평균변화율 및 유일성을 통한 진화 프로그래밍에서 레비 돌연변이 연산 분석

(Analysis of the Levy Mutation Operations in the Evolutionary Programming using Mean Square Displacement and Distinctness)

이 창 용 †

(Chang-Yong Lee)

요 약 본 논문에서는 진화 프로그래밍에서 레비 확률분포(Levy probability distribution)를 사용한 돌연변이 연산의 유용성을 레비 돌연변이 연산 후의 변수의 평균변화율(mean square displacement) 및 유일성(distinctness) 등을 통하여 분석하였다. 레비 확률분포는 무한의 분산(infinite second moment)을 가지는 확률분포로 쪽거리(fractal)와 연계되어 최근 연구가 활발히 진행되고 있는 확률분포이다. 레비 확률분포를 사용한 레비 돌연변이 연산은 변화가 작은 자손(offspring)뿐만 아니라 기존의 정규분포를 사용한 돌연변이 연산에 비하여 상대적으로 변화가 큰 자손을 생성할 수 있다. 이러한 사실에 기초하여 레비 돌연변이 연산은 보다 넓은 탐색 공간을 효율적으로 조사할 수 있음을 평균 변화율 및 유일성 등의 조사를 통하여 수학적으로 증명하였다. 이를 통하여 진화 프로그래밍에서 레비 확률분포에 기초한 돌연변이 연산이 정규분포를 사용한 돌연변이 연산보다 다변량 함수의 최적화의 경우 일반적으로 효율적인 연산임을 알 수 있었다.

Abstract Abstract In this work, we analyze the Levy mutation operations based on the Levy probability distribution in the evolutionary programming via the mean square displacement and the distinctness. The Levy probability distribution is characterized by an infinite second moment and has been widely studied in conjunction with the fractals. The Levy mutation operators not only generate small varied offspring, but are more likely to generate large varied offspring than the conventional mutation operators. Based on this fact, we prove mathematically, via the mean square displacement and the distinctness, that the Levy mutation operations can explore and exploit a search space more effectively. As a result, one can get better performance with the Levy mutation than the conventional Gaussian mutation for the multi-valued functional optimization problems.

1. 서 론

진화 프로그래밍(EP, evolutionary programming)은 생물학적 진화와 자연선택에 기초한 계산 방법론인 진화 연산(evolutionary computations)의 한 분야로 인공지능을 위한 진화적 접근의 한 방법으로 시작되어[1], 80년대 중반에 D. Fogel에 의하여 최적화 알고리즘의 한 방법론으로 발전하게 되었다[2]. 일반적으로 진화 연산에서 사용되고 있는 연산은 크게 교차(crossover), 선택(selection), 그리고 돌연변이(mutation) 연산 등 3가지로 구분할 수 있다. 진화 연산의 대표적인 알고리즘인 유전자 알고리즘(genetic algorithms)[3]에서는 교차 연산이 주요 연산인 반면, 진화 프로그래밍과 진화 전략(evolutionary strategies)[4]에서는 돌연변이 연산이 주된 연산이다.

기존의 진화 프로그래밍에서 돌연변이 연산은 다음 세대(next generation)의 자손(offspring)을 얻기 위하여 현재 세대(present generation)의 부모(parent) 값에 정규분포(Gaussian distribution)를 따르는 임의 확률변

· 이 논문은 1999년도 한국학술진흥재단의 연구비에 의하여 연구되었음 (KRF-99-003-E00375).

† 정 회 원 : 공주대학교 산업정보학과 교수
clce@kongju.ac.kr

논문접수 : 2000년 11월 22일
심사완료 : 2001년 8월 24일

수(random number)를 더하는 연산이다. 여기서 정규분포를 따르는 임의의 확률변수를 사용하여 돌연변이 연산을 수행하는 주된 이유는 중심극한정리(central limit theorem)에 기인한다고 할 수 있다. 즉, 임의의 유한한 분산을 가지는 확률 분포에서 무작위로 표본(sample)을 추출하는 경우, 이 표본들의 분포는 표본의 개수가 많은 경우 모집단(population)의 확률분포에 상관없이 정규분포로 수렴함으로 돌연변이 연산에서 임의의 확률변수를 생성할 때 정규분포를 사용함은 자연스럽다 하겠다. 그러나 진화 프로그래밍에서 돌연변이 연산을 위하여 반드시 정규분포를 사용해야 하는 것은 아니다. 특히 정규분포를 돌연변이 연산에 적용하는 경우, 돌연변이 연산 후 생성되는 자손의 변화가 크지 않아 탐색 공간이 넓은 문제의 경우 탐색 공간 전체를 효율적으로 조사하지 못함으로 최적해로 수렴하는 속도가 느리며 보다 좋은 최적해를 찾지 못한다는 단점이 있다[5].

진화 프로그래밍에서 정규분포가 아닌 다른 확률분포를 사용한 돌연변이 연산에 대한 연구가 시작되었는데, 대표적인 것으로는 코시 확률분포(Cauchy probability distribution)에 기초한 돌연변이 연산을 진화 프로그래밍과 진화 전략에 적용한 연구[5, 6]와 레비 확률분포(Levy probability distribution)[7]을 사용한 돌연변이 연산에 관한 연구[8] 등이 그것이다. 이 연구들을 통하여 기존의 정규 분포를 사용한 돌연변이 연산의 단점 중 하나인 느린 수렴 속도를 개선하고 보다 향상된 최적해를 구할 수 있었다.

일반적으로 최적화 과정은 고려 대상이 되는 변수 영역을 골고루 넓게 탐색하는 것(exploration)과 특정 영역을 집중적으로 탐색하는 것(exploitation)들을 조화롭게 적용하는 과정으로 생각할 수 있다. 대부분의 최적화 알고리즘은 최적화의 초기 단계에서는 변수 영역을 되도록 넓게 탐색하여 최적해의 근처에 도달한 다음 최적해 근처의 변수 영역을 집중적으로 탐색하는 방법을 취하고 있다. 이것은 최적화 알고리즘 중 하나인 시뮬레이티드 어닐링(simulated annealing)에서도 사용하는 방법으로, "온도"를 점차적으로 낮추면서, 즉 탐색하는 변수의 영역을 줄여가면서 최적 값을 찾아가는 방법에서도 찾아 볼 수 있다. 또한 최근 C.-Y. Lee와 X. Yao[9]의 연구 결과에 의하면 진화 프로그래밍의 초기 단계에서는 돌연변이 연산 후 변수의 변화를 크게 하는 것이 효율적이며, 이 후에는 변수의 변화를 작게 하는 것이 효율적임이 알려졌다. 따라서 최적화에서는 보다 넓은 변수 영역을 탐색하는 것이 중요하며 이것은 최적화의 수렴 속도 및 결과에 직접적으로 영향을 미친다 할 수 있다.

본 연구에서는 위에서 언급한 X. Yao와 Y. Liu의 연구[5, 6]와 레비 확률분포를 진화 프로그래밍의 돌연변이 연산에 적용한 기존의 초기 연구[8]를 기반으로 진화 프로그래밍에서 레비 확률분포 사용의 장점을 이론적으로 규명하고자 한다. 기존의 연구[8]는 돌연변이 연산 후 변수의 변화를 크게 해 주는 레비 확률분포를 진화 프로그래밍에 접목시켜 기존 돌연변이 연산의 단점인 느린 수렴 속도를 개선하고 보다 나은 최적 값을 구할 수 있다는 사실을 여러 함수에 대하여 실행한 실험 결과를 통하여 기술한 것이다.[1] 이러한 초기 연구를 바탕으로 본 연구에서는 수렴 속도 개선 및 최적화 성능 향상은 레비 돌연변이 연산[2]을 통해 보다 넓은 변수 영역을 탐색함으로 가능하다는 것을 수학적으로 증명하고, 또한 레비 돌연변이 연산은 단지 변수의 변화를 크게 해 주는 역할만을 수행하는 것이 아니라, 변수의 변화가 작은 개체들도 생성함으로 일단 전역 최적해 근처에 도달한 상태에서는 변수의 변화가 작은 개체들이 중요한 역할을 수행함을 실험적으로 입증하고자 한다. 특히 넓은 변수 영역 탐색은 단순히 돌연변이 연산 후의 변수의 변화가 크게 함으로만 성취되는 것은 아니며, 생성된 변수들간의 중복이 적어야 가능함으로 본 연구에서는 레비 돌연변이 연산이 이러한 장점을 가지고 있음을 증명하고자 한다.

본 논문은 다음과 같은 순서로 구성되어 있다. 제 2 장에서는 레비 확률분포의 정의에서 출발하여 레비 확률분포의 특성과 레비 돌연변이 연산 후 생성되는 후보 개체들의 선택 과정을 통하여 레비 돌연변이의 장점을 실험적으로 입증하였고, 제 3 장은 레비 확률분포를 사용한 진화 프로그래밍의 장점을 이론적으로 고찰하였으며, 마지막 장은 본 논문의 결론을 맺었다.

2. 레비 돌연변이 연산

진화 프로그래밍에서 주된 연산인 돌연변이 연산은 다음과 같이 개략적으로 표현할 수 있다. 모집단(population)의 어느 한 개체를 x라 표현하면, 다음 세대의 자손 x'은 x에 임의의 확률변수 Y를 더함으로 구할 수 있다. 즉,

---

1) 참고 문헌[8]의 결과는 레비 돌연변이 연산을 몇 개의 벤치마크(benchmark) 함수들의 최적화에 있어 그 우수성을 실험을 통하여 입증한 것이며, 레비 돌연변이 연산이 모든 최적화 문제에서 정규 돌연변이 연산에 비해 우수한 성능을 지닌다고 말할 수는 없다.

2) 진화 프로그래밍에서 레비 확률분포를 사용하여 돌연변이 연산을 수행하는 과정을 레비 돌연변이 연산이라 칭한다.

$$x' \sim x + Y$$

으로 표현된다. 정규분포를 사용한 돌연변이 연산의 경우, Y 는 정규분포를 따르는 확률변수이며, 레비 확률분포를 사용한 돌연변이 연산의 경우, Y 는 레비 확률분포를 따르는 확률변수이다.

레비 확률분포는 코시 확률분포의 일반화 형태로 이 분포의 특징은 분포의 꼬리 부분에서의 확률이 코시 확률분포와 마찬가지로 멱 급수적(power-law)으로 감소한다는 것이다. 멱 급수적 분포는 분포의 확률변수 사이에 특성 축척(characteristic scale)이 존재하지 않는 특징을 가지고 있으며 이는 분산을 특성 축척으로 가지는 정규분포의 경우와 구별된다. 또한 멱 급수적 분포는 쪽거리(fractal) 현상의 특징을 설명하는 기본적인 성질임으로 쪽거리 연구[10, 11]에 많이 사용되는 분포이다.

2.1 레비 확률 분포

유한의 분산(혹은 이차 모멘트)을 가지는 확률변수들이 유한의 분산으로 인하여 특성 축척(characteristic scale)을 가지며 중심극한 정리에 의하여 정규분포로 수렴한다는 연구가 활발히 진행되고 있을 무렵, P. Levy 는 무한의 이차 모멘트(infinite second moment)를 가지는 경우에도 중심극한정리를 만족하는 확률분포에 대한 연구를 수행하였다[7]. 그는 무한의 이차 모멘트를 가지면서 중심극한정리를 만족하는 일련의 확률분포함수를 발견하였는데 이러한 확률분포를 레비 확률분포라 부르며 다음과 같은 식으로 표현된다.

$$P_{\alpha,\gamma}(x) = \frac{1}{\pi}\int_0^\infty e^{-\gamma q^\alpha}\cos(qx)\,dq \qquad (1)$$

위의 식에서 볼 수 있듯이 레비 확률분포는 $x=0$ 을 기준으로 대칭인 분포를 하고 있으며, 두 개의 매개변수(parameters) α, γ 를 가지고 있다. γ 는 축척인수(scaling factor)로 $\gamma>0$를 만족하며 α 는 $0<\alpha<2$ 를 만족하는 실수이다. 위 식에서 볼 수 있듯이 레비 확률분포는 적분 형태를 취하고 있는데 특정한 α 값을 제외한 일반적인 값에 대해서 적분 값이 해석적으로 알려져 있지 않다.[3] 또한 $\alpha \to 2$인 경우, 정규분포로 수렴함이 알려져 있다. 따라서 레비 확률분포를 사용하기 위해서는 적분을 계산하여야 하며, 이것이 레비 확률분포가 가지고 있는 가능성에 비해 연구가 비교적 덜 진행된 이유중 하나이다. 매개변수 α 는 분포의 모양을 결정하는 것으로 α 값에 따라서 분포의 모양이 변하는 특징을 가지고 있다. 즉, α 가 작을수록 분포의 꼬리부분의 확률이 크게된다. 그러나 레비 확률분포의 가장 중요한 특징은 축척에 따른 분포의 변화이다. 즉, 만약 위의 식에서 $x \to bx$ 로 치환하여 모든 변수의 값을 일정한 크기만큼 증가(혹은 감소)시켰다고 가정하자. 그리하면 위의 식 (1)을 사용하면 다음과 같은 관계를 얻을 수 있다.

$$P_{\alpha,\gamma}(bx) = \frac{1}{b}P_{\alpha,\gamma'}(x) \qquad (2)$$

여기서 $\gamma' = \gamma b^{-\alpha}$ 이다. 따라서 $x \to bx$ 로 변환 후에도 확률변수는 같은 레비 확률분포를 하며 분포의 모양에는 변화가 없다. 또한 위의 식 (2)를 이용하면 특별한 γ 에 대한 확률분포를 아는 경우, 일반적인 γ 에 대해서도 분포의 값을 구할 수 있다. 따라서 본 연구에서는 편이상 $\gamma=1$ 으로 가정한다. 특히 레비 확률분포에서 x 값이 큰 경우 아래와 같이 확률분포가 확률변수의 멱 급수(power-law)를 함이 알려져 있다[7].

$$P_{\alpha,1}(x) \propto x^{-(\alpha+1)} \qquad (3)$$

따라서 $\alpha=2$인 경우를 제외하고 이 분포는 무한 분산 혹은 무한 이차 모멘트(infinite second moment)를 가진다는 것을 쉽게 알 수 있다. 그림 1은 α값에 따른 레비 확률분포와 정규분포를 비교한 것이다. 그림 1에서 보는 것처럼 α값이 작을수록 가장 자리의 x값에 대한 확률이 멱 급수적으로 변함을 알 수 있다.

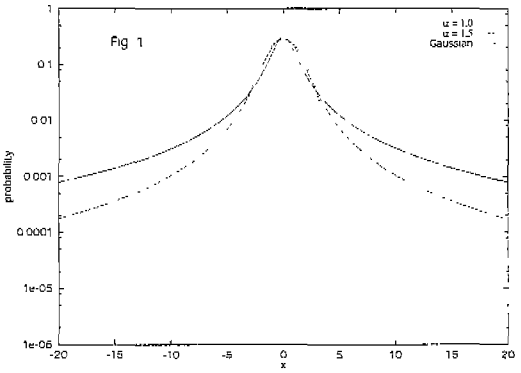

그림 1 정규분포와 레비 확률분포($\alpha=1.0, 1.5$)의 비교. y축(확률)은 로그(log) 축척을 사용하였다.

2.2 최적화 과정에서 레비 돌연변이 연산의 작용

진화 프로그래밍에서 각 개체(individual)는 실수 값을 가지는 다음과 같은 한 쌍의 벡터 $(\vec{x_i}, \vec{\sigma_i})$, $i=1,2,\cdots,\mu$ 로 표시할 수 있는데, 여기서 $\vec{x_i}$는 n개의 성분을 가지는 벡터로 $\vec{x_i} = (x_i(1),\cdots,x_i(n))$으로 표시

3) 예를 들어 $\alpha=1$인 경우, 위의 적분은 해석적으로 적분이 가능하고 그 결과는 코시(Cauchy) 확률분포를 함이 알려져 있다. 따라서 레비 확률분포의 특수한 경우가 코시 확률분포라 할 수 있다.

표 1 최적화에 사용된 함수들. 여기서 n은 변수의 차원을 나타내며 S는 그 변수들의 영역을 나타내며 $S \subseteq R^n$을 만족한다.

Test Functions	N	S
$f_1 = -\sum_{i=1}^{N} x_i \sin(\sqrt{x_i})$	30	$(-500, 500)^N$
$f_2 = \sum_{i=1}^{N} \{x_i^2 - 10\cos(2\pi x_i) + 10\}$	30	$(-5.12, 5.12)^N$
$f_3 = -20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_{[i}^2}\right) - \exp\left\{\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_1)\right\} + 20 + e$	30	$(-32, 32)^N$
$f_4 = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \Pi_{i-1}^{N}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$(-600, 600)^N$

되며, $\overrightarrow{\sigma_i}$ 는 돌연변이 연산 수행 시 환경에 따라 변하게 되는 표준편차로 개체의 각 성분에 대하여 존재하며 $\overrightarrow{\sigma_i} = (\sigma_i(1), \cdots, \sigma_i(n))$으로 나타낼 수 있다. 돌연변이 연산은 아래와 같이 각 개체 $(\overrightarrow{x_i}, \overrightarrow{\sigma_i})$, $i = 1, 2, \cdots, \mu$ 에 대하여 그 자손 $(\overrightarrow{x_i'}, \overrightarrow{\sigma_i'})$, $i = 1, 2, \cdots, \mu$ 을 다음과 같이 생성한다.

$$x_i'(j) = x_i(j) + \sigma_i(j) L_j(\alpha) \quad (4)$$

여기서 $L_j(\alpha)$는 매개변수가 α 인 Levy분포를 하는 임의의 확률변수이며, 각각의 j에 대하여 독립적인 확률변수를 생성함을 나타낸다.[4)]

레비 돌연변이 연산은 정규 돌연변이 연산에 비해 연산 후 변수의 변화가 크다. 따라서 진화의 초기 단계에서는 유용하나, 알고리즘이 전역 최적해 근처에 도달한 후에도 레비 돌연변이 연산을 적용하면 변수의 변화가 큼으로 인하여 오히려 효율성이 떨어지는 경향이 있을 수 있다. 그러나 레비 돌연변이 연산을 진화 프로그래밍에 적용하는 경우, 전역 최적해 근처에서도 정규 돌연변이 연산과 유사한 효율성을 가질 수 있는데 그 이유는 다음과 같다. 진화 프로그래밍에서는 현재 세대의 개체 μ 개 각각에 대하여 돌연변이 연산을 수행한 후 생성되는 μ 개의 새로운 개체를 합한 2μ 개 후보 중에서 적합도(fitness)가 큰 개체가 선택될 확률을 크게 하여 μ 개의 개체를 선택하여 다음 세대를 구성한다. 레비 돌연변이 연산은 정규 돌연변이 연산에 비해 변수의 변화가 큰 개체를 상대적으로 많이 생성하는 것은 사실이나, 레비 돌연변이 연산 내에서만 보면 변수의 변화가 작은 개체가 큰 개체보다 상대적으로 많이 생성된다. 이것은 레비 확률분포의 모양을 통해 알 수 있다. 따라서 레비 돌연변이 연산의 경우 후보 개체들간의 편차는 정규 돌연변이 연산의 후보 개체들 보다 크다. 이것은 정규 돌연변이의 경우에는 후보 개체들을 나타내는 변수들이 공간적으로 서로 뭉쳐져 존재하는 반면, 레비의 경우에는 퍼져있다고 생각할 수 있다. 따라서 레비 돌연변이 연산은 정규 돌연변이 연산에 비해 변수의 변화가 큰 개체를 상대적으로 많이 생성함으로 보다 넓은 영역을 조사할 수 있어서 최적으로의 수렴 속도가 빨라지게 되며, 일단 최적해 근처에 도달한 상태에서는 변수의 변화가 작은 개체들의 적합도가 상대적으로 큼으로 변수의 변화가 작은 개체들이 선택될 확률 역시 크게 되어 최적해로의 수렴 상태를 유지할 수 있다.

위의 사실을 입증하기 위하여 다음과 같은 거리 함수(distance function)를 정의하자.

$$d(g) = \sqrt{(\overrightarrow{x_{g+1}} - \overrightarrow{x_g})^2} = \sqrt{\sum_{k=1}^{n}\{x_{g+1}(k) - x_g(k)\}^2} \quad (5)$$

여기서 $\overrightarrow{x_g}$ 는 g 번째 세대의 개체들 중 적합도가 가장 높은 개체로 n 개의 성분을 가진다. 따라서 주어진 거리 함수는 돌연변이 연산 수행 후 적합도가 가장 높은 개체의 변화를 g 번째와 $g+1$ 번째 세대간의 함수로 나타낸 것이다. 즉, 돌연변이 연산 수행 후 세대에 따른 변수의 변화는 거리 함수를 계산함으로 살펴볼 수 있다.

위에서 정의한 거리 함수를 계산하기 위해 실험에 사용한 함수와 변수들의 영역 및 차원은 표 1과 같다. 일반적으로 최적화 알고리즘은 주로 국소 최적치가 많아 전체 최적 해를 찾기 어려운 문제들을 해결하기 위하여 주로 사용되고 있음으로 본 실험에서도 국소 최적치가 많은 함수들을 사용했으며, 이들은 진화 프로그래밍에서 벤치마크(benchmark)로 사용되는 함수[12]들로 참고문헌 [8]에서 사용한 함수와 같다. 또한 함수들의 최적화 과정에서 사용된 진화 프로그래밍의 매개변수는 참고문헌 [8]에서 사용한 값과 동일하다.

4) 레비 돌연변이 연산을 사용한 진화 프로그래밍 알고리즘은 참고 문헌 [8]에 자세하게 설명되어 있다.

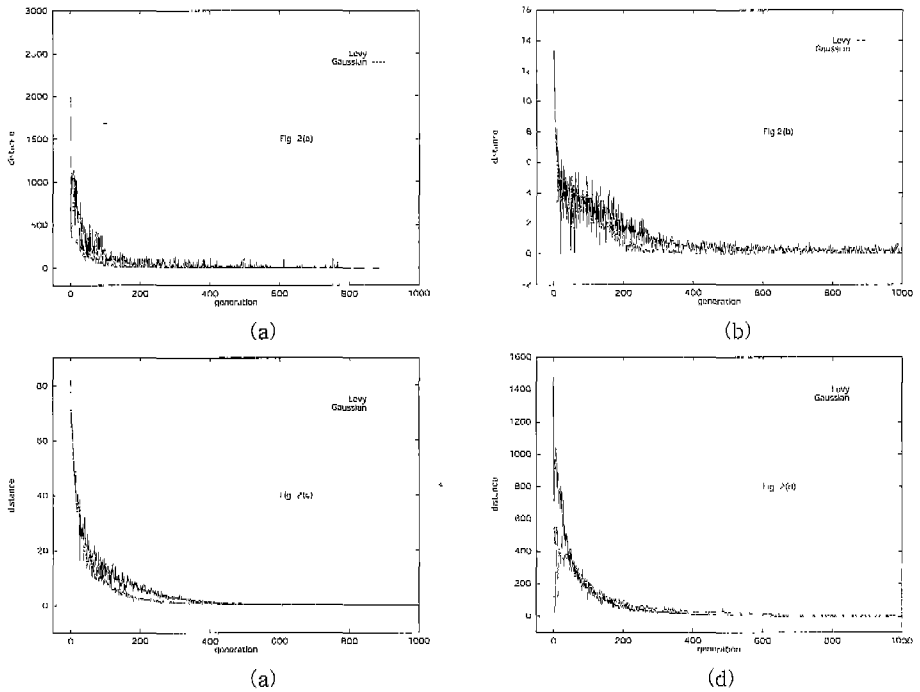

그림 2 표 1의 함수들에 대하여 거리 함수를 구한 결과. 그림 (a), (b), (c), (d)는 각각 표 1의 함수 f_1, f_2, f_3, f_4에 대하여 레비 돌연변이 연산과 정규 돌연변이 연산을 수행하여 거리 함수를 계산한 결과이다.

그림 2는 위의 함수들을 사용하여 구한 거리 함수의 결과이다. 레비 돌연변이 연산의 경우, 고려한 모든 매개변수($\alpha=0.8, 1.0, 1.2, 1.4, 1.8$)에 대하여 유사한 결과를 얻었을 수 있었음으로 그림에서는 하나의 매개변수에 대한 결과만을 정규 돌연변이 연산의 결과와 비교했다. 즉, f_1, f_2 의 경우 $\alpha=0.8$ 을 사용한 결과를, f_3, f_4 의 경우 $\alpha=1.4$ 를 사용한 결과를 나타냈다. 그림 2에서 볼 수 있듯이 두 돌연변이 연산 모두에 대하여 최적화의 초기 단계에서는 변수의 변화가 큰 개체들이 적합도가 좋음을, 최적치 근처에 도달한 후에는 변수의 변화가 적은 개체들이 적합도가 좋음을 알 수 있다. 이러한 사실은 두 돌연변이 연산이 공통으로 최적화 초기에는 보다 넓은 변수 영역을 탐색하고, 이후에는 해당 영역 주변을 집중적으로 탐색하고 있음을 반영한다. 그러나 최적화 초기 단계에서는 레비 돌연변이 연산이 정규 돌연변이 연산에 비해 상대적으로 변수의 변화가 큰 개체들의 적합도가 좋음을 알 수 있고, 일단 최적치 근처에 도달하면 정규 돌연변이 연산과 마찬가지로 세대간 변수들의 변화가 적음을 알 수 있다. 따라서 레비 돌연변이 연산은 정규 돌연변이 연산에 비해 변수의 변화가 큰 개체를 상대적으로 많이 생성함으로 보다 넓은 영역을 조사할 수 있어서 최적으로의 수렴 속도가 빨라진다. 그리고 일단 최적해 근처에 도달한 상태에서는 변수의 변화가 적은 개체들의 적합도가 더 좋음으로 그들이 선택될 확률 역시 높게되어 최적해로의 수렴 상태를 유지할 수 있다. 따라서 레비 돌연변이 연산이 기존의 돌연변이 연산에 비해 상대적으로 효과적임을 입증하기 위해서는 레비 돌연변이 연산이 보다 넓은 변수 영역을 효율적으로 탐색할 수 있다는 것을 증명하면 된다.

3. 평균변화율과 유일성을 통한 레비 돌연변이 연산

제 2 장에서 언급한 것처럼 레비 돌연변이 연산은 정규 돌연변이 연산에 비해 상대적으로 변수의 변화가 큰 개체를 생성할 확률이 높아 보다 넓은 변수 영역을 탐색할 수 있는 가능성이 있다. 특히 넓은 변수 영역을 탐색하기 위해서는 단순히 돌연변이 연산 후의 변수의 변화가 크게 함으로만 성취되는 것은 아니며, 생성된 변수

들간의 중복이 적어야 가능하다. 본 장에서는 레비 돌연변이 연산이 이러한 성질을 만족함을 수학적으로 증명하고자 한다.

3.1 평균 변화율(mean square displacement)

이 절에서는 진화 프로그래밍에서 돌연변이 연산이 어느 정도 탐색 공간을 효율적으로 조사하는가를 알아보기 위하여, 여러 세대를 통하여 돌연변이 연산이 수행된 후의 자손 값을 돌연변이 연산 수행 전의 값과 비교하여 그 차이에 대한 평균값인 평균변화율(mean square displacement)을 계산한다. 이를 위하여 $x(n)$ 을 n 번째 세대의 개체의 값이라 하자. 또한 계산을 보다 간편하기 위하여 자체 적응적인 돌연변이 연산은 임의의 확률변수가 아니라 상수로 가정하자. 위의 가정을 바탕으로 n 세대 후의 개체의 값은 $n-1$ 세대 후의 개체의 값을 사용하여 다음과 같이 표현 가능하다.

$$x(n)=x(n-1)+\beta Y_{n-1} \tag{6}$$

여기서 β 는 돌연변이 연산의 정도를 나타내는 상수이며, Y_{n-1} 은 $n-1$ 세대에서 돌연변이 연산을 위하여 생성된 임의의 확률변수이다. 이 확률변수는 기존의 진화 프로그래밍의 경우 표준정규분포를 따르며, 레비 확률분포를 사용한 진화 프로그래밍인 경우 레비 확률분포를 따른다. 위의 식(6)을 순환적으로 적용하면 n 세대 이후의 개체 값의 변화는 다음과 같은 양으로 표현 가능하다.

$$d(n)\equiv x(n)-x(0)=\beta\sum_{k=0}^{n-1}Y_k \tag{7}$$

따라서 평균변화율은 $d(n)^2$ 의 평균 제곱근을 구한 것으로 $\sqrt{\langle d(n)^2\rangle}$ 으로 표현할 수 있다. 여기서 $\langle\cdots\rangle$ 는 기대치를 나타낸다.

정규분포를 사용한 진화 프로그래밍의 경우 각 Y_k 는 표준정규분포를 따르는 확률 변수임으로 그들의 합인 $\sum_{k=0}^{n-1}Y_k$ 는 평균이 0이고 분산이 n 인 정규분포를 따른다. 따라서 평균변화율은 다음과 같이 주어진다.

$$\sqrt{\langle d(n)^2\rangle}=\beta\sqrt{n}\propto\sqrt{n} \tag{8}$$

이것은 통계역학의 브라운 운동과 유사한 형태인데 위의 식 (8)에서 n 세대 후의 정규분포를 사용한 돌연변이 연산을 통한 개체의 변화는 $\sqrt{n}$ 에 비례함을 알 수 있다. 즉 n 번의 돌연변이 연산을 수행한 후의 개체의 값 $x(n)$ 과 초기 개체 값 $x(0)$ 의 차이는 평균적으로 $\sqrt{n}$ 에 비례한다. 따라서 개체 값의 변화를 크게하기 위해서는 세대 수 n 을 증가시켜야 함을 알 수 있다.

레비 확률분포를 사용한 돌연변이 연산의 경우의 평균변화율은 레비 확률분포가 안정적 분포(stable distribution)라는 특징을 이용한다. 즉 n개의 확률변수 $Y_0,\cdots,Y_{n-1}$이 각각 평균값이 0이고 매개변수가 α 인 레비 확률분포를 가질 때, 이 확률 변수의 선형결합인 $Z=Y_0+\cdots+Y_{n-1}=\sum_{k=0}^{n-1}Y_k$도 역시 평균이 0이고 매개변수가 α인 레비 확률분포를 가진다. 또한 레비 확률분포는 변수의 값이 큰 영역에서 다음과 같은 멱급수 분포를 함으로 $Z=\sum_{k=0}^{n-1}Y_k$ 의 확률분포는 아래와 같다.

$$P_\alpha(z)\sim\frac{1}{z^{\alpha+1}}\ ,\ \text{if } |z|\gg 1 \text{ and } 0<\alpha<2 \tag{9}$$

위의 성질을 이용하여 임의의 레비 확률분포를 따르는 확률변수가 있을 때 그 변수 제곱의 평균, 즉 분산을 구해 보면 다음과 같다.

$$\langle Z^2\rangle\sim\int_{-\infty}^{\infty}\frac{z^2}{z^{\alpha+1}}dz=\infty \quad \text{for} \quad 0<\alpha<2 \tag{10}$$

따라서 n 세대후의 개체 값의 평균 변화율은 다음과 같이 주어진다.

$$\sqrt{\langle d(n)^2\rangle}\propto\langle Z^2\rangle=\infty \tag{11}$$

따라서 레비 확률분포를 사용한 돌연변이 연산의 경우, 개체의 평균 변화율은 돌연변이 연산 횟수, 즉 세대에 무관하며 무한의 크기를 가짐[5]을 알 수 있다. 그러므로 레비 확률변수를 사용한 돌연변이 연산을 수행하면 정규분포를 사용한 돌연변이 연산에 비하여 변화가 많은 개체를 생성할 수 있다. 이것은 정규분포를 사용한 돌연변이 연산의 경우, 개체 값의 변화가 $\sqrt{n}$ 에 비례하는 것과 대조를 이룬다.

예를 들어 개체가 2개의 성분을 가지는 2차원 변수로 표현되는 경우, 정규분포를 사용한 돌연변이 연산과 레비 확률분포를 사용한 돌연변이 연산 사이의 변수 공간을 탐색한 정도를 나타내기 위하여 다음을 고려하자. 즉 개체 x 를 2개의 성분으로 구성된 벡터 $x=(x_1,x_2)$ 로 생각하면, n 세대 돌연변이 연산 후의 각 성분은 다음과 같이 주어진다.

$$x_i(n)=x_i(n-1)+\beta Y\ ,\ i=1,2 \tag{12}$$

그림 3은 Y 를 정규분포와 레비 확률분포를 따르는 경우로 구분하여 각 경우에 대하여 1000번의 돌연변이 연산을 수행한 후 $(x_1(n),x_2(n))$, $n=1,2,\cdots,1000$ 의 값을 2차원 상에 나타낸 것이다. 그림 3을 통해 볼 수 있듯이 레비 확률분포를 사용한 돌연변이 연산의 경우가 훨씬 넓은 영역의 변수 공간을 탐색할 수 있음을 알 수 있다.

---

5) 레비 확률분포는 이론적으로는 무한의 분산을 가지나 컴퓨터 상에서 구현할 경우 무한의 값을 가지지 못하며 단지 무한에 어느 정도 근접하는 값을 생성한다.

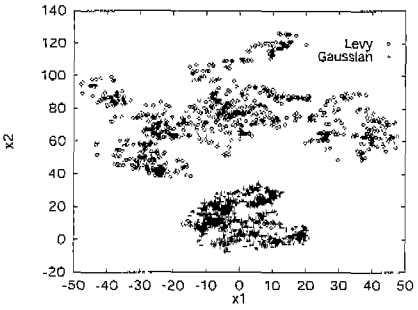

그림 3 2차원 변수들에 대한 정규분포를 사용한 돌연변이 및 레비 확률분포($\alpha=1.5$)를 사용한 돌연변이 연산의 결과. 2차원 변수를 원점에서 출발하여 1000번의 돌연변이 연산을 각각 수행하여 그 결과를 그림으로 표현한 것이다.

정규분포를 사용한 돌연변이 연산과 달리 레비 확률분포를 사용한 돌연변이 연산의 경우, 세대가 진행됨에 따라 개체 값의 변화를 크게 할 수 있음으로 보다 넓은 조사 영역을 탐색할 수 있다는 장점이 있다. 결과적으로 국소 최적치가 많은 문제, 특히 국소 최적치들 사이가 높은 적합도 장벽으로 이루어진 문제의 경우, 개체 값의 변화를 크게 하는 알고리즘을 사용하지 않는다면 국소 최적치간의 적합도 장벽을 넘기 어렵게 되고, 따라서 전체 최적치를 찾아가기 어렵다. 따라서 레비 확률분포를 사용한 돌연변이 연산을 적용하면 국소 최적치가 많은 문제에서 전체 최적치를 찾는 어려움을 어느 정도 해결할 수 있다고 하겠다.

3.2 돌연변이 연산에 의해 생성된 상태의 고유성 (number of distinct states obtained by mutation operations)

진화 프로그래밍에서 각 세대를 통하여 생성된 자손 개체는 부모 개체의 값에 돌연변이 연산의 결과인 임의의 확률변수 값을 더하여 생성된다. 최적화 문제에서 변수 공간을 탐색하고자 할 때 중복되지 않는 효율적인 탐색이 필요하다. 즉 확률변수를 통하여 새로운 값을 생성할 때 중복되지 않는 값이 생성되어야 새로운 변수 영역을 탐색할 수 있다. 그러나 진화 프로그래밍에서는 특정한 확률분포를 가지는 확률변수를 생성하여 변수 공간을 탐색함으로 생성된 자손의 값이 중복될 가능성이 있으며 이것은 돌연변이 연산에 사용되는 확률분포(정규분포 혹은 레비 확률분포)에 따라 중복 정도가 결정된다. 이 절에서는 n 세대를 통하여 생성된 자손 중에서 중복된 자손의 개수를 평균적으로 계산하고자 한다. 즉 돌연변이 연산을 위하여 정규분포 혹은 레비 확률분포를 사용하느냐에 따라 임의의 확률변수가 어느 정도 중복된 값을 생성하는가를 조사한다.

이를 위하여 D_n 을 n 세대의 돌연변이 연산을 수행하여 생성한 자손 값 중 중복되지 않은 값의 개수라 하자. D_n 에 대한 일반적인 통계량을 구하기 위해서는 D_n 의 확률분포를 알아야 하는데 D_n 의 확률분포를 구하는 것은 특별한 경우를 제외하고는 간단하지 않다. 그러나 D_n 의 특별한 통계량 중 하나인 D_n 의 평균값을 구하는 것은 D_n 의 확률분포에 대한 정보가 없어도 가능하다. 즉 D_n 의 평균, $\langle D_n \rangle$ 은 각 세대에서 중복되지 않은 새로운 값을 생성하는 확률을 사용하여 그 합의 함수로 표현 가능하다. 즉, Δ_k 를 k 번째 세대에서 중복되지 않은 새로운 값이 생성될 확률이라 하면 각 세대에서 임의의 확률변수를 생성한 결과는 새로운 값을 얻거나 그렇지 않거나 2 가지중 하나이다. 따라서 만약 새로운 값을 생성한 경우 새로운 값을 생성한 개수는 1이 되고, 그렇지 않으면 0이다. 그러므로 n 세대 후의 새로운 값을 얻는 양의 평균은 다음과 같이 주어진다.

$$\langle D_n \rangle = \sum_{k=1}^{n} \{1 \cdot \Delta_k + 0 \cdot (1-\Delta_k)\} = \sum_{k=1}^{n} \Delta_k \qquad (13)$$

여기서 n 세대 후의 새로운 값의 개수에 대한 평균 $\langle D_n \rangle$을 각 세대에서 구한 평균의 합으로 표현하였다.

위의 통계량 $\langle D_n \rangle$ 은 Δ_k 의 확률분포에 따라 다르게 되는데, 여기서는 레비 확률분포와 정규분포의 경우를 비교한다. 첫 번째, 유한의 분산을 가지는 정규분포의 경우, 위의 양은 다음과 같은 값을 가짐을 계산할 수 있다.[6]

$$\langle D_n \rangle \sim \sigma \sqrt{\frac{8n}{\pi}} \propto n^{1/2} \qquad (14)$$

여기서 σ 는 분포의 표준편차에 해당한다. 또한 무한의 분산을 가지는 레비 확률분포인 경우, 레비 확률분포의 매개변수인 α 값에 따라 다음과 같이 주어진다.

$$\begin{aligned} \langle D_n \rangle &\propto n, \quad \text{for } 0<\alpha<1 \\ &\propto \frac{n}{\ln n}, \quad \text{for } \alpha=1 \\ &\propto n^{1/\alpha}, \quad \text{for } 1<\alpha<2 \end{aligned} \qquad (15)$$

위의 결과에서 볼 수 있듯이 정규분포의 경우 세대의 수가 증가함에 따라 중복되지 않는 확률변수의 생성은 제곱근에 비례하여 증가함으로 세대수가 증가하는 속도보다 중복되지 않는 확률변수의 생성 횟수가 제곱근에

6) 유한의 분산 및 무한의 분산에 대한 $\langle D_n \rangle$계산에 필요한 과정은 참고 문헌[13]에 자세히 설명되어 있음으로 이 논문에서는 생략하기로 한다.

반비례한다. 그러나 레비 확률분포를 사용한 경우, 특히 $0<\alpha<1$ 에서는 중복되지 않는 확률변수의 생성 속도와 세대 증가 속도가 같은 비율로 증가한다. 위의 두 결과를 통해서 볼 때, 비록 두 확률분포 모두에서 중복되지 않은 값의 평균치가 세대에 의존함을 알 수 있으나 레비 확률분포를 사용한 경우가 정규분포를 사용한 경우에 비하여 모든 매개변수의 값에 상관없이 중복되지 않은 새로운 값의 평균치가 더욱 큼을 알 수 있다. 따라서 레비 확률분포를 사용한 돌연변이 연산이 더욱 많은 새로운 변수 값을 생성할 수 있기 때문에 보다 많은 개수의 변수들을 조사할 수 있다는 것을 알 수 있다.

4. 결 론

본 논문에서는 레비 확률분포를 사용한 돌연변이 연산에 기초한 진화 프로그래밍을 기존의 정규분포를 사용한 진화 프로그래밍과 비교하여 레비 확률분포의 사용에 대한 장점을 살펴보았다. 특히 돌연변이 연산을 적용한 결과로 생성되는 변수의 평균 변화율 및 새로운 변수 생성의 평균 수량 등을 수학적으로 고찰하여 레비 확률분포에 기초한 돌연변이 연산의 우수성을 입증하였다. 변수의 평균 변화율의 경우, 레비 확률분포를 사용한 돌연변이 연산이 정규분포를 사용한 경우보다 변화가 큰 변수를 생성할 수 있어서 보다 넓은 변수 영역을 탐색할 수 있었다. 또한 레비 확률변수를 돌연변이 연산에 적용하면 세대가 증가함에 따라 정규분포의 경우에 비하여 덜 중복된 새로운 값들이 생성됨을 알 수 있었다. 이러한 측면에서 볼 때 돌연변이 연산을 위하여 레비 확률분포를 사용함이 더욱 효율적임을 이론적으로 뒷받침 해 준다 할 수 있다.

레비 확률변수에 기초한 돌연변이 연산은 레비 확률분포의 매개변수인 α 값에 따른 성능은 함수에 따라 차이가 있어 일률적인 결론을 내기는 어렵다. 그러나 일반적으로 공학적 응용 분야에서 요구되는 최적화 문제는 국소 최적치가 많은 문제임으로 이러한 문제는 보다 넓은 변수 영역을 효율적으로 탐색하는 것이 필요하며, 따라서 레비 확률분포를 사용한 진화 프로그래밍이 더욱 효율적임을 알 수 있다.

레비 확률분포를 사용한 진화 프로그래밍은 앞으로 많은 연구를 필요로 하고 있는데 특히 다음의 세 가지 측면은 언급할 만하다. 첫째, 본 연구에서는 레비 확률분포의 매개변수인 α 값이 가질 수 있는 모든 영역을 조사하지 못하고 단지 $0.8<\alpha<2.0$ 인 영역에서만 적용하였는데 이것은 $0.0<\alpha<0.8$ 인 영역에서는 레비 확률분포를 가지는 확률변수를 생성하는 빠른 알고리즘이 아직 개발되지 않았기 때문임으로 이 영역에 대한 연구가 필요하다. 둘째, 기존의 정규분포를 사용한 진화 프로그래밍 알고리즘에서는 환경에 적응하는 표준편차를 사용하는데, 주로 정규 분포를 사용한다. 그러나 환경적응적 표준편차에도 레비 확률분포를 적용할 수 있을 것으로 생각되며 앞으로 이 방향의 연구도 필요하다 하겠다. 마지막으로 본 연구에서는 레비 확률분포 축척인수 γ 혹은 매개변수 α 를 세대에 따라 변화시키지 않았는데 이것 역시 환경에 적응하는 축척인수 혹은 매개변수를 사용하는 진화 프로그래밍이 보다 자연스러운 연산이라 생각된다.

본 논문은 레비 확률분포를 사용한 진화 프로그래밍의 장점을 살펴보기 위하여 기존의 연구에 더하여 보다 이론적인 기반을 마련하기 위한 시도임으로, 앞으로 보다 많은 함수들의 최적화와 공학적 최적화 문제[14]에 적용하여 레비 돌연변이 연산의 특성에 대한 연구가 더욱 필요하다고 생각된다.

참 고 문 헌

[1] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial Intelligence Through Simulated Evolution," John Wiley & Sons, New York, NY, 1966.

[2] D. Fogel, "Evolutionary Computation : Towards a New Philosophy of Machine Intelligence," IEEE Press, NY, 1995.

[3] D. Goldberg, "Genetic algorithm in search, optimization and machine learning," Addison Wesley, 1989.

[4] T. Back and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization." Evolutionary Computation Vol. 1, No. 1, pp. 1-23, 1993.

[5] X. Yao and Y. Lin, "Fast evolutionary programming," in Evolutionary Programming V: Proceedings of the Fifth Annual Conference on Evolutionary Programming, MIT Press Cambridge, MA, 1996.

[6] X. Yao and Y. Lin, "Fast Evolutionary Strategies," in Evolutionary Programming VI: Proceedings of the Sixth Annual Conference on Evolutionary Programming, pp151-161, Springer, 1997.

[7] P. Levy, "Theorie de l'Addition des Veriables Aleatoires," Gauthier-Villars, Paris, 1937 ; B. Gnedenko and A. Kolmogorov, "Limit distributions for Sums of Independent Random Variables," Addition-Wesley, Cambridge, MA, 1954.

[8] 이창용, "Levy 확률 분포를 사용한 빠른 진화 프로그

래밍", 정보과학회논문지(B), 25권, 1호, pp.141-149, 1998.
[9] C.-Y. Lee and X. Yao, "Evolutionary algorithm with adaptive Levy mutations," Proceedings of CEC2001, pp. 568-575, 2001.
[10] B. B. Mandelbrot, "The Fractal Geometry of Nature," Freeman, San Francisco, 1982.
[11] R. Mantegna and E. Stanley, "Scaling behavior in the dynamics of an economic index," Nature, Vol. 376, No. 6, pp. 46-49, 1995.
[12] D. Whitley, K. Mathias, R. Rana, and J. Dzubera, "Building better test functions," Proceedings of th.e 6th International Conference on Genetic Algorithms, L. Eshelman, ed. pp. 239-246, 1995.
[13] A. Bunde and S. Halvin (ed), "Fractals in Science," chapter 5, Springer-Verlag, 1994.
[14] P. J. Angeline, G. M. Saunders, and J. B. Pollack, "An Evolutionary Algorithm that Constructs Recurrent Neural Networks," IEEE Trans. Neural Networks, Vol. 5:1, pp. 54-65, 1994.

이 창 용

1983년 서울대학교 계산통계학과 졸업(이학사). 1995년 미국 텍사스 주립대학교 물리학과 졸업(이학박사). 1996년 ~ 1998년 한국전자통신연구원 선임연구원. 1998 ~ 현재 공주대학교 산업정보학과 조교수. 관심분야는 진화연산을 사용한 최적화 문제, 복잡계(Complex systems), 정보이론 등