Proceedings of KFIS Autumn Conference 2007 Vol. 17, No. 2.

점진형 유전프로그래밍과 거리기반형 진화연산자

Steady State Genetic Programming and Distance based Genetic Operator

방철혁, 서기성

서울시 성북구 서경대학교 전자공학과

E-mail: {brilliant, ksseo}@skuniv.ac.kr

요 약

유전프로그래밍(GP)은 GA, ES, 그리고 EA등에 비해 구조의 복잡함으로 인해 상대적으로 진화방식 및 진화연산자에 대한 연구가 미진한 실정이다. 본 논문에서는 유전프로그래밍에 대한 점진형 진화 방식과 트리 깊이 및 부모간의 거리를 기반으로 한 새로운 진화연산자를 제안한다. 이항식 벤치마크 문제에 대하여 실험을 수행하였고, 세대형 진화 방식 및 기존 연산자와의 성능을 비교하였다.

Key Words : Steady State Genetic Programming, Distance based Genetic Operator, Binomial-3 Problem

1. 서 론

유전프로그래밍(Genetic Programming, GP)은 진화 연산(Evolutionary Computation)의 한 부류로서 개체의 표현에 가변적인 크기의 트리를 사용한다. 이 때 트리에서의 각 노드는 하나의 함수를 나타내며, 각 개체는 함수의 집합, 즉 컴퓨터 프로그램으로 구성된다.[1]

GP는 염색체 표현으로 트리를 이용하므로, 상대적으로 이진스트링으로 표현되는 유전알고리즘(Genetic Algorithm, GA)이나 실수 최적화에 널리 쓰이고 있는 진화전략(Evolutionary Strategy)[2] 및 진화알고리즘(Evolutionary Algorithm, EA)[3]에 비해서, 복잡한 구조를 가지고 있고, 실수나 비트스트링과는 달리 트리 개체의 공간 분포 특성을 파악하기 어렵기 때문에 진화방식 및 진화 연산자에 대한 연구가 아직까지 많이 시도되지 않고 있다.

진화전략(ES)과 진화 알고리즘(EA)은 GA나GP 등의 세대형(generational)과는 달리 점진형(steady state) 진화방식 - 부모집단의 부분 교체방식 - 을 가지고 있다.

최근에 GP 에 진화전략(ES)에 의한 진화방식을 적용한 연구가[4] 소개되었으며, 부분적인 성능 향상을 보여주고 있다.

GP 에 세대형(generational) 진화방식과 점진형(steady state) 진화방식중 어느것이 더 적합하고 효율적인지, 또는 대상 문제마다 다른지등에 대해서는 아직까지 체계적으로 실험되고 조사된 바가 거의 없다.

단지 특정 문제에 대해서 GP의 세대형 방식과 점진형 진화방식을 비교한 연구가 최근에 나왔으며, 이 문제에 국한하여 점진형이 좀 더 강인함을 보인다고 언급하고 있을 뿐이다.[5]

진화방식보다 일반적으로 성능에 더 영향을 미칠수 있는 진화 연산자도 GP의 경우에는 아직까지 다양한 연구가 시도되지 않고 있다.

기존의 유전프로그래밍이 가지는 진화연산자에서 교배나 돌연변이의 연산 위치는 트리 노드중 임의로 선택되고 있다. 단지 함수노드와 터미널 노드만을 분리하거나, 형 제한(strong type) 의 경우 노드의 종류를 구분하고 있는 실정이다. 또한, 재조합 연산자인 교배에 참여하는 부모의 갯수가 두 개인 전통적인 방식만이 시도되고 있다.

따라서 본 연구에서는 개체가 가지는 형질중의 몇 가지 특성을 이용하는 GP 에 대한 새로운 진화연산자를 제시하고자 한다.

새로운 진화연산자는 개체의 적합도와 트리 구조가 가지는 형태 및 깊이 그리고 교배가 이루어지는 개체간의 거리를 이용하였다.

3차 이항식 문제에 대해서, 제안된 점진형 진화방식과 진화 연산자를 조합한 방법을 기존의 GP 와 비교 실험하였다.

2. 점진형 유전 프로그래밍

진화방식에서 기존의 유전 프로그래밍은 세대형(generational) 알고리즘을 취하고 있다.

이러한 방식은 모든 개체들이 매세대마다 교체되며, 유전연산에 의해서 적합도가 저하되지 않는 자손들을 생성해야하는 더 큰 압력이 존재한다. 실제 GP에서 코드의 성장(code growth)을 제한시킬 경우, 세대형 방식보다 점진형 진화방식이 해의 탐색이 더 효율적임을 보이고 있다[4].

점진형 진화방식은 주로 진화전략(ES)과 진화 알고리즘(EA)에서 사용되고 있다. ES 에서는 μ 개의 부모로부터 λ 개의 자손을 생성하고, 다음세대의 개체를 선택하는 방법으로 (μ+λ) 또는 (μ,λ) 방식을 이용한다[3].

EA 는 ES 방식과 유사하나 유전연산에 참여한 부모와 자손에 의해서 대치되는 부모가 각각 다르게 선택되는 특징이 있다. 본 연구에서는 실수최적화 문제에서 우수한 성능을 G3 모델[3] 방식을 사용한다. 구체적인 알고리즘은 다음 표 1과 같다.

G3 알고리즘은 기존의 세대형 진화방식에서는 잘 사용되어지지 않던 랜덤선택을 이용함으로써 적합도가 낮은 개체에 대해서도 균등한 기회를 부여하였고, 더불어 최우수 개체를 포함시킴으로서 진화 속도를 빠르게 하는 효과도 동시에 가지고 있는 구조이다.

본 논문에서는 기존의 세대형 GP 프로그램 진화 방식 구조를 G3 알고리즘을 기반으로한 점진형 진화방식으로 수정하였고 GP 트리에 대한 새로운 진화연산자를 구현하였다.

3. 새로운 GP 진화 연산자

3.1 깊이 차등 돌연변이

기존 돌연변이 연산자는 트리의 임의의 지점에서 수행되고, 임의로 생성된 서브트리로 대체된다.

표 1. G3 알고리즘

G3(Generalized Generation Gap)
1. 집단 P 로부터 best 부모를 선택하고, 나머지 μ-1개의 부모를 랜덤으로 선택 2. μ 개의 부모로부터 λ 개의 자손을 recombination 방법으로 생성 3. 집단 P 로부터 2 개의 부모를 랜덤으로 선택 4. 선택된 2 개의 부모와 λ 개의 자손의 합집합에서, 2 개의 best 개체를 뽑아서 3번에서 선택된 2 개의 부모를 대치

따라서, 주로 개체 집단에 새로운 개체를 유입시켜, 지역수렴에서 벗어날 수 있는 기회를 제공하는 것이 주 용도이다.

본 연구에서는 트리의 깊이와 적합도등의 특성을 고려한 깊이 차등 돌연변이 연산자를 구현하였다.(그림 1) 서브트리의 깊이에 따라 룰렛휠 방식으로 서브트리를 선택하고, 이를 다시 적합도에 따라 돌연변이 연산을 수행할 노드를 결정한다. 자세한 알고리즘은 표 2 에 정리되어 있다.

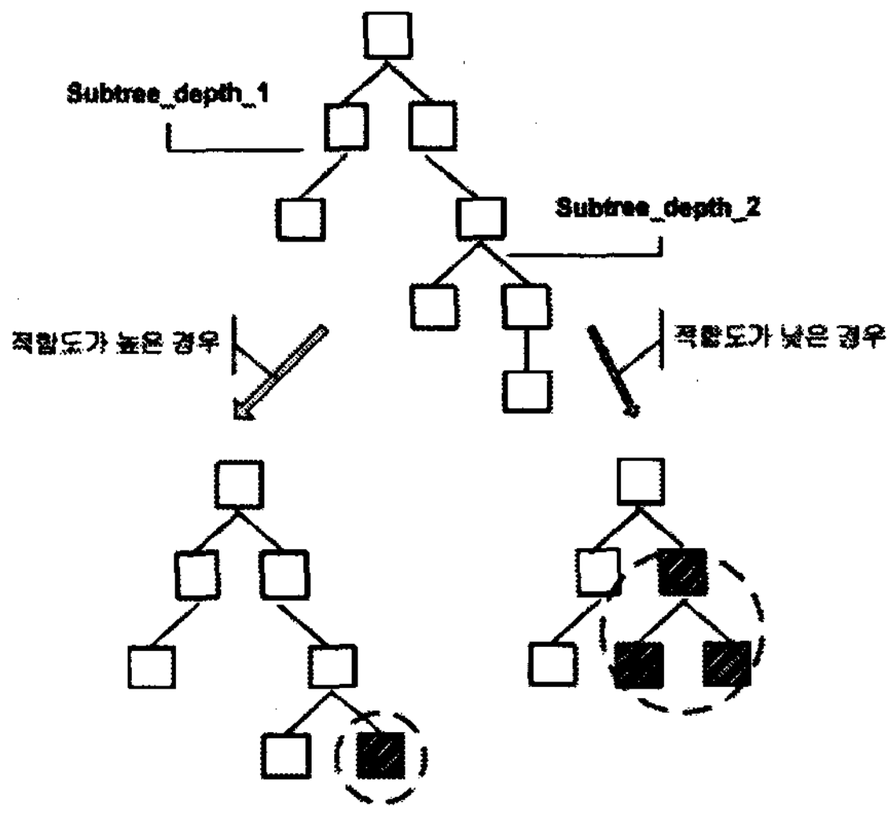

그림 1. 깊이 차등 돌연변이

즉, 점진형 진화에서는 부분적인 개체 교체를 통해 진화를 수행하는 방식이기 때문에 임의성에 의한 연산보다는 방향성을 가지는 연산이 진화의 효율 향상에 도움이 된다고 기대할 수 있다.

표 2 깊이 차등 돌연변이

깊이 차등 돌연변이
1. 선택된 개체에 대하여 각 가지의 서브트리의 깊이를 측정 2. 깊이가 큰 쪽을 우선으로 룰렛휠 선택에 따라 연산자를 수행할 서브트리를 결정 3. 개체의 적합도에 따라 연산자 수행 노드결정 if 적합도가 높을 경우 루트노드에 가까운 노드 else 적합도가 낮은 경우 터미널에 가까운 노드 4. 결정된 노드에서 연산자 수행

3.2 거리기반 교배연산자

실수 염색체의 재조합 방식중 하나인 PCX 연산자를[5] 트리구조에 맞도록 변화시켜 적용하였다.(그림 2)

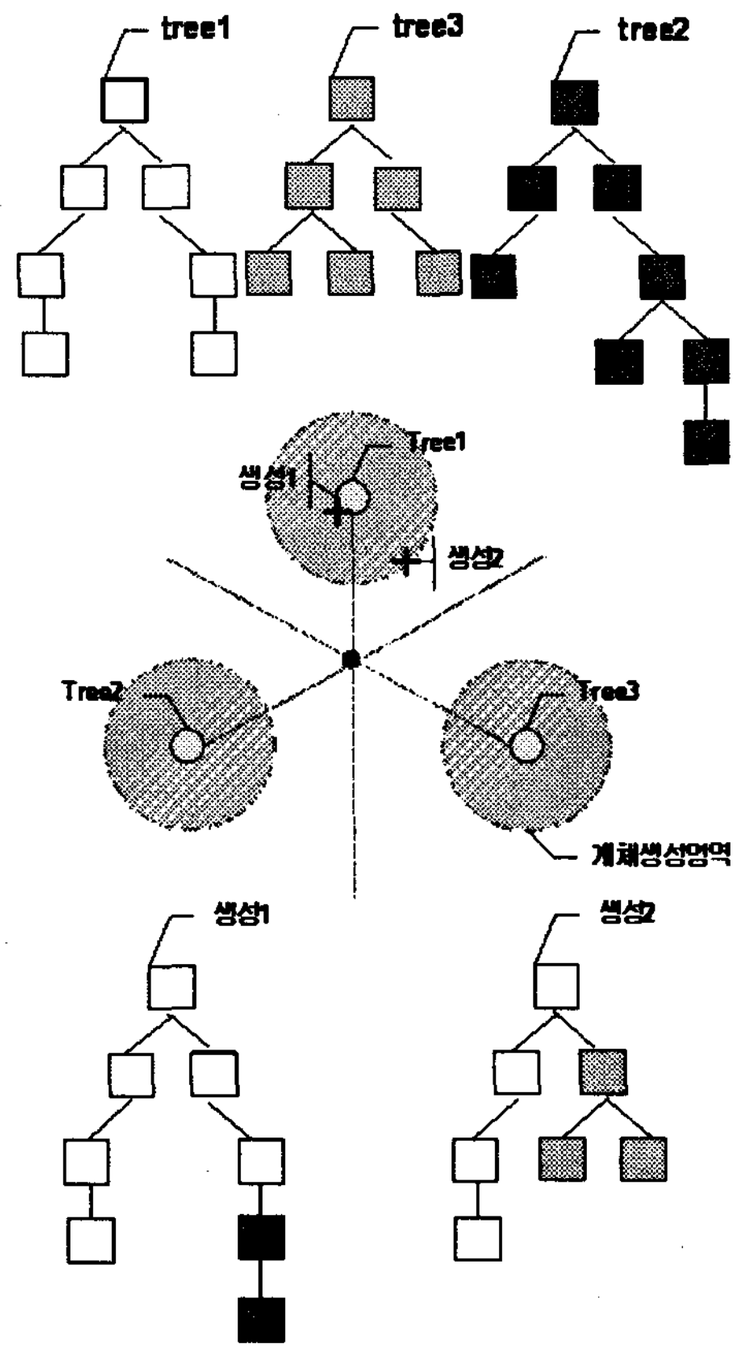

그림 2. 거리기반 교배연산자

실수 파라미터에 적용된 PCX 연산자는 전체의 군집에서 각 부모를 중심으로 자손을 생성하며, 부모간의 중심, 임의의 부모와의 거리, 방향 벡터의 성분등을 이용한다. 실수 공간과 GP 트리의 공간은 서로 다르므로, PCX 연산자의 기본 아이디어를 트리구조의 특성을 고려하여 수정 구현한다.

표 3. 거리기반 교배연산자 알고리즘

거리기반 교배연산자
1. 세 개의 개체선택 2. 생성하고자 하는 개체의 위치 결정 3. 생성위치에 따라 두 개의 개체 선택 4. 선택된 개체의 적합도를 기준으로 연산자 수행 노드 결정 (깊이 차등 돌연변이 연산자와 동일) 5. 결정된 노드에서 연산자 수행

그림 2에 설명된 바와 같이, 복수개의 부모 개체를 가상적인 공간상에 배치하고, 생성하고자 하는 개체의 위치를 결정한다. 이때 위치는 어떤 부모에 가까운 자손 개체를 형성하고자 하는지를 의미한다. 그림에서 생성된 자손 1은(왼쪽 아래) 부모 1 을 중심하여 생성된 것이며, 부모 2의 방향이기 때문에, 부모 2의 서브 트리를 가져와 터미널 노드에 가깝게 교배연산이 수행된 것을 보여준다. 우측의 생성 자손 2 는 부모 중심에서 더 멀기 때문에 교배연산이 터미널보다는 루트 쪽에 가깝게 수행되었음을 나타낸다. 표 3 에 거리기반 교배연산자의 알고리즘이 정리되어 있다.

4. 실험

4.1 실험 문제

대상 실험문제로 식 (1)의 3차 이항식 (binomial-3)에 대한 회귀분석을 선택하였다.

$$f(x) = 1 + 3x + 3x^2 + x^3 \qquad (1)$$

이 문제의 정의 및 실험 방법의 설계는 Daida[6]에 의해 제안되어졌고, 문제의 특성이 GP 의 다른 문제에서 공통적으로 나타나는 많은 특성을 공유하고 있다. 특히, 해의 탐색 조정이 어렵다고(tunably difficult) 알려진 문제이다.

이 문제는 유전프로그래밍의 탐색 성능의 판단 기준으로써 사용되어져 왔으며, 문제를 구성하는 실수의 생성범위를 조절함으로써 문제의 난이도를 상향 혹은 하향 조절할 수 있는 스케일을 가진 문제이다.

이항식에 대한 적합도는 주어진 구간에서 등간격으로 50개의 지점 값들을 비교하여 오차 0.01 범위내에 든 갯수로 정의된다.

함수집합(function set)의 구성은 {+,-,×,÷}의 네 가지이며, 터미널집합(terminal set)은 {X,R}의 두 가지로 구성된다. 터미널 집합의 X는 방정식의 x와 동일한 변수이며, R은 문제의 난이도를 결정짓는 일정 범위에서 무작위로 생성되는 실수이다.

실수 R은 [-a,a]의 범위에서 생성되며, 범위내에서 균등분포곡선에 따라 발생하게 된다. ERC의 형성에 따라 적용되는 문제의 난이도에 대한 제시된 알고리즘의 성능비교를 위해 3가지의 발생범위를 지정하였으며, ERC의 범위는 [-a,a] 로 구성되며, 그 경계값 a는 {0, 0.1, 1}의 세 가지 조건을 두었다. 이 때, a가 0인 경우는 ERC가 존재하지 않는 경우이다.

4.2 실험 방법 및 결과

제시된 진화연산자의 성능비교를 위해 기존의 연산자와 새로이 제시된 연산자를 조합한 6가지 방법에 대해서, ERC의 범위에 따라 다시 세 가지 경우로 나누어 실험하고 그 결과를 비교하였다. (표 4, 5)

프로그램 코드는 lil-gp[7]를 수정하여 구현하였다. 에 알고리즘 및 연산자 조합에 의해서

구성된 6 가지 실험 방법이 나와 있다.

표 4. 알고리즘 및 연산자 조합에 따른 실험 방법

실험방법	알고리즘 조합
1	일반 GP에서 돌연변이만 사용
2	1 + 교배
3	점진형 GP + 돌연변이
4	점진형 GP + 깊이차등돌연변이
5	3 + 교배
6	4 + 거리기반교배

표 5. 데이터 셋의 실수 구간 범위

data set 1	ERCs
N	—
1	[-0.1 , 0.1]
2	[-1 , 1]

적합도는 비교한 50개의 좌표의 절대값 오차를 이용하였으며, '0'이 되면 완전히 해를 찾은 것이다. 실험은 각각에 대해여 10회씩 반복하였으며 평균 결과가 표 6 에 나와 있다.

표 6. 알고리즘 조합에 따른 3차 이항식 실험결과

실험 방법	data set 1	적합도	일치 좌표수	연산횟수
1	N	0.4974	39	60611
	1	0.7598	26.56	137277
	2	1.8966	17.22	150000
2	N	0	50	2611
	1	0.1023	49.22	24433
	2	0.0987	46.89	46866
3	N	0.2475	45.78	34351
	1	1.4839	32.22	149535
	2	1.3376	33	139126
4	N	0.4005	45.56	15749
	1	1.4832	32.11	147269
	2	1.3319	32.78	139475
5	N	0.0304	49.88	49109
	1	1.3890	39.78	138732
	2	0.5885	42.89	169410
6	N	0.4431	49.56	41811
	1	1.8802	35.00	177000
	2	2.1134	37.78	178130

지금까지의 결과는 세대형 방식과 점진형 방식의 차이가 뚜렷하지 않고, 아직까지 새로이 제안된 연산자의 개선효과가 많이 나타나고 있지 않지만, 조합방법에 대한 실험 결과를 토대로 진화 방식과 연산자를 개선하면 충분한 탐색성능의 향상을 기대할수 있다고 사료된다.

5. 결 론

다른 진화연산 분야에 비해 상대적으로 진화 방식 및 진화연산자에 관한 연구가 미진한 유전프로그래밍(GP)에 대해서 점진형 진화 방식과 트리 깊이 및 부모간의 거리를 기반으로 한 새로운 진화연산자를 제안하였다. 이항식 문제에 대하여 제안된 방법과 기존의 방법에 대한 여러가지 조합의 실험을 수행하였고 성능 결과를 비교하였다. 앞으로 다른 문제에 대한 더 많은 실험과 분석을 통해 제안된 방법을 개선하면 GP의 진화방식과 진화연산자 연구에 대해 기여할수 있을것으로 기대한다.

참 고 문 헌

[1] J. Koza, "Genetic Programming: On the Programming of Computers by Means of Natural Selection", MIT press, 1992

[2] N. Hansen, S. Müller, P. Koumoutsakos, "Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evolutionary Computation, 11(1):1-18, 2003

[3] K. Deb, A. Anand, D. Joshi, "A Computationally Efficient Evolutionary Algorithm for Real Parameter Optimization", 2002 by the massachusetts Institute of Technology. Evolutionary Computation 10(4), pages 371-395

[4] E. O. Costa, "A Pozo. A New Approach to Genetic Programming based on Evolution Strategies", 2006 IEEE International Conference on Systems, pages 4832-4837

[5] T. Soule, J. Jones "Comparing Genetic Robustness in Generational vs. Steady State Evolutionary Algorithms. GECCO'06 ACM 1-59593-186-4, pages 143-149

[6] J. M. Daida, R. R. Bertram. S. A. Stanhope, J. C. Khoo, S. A. Chaudhary, O. A. Chaudhri, and J. A. Rolito 2, "What makes a problem gp-hard? analysis of a tunably difficult problem in genetic programming", *Genetic Programming and Evolvable Machines*, 2:165-191, June 2001.

[7] D. Zongker B. Punch, *Lil-GP User's Manual*. Michigan State University, July 1995.