의 클러스터 모델을 찾아내는 것이다. 이렇게 찾은 클러스터 모델은 Z 데이터에 대해 $g(Z_{event})$를 예측할 수 있어야 하며 이것이 본 논문에서 말하는 이벤트 클러스터링의 목적이다. 이는 시계열 데이터를 다차원 공간의 한 점으로 해석하고 유사한 특성을 갖는 데이터는 공간상에서 가까이 위치한다는 가정에 근거한다. 그림4에서 점으로 표시된 데이터는 임의의 특성화함수에 의해 분류된 X_{event}이고 x로 표시된 데이터는 X^c_{event}라 하자.

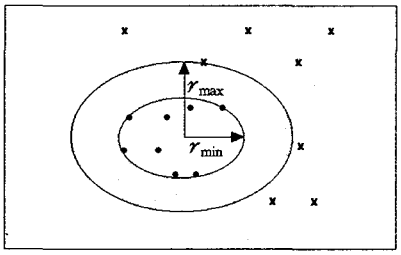

그림 4. 이벤트 예측을 위한 클러스터 모델
Fig. 4 Cluster model for event prediction

그림에서 보면 X_{event}를 예측하기 위한 최적의 클러스터 모델은 하나의 클러스터를 가지며 그 클러스터의 반경 r은 r_{min}에서 r_{max} 사이의 값을 가질 수 있다. 반경 r을 r_{min}으로 선택하면 모델링에 사용되지 않은 데이터에 대해서 이벤트가 아닌 것을 이벤트로 분류할 가능성은 작아지지만 이벤트인 것을 이벤트가 아닌 것으로 분류할 가능성 또한 증가한다. 반대로 반경 r을 최대인 r_{max}로 하면 이벤트로 분류할 가능성이 높아지는 만큼 이벤트가 아닌 것을 이벤트로 분류할 가능성도 높아진다. 따라서 반경 r은 r_{min}과 r_{max} 사이에서 애매한 특성을 보인다. 즉, 유사한 시계열 데이터가 공간상에서 얼마나 가까이 위치해야 하는지 불확실하다.

이 문제를 해결하기 위해 본 논문에서는 식(6)과 같은 퍼지 멤버쉽함수로 이벤트 예측 값을 표현한다. 평가용 데이터가 반경 r_{min}인 클러스터 내부에 존재하면 이벤트로 예측하며(멤버쉽함수 값 u=1), 반경 r_{max} 밖에 존재하면 이벤트가 아닌 것(u=0)으로 그리고 반경 r_{min}과 r_{max} 사이에서는 거리에 비례하는 값을 갖는다. 이벤트 예측 클러스터 모델을 찾기 위한 클러스터링 과정에서는 퍼지 멤버쉽함수를 사용하지 않으며 이벤트 분류를 최대화 할 수 있는 r_{min}을 결정한 다음 클러스터링 종료 후에 X^c_{event}에 대해 r_{max}를 계산한다. 이벤트 예측 모델을 구성하는 클러스터가 다수인 경우는 식(7)과 같이 개별 클러스터에 대한 이벤트 예측 값 가운데 최대 값을 취한다. 그 값이 0.5 보다 크면 이벤트인 것으로 판단하고 그렇지 않으면 이벤트가 아닌 것으로 한다. 퍼지 멤버쉽함수의 도입은 공간에서 근접하게 위치한 이벤트와 이벤트가 아닌 점들에 대한 분별력을 높여 줄 것이다. 다차원 공간에서 데이터와 클러스터 중심간의 거리는 식(8)에 의해 계산된다.

$$u_i=\begin{pmatrix} 0, & d_{ik}>r_{imax} \\ 1, & d_{ik}\le r_{imin} \\ 1-(d_{ik}-r_{imin})/(r_{imax}-r_{imin}), & r_{imin}<d_{ik}<r_{imax} \end{pmatrix} \quad (6)$$

여기서 d_{ik}는 다차원 공간상의 k번째 데이터와 i번째 클러스터 중심간의 거리로 식(8)과 같이 정의된다.

$$u=\max(u_i, i=1,\cdots,c) \quad (7)$$

c는 클러스터 수이고 u_i는 i번째 클러스터에 대한 멤버쉽 함수 값이다

$$d_{ik}=\sqrt{a_{1i}\cdot(x_{1k}-c_{1i})^2+\cdots+a_{ji}\cdot(x_{jk}-c_{ji})^2+\cdots+a_{qi}\cdot(x_{qk}-c_{qi})^2} \quad (8)$$

여기서 x_{jk}는 차원이 q인 k번째 데이터의 j번째 변수 값, c_{ji}는 i번째 클러스터의 j번째 변수에 대한 클러스터 중심 값이다. a_{ji}는 클러스터 모양을 결정하는 파라미터로 [0,1] 사이의 값을 가지며, 모두 동일한 값을 가지면 hyperspheric형 그렇지 않으면 ellipsoidal형이다.

3.1.2 진화기반 클러스터링

본 절에서는 앞서 설명한 이벤트 예측 모델을 얻기 위해 진화에 의한 이벤트 클러스터링(EEC: Evolutionary Event Clusteing) 알고리즘을 제안한다. 클러스터링을 적용할 때 가장 흔히 접하는 문제는 데이터 분할에 적합한 클러스터 수를 결정하는 것이다. 주로 사용되고 있는 방법은 클러스터 유효지수(validity index)[13]를 사용하는데, 이는 반복적인 처리를 필요로 하기 때문에 다차원이고 데이터가 많은 경우는 처리시간이 길어진다. 또한 데이터 특성에 따라 유효지수 함수의 국부 최소점이 명확하지 않은 경우 적절한 데이터 분할을 표시하지 못한다[14]. 이런 문제를 해결하기 위해 EEC는 교사에 의한 클러스터링과 클러스터 삭제를 채용하며 이는 최적의 클러스터 수를 찾는 동시에 다양한 수의 데이터를 포함하는 클러스터를 찾을 수 있도록 해준다.

EEC 알고리즘은 필요하다고 생각되는 클러스터의 수 보다 큰 값으로 초기화된다. 차분 진화 AFCDE3에 의해 목적함수인 식(9)를 최대화하며 동시에 식(10)을 최소화하도록 클러스터를 진화시킨다. 차분 진화 종료조건(최대 진화 세대수)을 만족하면 불필요한 클러스터 삭제가 이루어진다. 이벤트 데이터를 포함하고 있지 못한 클러스터와 다른 클러스터에 포함된 이벤트만을 갖고 있는 클러스터는 삭제된다. 진화가 충분히 이루어지면 후자에 해당하는 클러스터는 발생하지 안는다. 이는 목적함수로 사용된 식(9)와 식(10)에 의해서 이벤트 분류를 최대로 하는 동시에 사용된 클러스터들의 반경 합을 최소화하도록 클러스터링이 이루어지기 때문이다. 진화가 완료되면 클러스터의 최소 반경 r_{imin}과 최대 반경 r_{imax}을 계산하고 이것과 클러스터 중심 c_{ji}, 클러스터 모양 결정 파라미터 a_{ji}를 저장하면 예측 모델이 완성된다. 식(8)에 의해 데이터와 클러스터 중심간의 거리가 계산되면 식(6)과 식(7)에 의해 이벤트 여부를 예측할 수 있다.

$$f_{event}=\frac{1}{n}\cdot\left(\sum_{i=1}^{nX_{event}}B_i+\sum_{i=1}^{nX^c_{event}}B^c_i\right) \quad (9)$$

여기서 n은 다차원 데이터의 수, nX_{event}는 이벤트인 데이터 수, nX^c_{event}는 이벤트가 아닌 데이터 수로 $n=X_{event}+X^c_{event}$이다. B_i와 B^c_i는 이벤트를 이벤트로, 이벤트가 아닌 것을 이벤트가 아닌 것으로 분류하면 1이고 그렇지 않으면 0이다.

$$f_{radius}=\sum_{i=1}^{c}r_{imin} \quad (10)$$

여기서 c는 전체 클러스터 수이고 r_{imin}은 i번째 클러스터의 최소 반경이다.

그러나 식(9)는 최대화를 식(10)은 최소화를 위한 것이기 때문에 이들을 결합하여 하나의 최대화 또는 최소화 목적함수로 만들 필요가 있다. 또한 이 두 목적함수 값의 척도를 동등하게 해서 특정한 하나가 다른 것을 압도해서는 안 된다. 물론, 의도적으로 특정한 하나에 더 비중을 두어 진화가 이루어지도록 통제할 수 있어야 한다. 이를 위해 퍼지 목적프로그래밍을 도입한다.

3.2 퍼지 목적 프로그래밍에 의한 최적화

대부분의 실용적인 최적화 문제에서는 하나 이상의 목적함수를 동시에 고려해야 한다. 서로 다른 다수의 목적함수를 하나의 목적함수로 결합하는 것은 간단하지 않다. 그 이유는 목적함수의 척도와 그 중요도가 서로 다르고 어떤 경우는 그 목적이 최소화인 반면 다른 경우는 최대화일 수 있기 때문이다. 또한 최적화 목표를 명확하게 표현할 수 없는 경우가 종종 존재한다. 따라서 퍼지 이론을 사용하는 퍼지 목적 프로그래밍(Fuzzy Goal Programming)을 사용할 필요가 있다. 나라심한(Narasimhan)[15]이 목적 프로그래밍에 처음으로 퍼지 이론을 사용한 이래, 퍼지 목적 프로그래밍의 적용 사례는 많다. 최근에는 다수의 목적함수를 갖는 문제에 퍼지 이론과 진화알고리즘을 적용하는 것이 주목받고 있다[16-18].

이 절에서는 목표가 애매한 다수의 목적함수를 갖는 최적화 문제를 퍼지 목적 프로그래밍으로 변환하여 차분 진화로 해결하는 것을 설명한다. 퍼지 목적 프로그래밍의 핵심은 다수의 목적함수를 하나의 최대화 목적함수로 결합하는 것이다. 이렇게 결합된 목적함수는 추가적인 변환 없이 진화알고리즘의 적합도 함수로 바로 사용될 수 있기 때문에 퍼지 목적 프로그래밍에 진화알고리즘을 적용하는 것은 자연스럽다. 목적함수의 결합은 퍼지 이론의 멤버쉽함수라는 것을 통해 이루어진다. 멤버쉽함수는 다수의 목적함수를 동시에 처리할 수 있고 최적화 문제에 따라 조정될 수 있기 때문에 유용하다. 즉, 척도가 다르고 최적화 목적이 다른 목적함수들을 멤버쉽함수에 의해 하나의 목적함수로 결합할 수 있다. 퍼지 이론에서 목표(goal)는 목적함수 값에 따라 변하는 멤버쉽함수로 표현된다. 이 함수는 0과 1 사이의 값을 출력한다. 이벤트 클러스터링의 경우 멤버쉽함수를 이용하여 패턴 분류를 최대화하는 목적함수인 식(9)와 클러스터 반경 최소화를 위한 목적함수인 식(10)을 하나의 최대화 목적함수로 통합한다. 목적함수를 최대로 하는 경우는 식(11), 최소로 하는 경우는 식(12)와 같은 멤버쉽함수를 사용한다.

$$\mu(f_{event})=\begin{cases}0 & f_{event}<f_{event-\min}\\ 1 & f_{event}>f_{event-\max}\\ \dfrac{f_{event}-f_{event-\min}}{f_{event-\min}-f_{event-\max}} & f_{event-\min}<f_{event}<f_{event-\max}\end{cases} \quad (11)$$

여기서 f_{event}는 식(9)의 값, $\mu(f_{event})$는 f_{event}의 멤버쉽함수 값, $f_{event-\min}$는 목적함수 값이 허용할 수 있는 최소한계로 0, $f_{event-\max}$는 이 값을 넘어서면 더 이상의 증가가 무의미한 값으로 1을 사용한다. k는 상수로 1을 사용한다.

$$\mu(f_{radius})=\begin{cases}1 & f_{radius}<f_{radius-\min}\\ 0 & f_{radius}>f_{radius-\max}\\ \dfrac{f_{radius-\min}-f_{radius}}{f_{radius-\min}-f_{radius-\max}} & f_{radius-\min}<f_{radius}<f_{radius-\max}\end{cases} \quad (12)$$

여기서 f_{radius}는 식(10)의 값, $\mu(f_{radius})$는 f_{radius}의 멤버쉽함수 값, $f_{radius-\min}$는 이 값을 넘어서면 더 이상의 감소가 무의미한 값으로 1, $f_{radius-\max}$는 목적함수 값의 허용할 수 있는 최대한계로 0을 사용한다. k는 상수로 1을 사용한다.

서로 목적이 다른 목적함수들이 이런 멤버쉽함수를 통해 변형되고 비교, 결합될 수 있다. 개별적인 목적함수의 상대적 중요도는 그 영향 정도에 따라 변경될 수 있다. 개별적인 목적함수의 멤버쉽함수 값을 하나의 통합된 값으로 결합하기 위해 식(13)과 같이 목적함수 별로 가중치를 사용한다. 가중치 β_{event}과 β_{radius}는 0과 1 사이의 값으로 상대적 중요도를 표시한다. 여기서는 $\beta_{radius}=1-\beta_{event}$를 사용한다.

$$\mu_{total}(f)=\beta_{event}\cdot\mu(f_{event})+\beta_{radius}\cdot\mu(f_{radius}) \quad (13)$$

이상에서 설명한 이벤트 예측 모델 구성을 위한 클러스터링(ECC) 알고리즘을 정리하면 다음과 같다.

단계1 : 초기 클러스터 수, AFCDE3에 대한 집단 크기와 종료조건 결정. 클러스터링 데이터의 변수별 크기와 영향을 균등하게 하기 위해 모든 데이터를 [0.1,0.9] 사이로 정규화 한다.

단계2 : AFCDE3에 의해 목적함수인 식(13)을 최대로 하는 클러스터의 중심, 모양 결정 파라미터와 반경을 구한다.

단계3 : 불필요한 클러스터를 제거한다.

단계4 : 남은 클러스터에 대해 최소 반경과 최대 반경을 계산한다.

단계5 : 클러스터의 중심, 모양 결정 파라미터, 최소 및 최대 반경을 이벤트 예측 모델로 사용한다.

3.3 이벤트 클러스터링 시험

앞 절에서 제안한 EEC 알고리즘의 성능을 검증하기 위해 4가지 사례를 검토한다.

3.3.1 랜덤 데이터

두 그룹으로 구성된 이차원 랜덤 데이터를 고려한다. 그룹1은 표준정규분포(평균 0, 표준편차 0.2)로 발생된 300개의 샘플로 구성되고 그룹2는 표준정규분포(평균 2, 표준편차 0.05)로 발생된 5개의 샘플을 갖는다. 이렇게 그룹1과 그룹2의 데이터 수에 차이가 많은 경우 fuzzy c-means 클러스터링은 종종 데이터 수가 작은 그룹을 놓친다. 본 논문에서 제안한 이벤트 클러스터링 알고리즘을 적용한 사례가 그림5와 그림6에 보여진다. 여기서는 모든 데이터를 이벤트로 간주했으며 초기 클러스터의 수는 10, AFCDE3의 집단 크기는 50, 진화 세대수는 100으로 클러스터링 하였다. 알고리즘에서 불필요한 클러스터들이 제거된 결과 최종적으로 하나의 클러스터만이 남았다. 클러스터링 결과를 보면 식(13)의 목적함수 가중치에 따라 클러스터링이 달라짐을 알 수 있다. 그림5는 β_{event}을 0.99, 즉 이벤트 분류 최대화에 중요도를 크게 한 경우이다. 우측 상단에 있는 그룹2 데이터가 클러스터에 포함되어 있다. 반면에 그림8에서는 β_{event}을 0.5로 한 경우로 이벤트 분류 최대화와 클러스터 반경 최소화에 동일한 비중을 주었다. 결과로 적은 수의 이벤트를 분류하지 않은 대신 클러스터 반경을 최소화한 것을 알 수 있다.

또 다른 랜덤 데이터의 예로 서로 다른 표준정규분포 파라미터를 사용하여 5개 그룹으로 총 260개 이차원 데이터를 고려한다. 이 가운데 두 그룹은 이벤트가 아니고 세 그룹은 이벤트이다. 앞서의 경우와 동일한 조건에서 클러스터링을 실행한 결과가 그림7이다. 그림에서 이벤트는 점으로 표시되며 이벤트가 아닌 것은 x로 표시된다. 성공적으로 이벤트를 분류해냄을 알 수 있다.

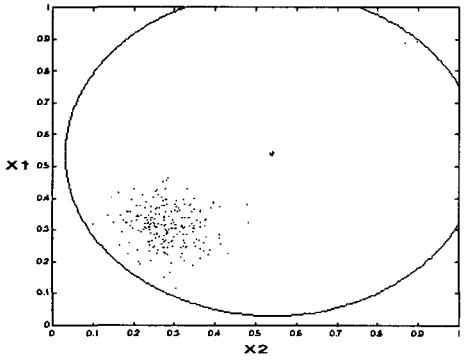

그림 5. 두 그룹 랜덤 데이터에 대한 클러스터 모델(β_{event}=0.99, 이벤트 분류 성공률 100%)

Fig. 5 Cluster model for two groups of random data(β_{event}=0.99, success rate of classification=100%)

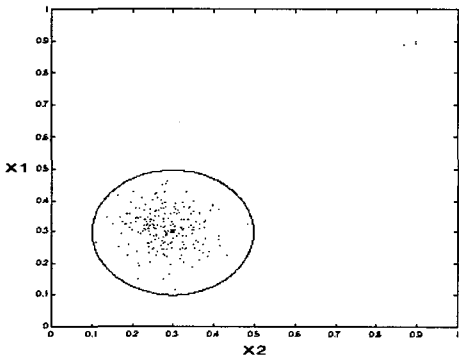

그림 6 두 그룹 랜덤 데이터에 대한 클러스터 모델(β_{event}=0.5, 이벤트 분류 성공률 98.4%)

Fig. 6 Cluster model for two groups of random data(β_{event}=0.5, success rate of classification=98.4%)

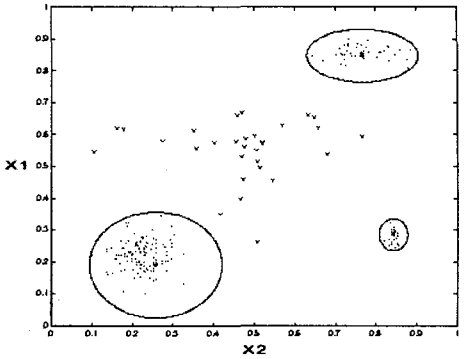

그림 7 다섯 그룹 랜덤 데이터 클러스터링(β_{event}=0.99, 이벤트 분류 성공률 100%)

Fig. 7 Cluster model for five groups of random data(β_{event}=0.99, success rate of classification=100%)

3.3.2 비선형 시스템

데이터 모델링에서 많이 사용되는 다음과 같은 비선형 시스템을 고려한다. 식(14)로부터 만들어진 50개의 데이터[19] 가운데 처음 25개는 이벤트 예측 모델용으로 사용하고 나머지 25개는 검증용으로 사용한다. 50개의 데이터에는 4개의 입력이 사용되었는데 이 가운데 실제 데이터 생성에 기여한 입력은 x_1과 x_2이고 x_3과 x_4는 임의의 값을 갖는다. 이벤트 특성화함수로 식(15)를 사용한다. 50개의 데이터 가운데 식(15)를 만족하는 것을 이벤트로 간주한다.

$$y=(1+x_1^{-2}+x_2^{-1.5}),\ 1\le x_1, x_2\le 5 \tag{14}$$

$$g(y\ge 2.5) \tag{15}$$

초기 클러스터는 10개, 집단크기 50, 진화 세대수 500으로 EEC를 실행했고 그 결과 그림8처럼 2개의 이벤트 클러스터를 얻었다. 이 결과에서 보면 식(8)에서 x_3과 x_4에 대한 파라미터 a_{ji}는 모두 0이었으며 두 번째 클러스터의 경우 x_1에 대한 것도 0 임을 알 수 있다. 이는 x_3과 x_4과 이벤트 분류에 관련이 없음을 의미하는 것으로 실제 데이터에 부합하는 결과이다. 두 번째 클러스터는 x_2만으로 이벤트를 분류할 수 있음을 보인 것이다. 그림9는 두 개의 클러스터를 검증용 데이터에 적용한 결과이다. 이벤트 예측 성공율은 88%이었다. 실제 검증용 데이터에는 식(14)에 만들어진 것이 아닌 데이터(outlier)가 있음을 감안하면 만족스러운 결과이다.

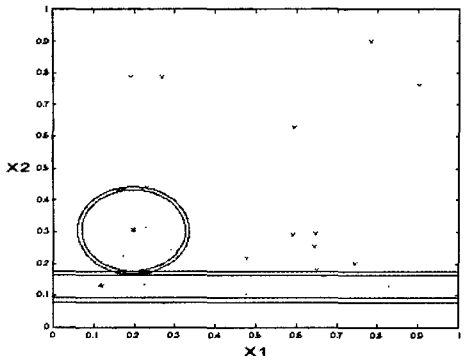

그림 8 이벤트 예측 모델(β_{event}=0.99, 이벤트 분류 성공률 96%)

Fig. 8 Event prediction model(β_{event}=0.99, success rate of classification 96%)

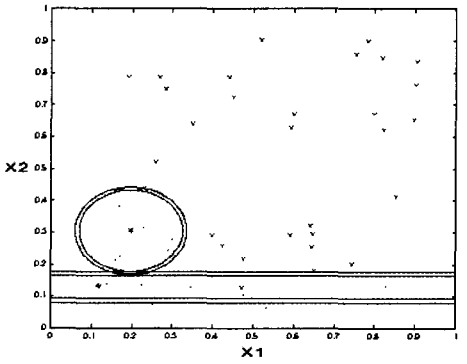

그림 9 이벤트 예측 결과(이벤트 예측 성공률 88%)

Fig. 9 Result of event prediction(success rate of prediction=88%)

3.3.3 가스로 데이터

데이터 모델링에 많이 인용되는 또 다른 예로 Box와 Jenkins가 제시한 동적 프로세스 데이터를 이벤트 클러스터링에 적용한다. 프로세스는 가스 흐름율인 입력 $u(k)$와 CO_2 밀집도인 출력 $y(k)$를 갖는 가스 용광로이다. 사용 가능한 데이터 집합은 296개이다[7,19]. 통상적으로 입력으로 $u(t-4)$와 $y(t-1)$가 주로 사용된다. 이용할 데이터 쌍은 292개이고 이 가운데 앞의 반은 이벤트 예측 모델링에 나머지 반은 이벤트 검증용으로 사용한다. 클러스터링을 위한 조건은 초기 클러스터 수 10, 진화 세대수 1000이다. 사용한 이벤트 특성화함수는 식(16)과 같다.

$$g(50 \le y \le 57) \tag{16}$$

그림10은 이벤트 예측 모델링 결과로 이벤트 분류 성공률은 98%이다. 그림11은 이벤트 예측 모델로 얻은 1 개의 클러스터를 검증용 데이터에 적용한 결과이다. 이벤트 예측 성공율은 93.1%이었다. 이번에는 입력 변수를 $y(t-1)$에서 $y(t-4)$까지와 $u(t-1)$에서 $u(t-6)$까지 총 10개를 사용하여 위와 동일한 조건에서 실험하였다. 결과는 클러스터 수 1, 이벤트 분류 성공률 99%, 이벤트 예측 성공률 93.8%이었다. 클러스터에서 10개 입력이 모두 사용되지는 안았다. 입력 변수를 늘렸음에도 이벤트 예측 성공률의 증가는 1% 미만이었다. 이는 입력이 $u(t-4)$와 $y(t-1)$만으로 충분함을 나타낸다.

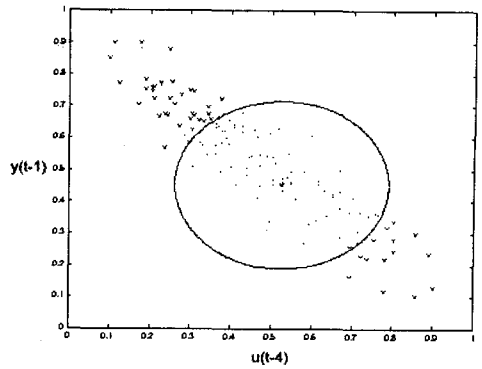

그림 10 이벤트 예측 모델(β_{event}=0.99, 이벤트 분류 성공률 98%)

Fig. 10 Event prediction model(β_{event}=0.99, success rate of classification 98%)

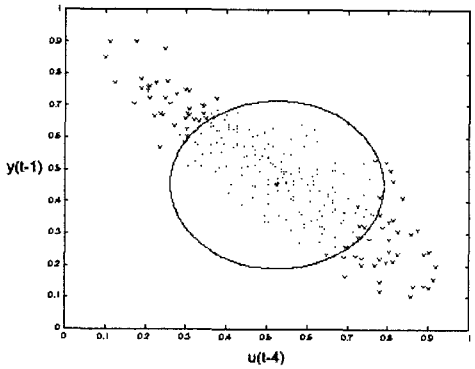

그림 11 이벤트 예측 결과(이벤트 예측 성공률 93.1%)

Fig. 11 Result of event prediction(success rate of prediction=93.1%)

3.3.4 시계열 데이터

데이터 모델링에 많이 사용되는 또 다른 예로 Mackey-Glass 시계열 데이터를 고려한다. 이 시계열 데이터는 유사 주기적이고 무질서한 특성을 갖는다[20]. 식(17)로부터 1500개의 데이터를 만들어 118번째로부터 500개는 이벤트 예측 모델용으로 그 뒤 500개는 모델 검증용으로 사용한다. 입력은 $x(t), x(t-6), x(t-12)$와 $x(t-18)$이고 출력은 $x(t+6)$이다.

$$\dot{x(t)} = \beta x(t) + \frac{\alpha x(t-\tau)}{1+x^{10}(t-\tau)}, \quad \alpha=0.2, \quad \beta=-0.1, \quad \tau=17, \quad x(0)=1.2 \tag{17}$$

이벤트 클러스터링을 위한 초기 조건으로 시작 클러스터 수는 10, 집단 크기 50, 진화 세대수 1000을 사용하였다. 이벤트 특성화함수는 식(18)을 사용한다.

$$g\left(\frac{x(t+6)-x(t)}{x(t)} > 0.01\right) \tag{18}$$

EEC 결과 2개의 클러스터로 분류되었고 입력은 $x(t)$, $x(t-6)$와 $x(t-12)$만으로 충분한 것으로 나타났으며 이벤트 분류 성공률은 97.8%이다. 검증용 데이터에 대해 이벤트 예측을 수행한 결과 성공률은 96%였다. 이상의 결과를 보면 본 논문에서 제시한 이벤트 클러스터링 알고리즘이 효과적으로 이벤트를 예측할 수 있음을 알 수 있다.

4. 결 론

본 논문에서는 새로운 교배용 벡터 생성법과 제어 파라미터의 적응 메커니즘을 결합한 차분 진화(ADE)를 제안했다. 12가지의 함수 최적화 문제를 통해 제안된 방법의 타당성을 검증하였으며 다른 알고리즘과 성능을 비교를 하였다. 그 결과 차분 진화의 빠른 수렴성을 해치지 않으면서 알고리즘의 강인성과 사용의 편이성을 개선할 수 있었다. 이는 진화 과정에서 제어 파라미터가 적합한 값으로 적응되었음을 의미한다. 응용 사례로서 시계열 데이터에서 이벤트 예측 모델을 찾는데 ADE를 적용하는 방법(EEC)을 제안하였다. 데이터와 클러스터 중심간의 거리에 기초한 이 예측 방법은 다차원이고 그 구조가 알려지지 않은 경우에도 이벤트 예측을 위한 패턴을 찾아낼 수 있었다. 데이터 모델링 검증에 널리 사용되는 4가지 사례를 통해 제안된 방법을 검증하였다. 사례2의 비선형 시스템을 제외한 모든 경우에서 모델링시의 이벤트 분류 성공률과 검증시의 이벤트 예측 성공률간에 차이를 5% 이내로 할 수 있었다. 사례2의 경우도 외래 데이터를 제외하면 차이는 5% 보다 작다.

참 고 문 헌

[1] Price and R. Stron, "Differential evolution: Numerical optimization made easy," *Dr. Dobbs Journal*, pp 18-24, April 1997.

[2] R. Storn, "Minimizing the real functions of the ICEC96 contest by Differential Evolution," *IEEE Conference on Evolutionary Computation*, pp. 842-844, Nagoya, 1996.

[3] R. Storn, System design by constraint adaptation and differential evolution, Technical Report TR-96-039, ICSI, Nov. 1996.

[4] R. Stron and K. Price, Differential Evolution a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, ICSI.

[5] Thomas Back and Hans-Paul Schwefel, An overview of evolutionary algorithms for parameter optimization, *Evolutionary Computation*, Vol. 1, No. 1, pp. 1-23, 1993.

[6] S. M. Pandit and S.-M. Wu, *Time series and system analysis with applications*, New York: Wiley, 1983.

[7] G. E. P. Box and G. M. *Jenkins, Time series analysis: forecasting and control*, Holden-Day, 1976.

[8] B. L. Bowermann and R. T. *O'Connell, Forecasting and time series: an applied approach*, Duxbury Press, 1993.

[9] D. J. Berndt and J. Clifford, "Finding patterns in time series: A dynamic programming approach," in Advances in knowledge discovery and data mining, AAAI Press, pp.229-248, 1996.

[10] E. J. Keogh and M. J. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," Proc. of AAAI Workshop on Predicting the Future: AI Approaches to Time Series Analysis, Madison, Wisconsin, 1998.

[11] M. T. Rosenstein and P. R. Cohen, "Continuous categories for a mobile robot," Proc. of Sixteenth National Conference on Artificial Intelligence, 1999.

[12] Richard J. Povinelli. "Using Genetic Algorithms to Find Temporal Patterns Indicative of Time Series Events," GECCO 2000 Workshop: Data Mining with Evolutionary Algorithms, pp. 80-84, 2000.

[13] X. L. Xie and G. A. Beni, "Validity measure for fuzzy clustering," IEEE Trans. on Pattern Anal. Machine Intell., vol.3, pp. 841-846, 1991.

[14] M. Setnes and U. Kaymak, "Fuzzy modeling of client preference from large data sets: an application to target selection in direct marketing," IEEE Trans. on Fuzzy Systems, vol.9, no.1, pp.153-163, Febuary, 2001.

[15] R. Narasimhan, Goal programming in a fuzzy environment, *Decision Sciences*, Vol. 11, pp. 325-336, 1980.

[16] R. N. Tiwari, S. Dhamar and J. R. Rao, Fuzzy Goal Programming-an additive model, *Fuzzy Sets and Systems*, Vol. 24, pp. 27-34, 1987.

[17] M. Sakawa, K. Kato, and T. Shibano, Fuzzy programming for multiobjective 0-1 programming problems through revised genetic algorithms, European Journal of Operational Research

[18] M. Gen and B. Liu, A genetic algorithm for nonlinear goal programming, *Research Report*, Vol. 23, pp. 141-148, Ashikaga Institute of Technology, 1996.

[19] M. Sugeno and T. Yasukawa, "A fuzzy logic based approach to qualitative modeling," *IEEE Trans. on Fuzzy Systems*, Vol. 1, No. 1, pp. 7-31, Feb. 1993.

[20] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *Journal of Intelligent and Fuzzy Systems*, Vol. 2, No. 3, 1994.

부 록

AFDE3와 AFCDE3를 시험하기 위해 사용된 12개의 함수 최적화 문제는 다음과 같다.

- $f_1(x_i)=\sum_{i=1}^{3} x_i^2$, $-5.12\le x_i\le 5.12$, 최소값은 0.
- $f_2(x_i)=100\cdot(x_1^2-x_2)^2+(1-x_1)^2$, $-2.048\le x_i\le 2.048$, 최소값은 0.
- $f_3(x_i)=25+\sum_{i=1}^{5}\lfloor x_i\rfloor$, $-5.12\le x_i\le 5.12$, 최소값은 0. $\lfloor\cdot\rfloor$는 가까운 정수 출력
- $f_4(x_i)=\sum_{i=1}^{30}(i\cdot x_i^4+rand())$, $-1.28\le x_i\le 1.28$, 최소값은 0. $rand()$는 균일분포의 랜덤변수 값

$$f_5(x_i)=\frac{1}{0.002+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ji})^6}},$$

$-65.536\le x_i\le 65.536$, 최소값은 0.998004.

여기서 $a_{j1}=-32,-16,0,16,32$, $j=0,1,2,3,4$ 이고

$a_{j1}=a_{j(\mathrm{mod})5,0}$

$a_{j2}=-32,-16,0,16,32$, $j=0,5,10,15,20$ 이고

$a_{j2}=a_{j+k}(k=1,2,3,4)$

$$f_6(x_i)=\sum_{i=1}^{4}\frac{0.15(z_i-0.05sgn(z_i))^2\cdot}{d_i\cdot x_i^2}\quad |x_i-z_i|<0.05$$

여기서 $z_i=\lfloor|\frac{x_i}{0.2}|+0.49999\rfloor\cdot sgn(x_i)\cdot 0.2$,

$d_i=1,1000,10,100$, $|x_i|<0.05$

$i=1,2,3,4$ $-1000\le x_i\le 1000$, 최소값은 0.

$f_7(x_i)=\sum_{i=1}^{10}\frac{x_i^2}{4000}-\prod_{10}^{i=1}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$, $-400\le x_i\le 400$, 최소값은 0.

$f_8(x_i)=9-x_1-x_2$, $x_i>0$, $(x_1-3)^2+(x_2-2)^2\le 16$, $x_1x_2\le 14$ 최소값은 0.

$f_9(x_i,z_i)=\sum_{i=0}^{16}x_i\cdot z^i$, $z_i\in[-1,1]$이면 $f_9(x_i,z_i)\in[-1,1]$, $z_i=\pm 1.2$ 이면 $f_9(x_i,z_i)\ge T_{16}(1.2)$ $-1000\le x_i\le 1000$, 최소값은 10558.1450229.

$$f_{10}(x_i)=-20\exp\left[-0.2\sqrt{\frac{1}{n}\cdot\sum_{i=1}^{30}x_i^2}\right]-\exp\left[\frac{1}{n}\sum_{i=1}^{30}\cos(2\Pi x_i)\right]+20+e,\quad -30\le x_i\le 30$$

최소값은 0.

$f_{11}(x_i)=\sum_{i=1}^{30}x_i^2$, $-30\le x_i\le 30$, 최소값은 0.

$f_{12}(x_i)=\sum_{i=1}^{30}\lfloor x_i+0.5\rfloor^2$, $-30\le x_i\le 30$, 최소값은 0.

저 자 소 개

황희수

Heesoo Hwang received his B.S., M.S. and Ph.D. degrees in Electrical Engineering from Yonsei University in 1986, 1988 and 1993, respectively. From 1993 to 2000 he worked for system engineering of Korea High-Speed Rail Development project as a senior and a principal researcher in KRRI and KHSR, respectively. Since 2001 he is an assistant professor in the School of Electrical Engineering at Halla University. His research interests include pattern classification, event prediction, and surveillance using fuzzy logic, neural network, and evolutionary algorithms.