인공 진화에 의한 학습 및 최적화

장 병 탁

건국대학교 컴퓨터공학과

1. 서 론

1859년에 출간된 찰스 다윈(Charles Darwin)의 종의 기원(The Origin of Species)은 자연 선택을 통한 진화의 원리를 기술하고 있는데 그 내용은 다음과 같이 요약될 수 있다.

- 생태계에 존재하는 개체들은 그 특성을 후손에게 전해주는 경향이 있다.
- 그럼에도 불구하고 자연은 상이한 특성을 지닌 개체를 생성한다.
- 환경에 잘 적응한 개체들은 그렇지 못한 개체들보다 더 많은 후손을 생성하는 경향이 있다.
- 변화가 축적되면 특정 생태계에 적합한 특성을 지닌 전혀 새로운 종이 생성될 수 있다.

현재로서도 진화가 진행되는 메커니즘이 완전히 규명된 것은 아니지만, 적어도 분자 생물학적인 관점에서 자연 선택은 염색체의 교환과 돌연변이로부터 발생하는 변화에 의해서 이루어진다는 것은 잘 알려진 사실 중의 하나이다. 염색체 교환은 기존의 유전자를 새로운 조합으로 결합시킨다. 돌연변이는 이전에 염색체에 포함되지 않았던 새로운 유전자를 생성한다.

유전 알고리즘(genetic algorithms) 또는 진화 알고리즘(evolutionary algorithms)은 이러한 자연 세계의 진화 과정을 컴퓨터상에서 시뮬레이션 함으로써 복잡한 실세계의 문제를 해결하고자 하는 계산 모델이다. 진화 알고리즘은 특히 적응적 탐색과 학습 및 최적화를 통한 공학적인 문제의 해결에 많이 응용된 바 있으며, 최근 들어 특히 신경망과 퍼지 로직과의 결합으로 그 응용 범위는 점점 늘어나고 있다.

진화 알고리즘은 염색체를 표현하는 방법과 사용되는 유전 연산자의 종류 및 특성에 따라서 다시 여러 가지 모델로 구분된다. 그 하나는 흔히 GA라고 하는 협의의 유전 알고리즘(genetic algorithm)으로서 여기서는 보통 고정된 길이의 이진 스트링을 염색체로 사용한다「Holland 1975, Goldberg 1989」.이에 반해 진화 전략(evolution strategy, ES)은 실수의 값을 취하는 유전자들로 구성된 벡터를 염색체로 사용한다「Rechenberg 1973, Baeck & Schwefel 1993」. 그 밖에도 그래프와 트리를 염색체 표현에 사용하는 진화 프로그래밍(evolutionary programming, EP)과 유전 프로그래밍(genetic programming, GP) 등이 있다「Fogel 1966, Fogel 1995, Koza 1992」. 진화적 탐색에 사용되는 연산자로 EP와 ES는 돌연변이(mutation), GA와 GP는 교차(crossover) 연산자를 주로 사용한다. 역사적으로 볼 때 EP, ES, GA는 1960년대와 70년대에 개발되었으며 GP는 90년대에 들어와 연구되기 시작한 분야이다.

본고에서는 진화 계산의 동작 원리와 이론적 기반에 대해 살펴봄으로써 그 원리를 이해하고 앞으로의 응용 가능성에 대하여 고찰하고자 한다. 이를 위해 먼저 대부분의 진화 알고리즘에 공통되는 기본 구성 요소와 계산 절차를 기술하고, 진화 알고리즘을 이용하여 특정 문제를 풀고자 할 때 고려할 사항에 대하여 기술한다. 다음에는 간단한 응용 문제를 예로 들어 이 문제에 진화 알고리즘을 적용하고 그 동작 과정을 추적함으로써 실제 적용에 있어서의 여러 가지 결정 사항과 그 수행 과정을 구체적으로 살펴본다. 또한 진화 알고리즘의 이론적 배경을 이해하기 위해 스키마와 빌딩 블록 그리고 스키마 정리에 대해서 알아본다. 마지막으로 진화 계산 방식과 다른 지능적 계산 기술들과의 융합 가능성의 예로서, 유전 프로그래밍에 의한 신경망 구조의 설계 및 학

습에 대하여 살펴본다.

2. 진화 알고리즘의 기본 구성

진화 알고리즘은 자연 선택과 유전학에 기반한 탐색 방법이다. 진화 알고리즘은 풀고자 하는 문제에 대한 가능한 해들을 염색체(chromosomes)로 표현한 다음 이들을 점차적으로 변형함으로써 점점 더 좋은 해들을 생성한다. 각각의 가능한 해를 하나의 개체(individual)로 보며 이들의 집합을 개체군(population)이라 한다. 하나의 개체는 보통 한 개 또는 여러 개의 염색체로 구성되며 염색체를 변형하는 연산자들을 유전 연산자(genetic operators)라 한다. 진화 알고리즘이 기존의 탐색 방법과 구별되는 점 중의 하나는 점(point)에 의한 탐색이 아니라 군(population)에 의한 탐색이라는 것이다. 진화적 탐색에 있어서는 임의의 값들로 초기화된 가능한 해들의 집합 즉 탐색군으로 시작하여 세대 교체를 반복함으로써 점점 더 우수한 해의 집합들을 생성하려 시도한다. 진화 알고리즘 모델에 따라 약간의 차이는 있으나 전체적인 알고리즘의 흐름은 다음과 같이 요약될 수 있다.

1. M개의 개체를 가진 최초의 개체군 P(0)를 형성한다. 세대 g=0.
2. 개체군 P(0)에 있는 모든 개체에 대해 적합도를 평가한다.
3. 원하는 수준의 해를 가진 개체가 발견되었으면 수행을 멈춘다.
4. P(g)로부터 적합도에 기반해 부모 개체를 선택하고 이를 교차나 돌연변이와 같은 유전 연산자에 의해 변형함으로써 새로운 M개의 개체를 생성한다.
5. 새로운 개체군을 P(g+1)라 하고 세대의 수를 하나 증가한다(g=g+1).
6. 앞의 2단계로 가서 위의 과정을 반복한다.

진화 알고리즘을 이용하여 어떤 문제에 대한 해를 찾기 위해서는 먼저 두 가지의 준비 작업이 필요하다. 하나는 풀고자 하는 문제에 대한 가능한 해를 염색체의 형태로 표현(encoding)하는 것이다. 또 다른 하나는 각 염색체가 문제를 해결하는데 얼마나 좋은지를 측정하기 위한 평가 함수 즉 적합 함수(fitness function)를 결정하는 것이다.

2.1 염색체 표현 방법과 적합 함수

문제에 대한 해를 염색체로 표현하는 기법은 문제마다 다르고 또한 적용하는 진화 알고리즘의 종류마다 다르다. 이 절과 다음절에서는 논술의 편의상 각각 정수 값과 이진 치의 스트링으로 구성된 염색체만을 예로 사용하기로 한다. 그 외에도 앞에서 언급된 바와 같이 k-ary 코드, 실수의 스트링, 트리, 그래프 등 여러 가지 염색체 표현법이 있다.

각 개체의 적합도를 평가하는 적합 함수는 보통 풀고자 하는 최적화 문제의 목적 함수(objective function)나 다른 주관적 평가 함수에 기반하여 결정된다. 목적 함수의 값의 범위는 문제마다 다르기 때문에 보통 정해진 구간 사이의 양수의 값을 갖도록 표준화된 값을 적합도로 사용한다. 엄격히 구별하자면, 표준화하기 전의 적합도의 값을 raw fitness라고 하며 표준화되어서 실제로 개체 선택의 기준이 되는 함수를 적합 함수라고 한다. 표준화하는 한 가지 방법은 목적 함수의 값을 다음과 같이 선형적으로 재조정하는 것이다.

$$f' = af + b$$

여기서 f는 raw fitness를 나타내고 f'는 표준화된 적합 함수의 값을 나타낸다. 상수 a와 b는 알고리즘 수행의 처음부터 끝까지 고정된 값을 가질 수도 있고 매 세대마다 재조정될 수도 있다.

2.2 개체 선택법

유전 알고리즘은 기본적으로 선택(selection), 교차(crossover), 돌연변이(mutation) 연산자로 구성되어 있다. 선택 연산자는 진화 알고리즘에서 적자 생존의 원리를 구현한 것이다. 선택의 기반이 되는 것은 적합 함수이며, 여러 가지 선택 방법들이 존재하지만 그 기본 원리는 더 좋은 개체들에게 특권을 부여한다는 것에 있어서 공통적이다. 일반적으로 널리 사용되는 선택법으로서는 다음의 세 가지를 예로 들 수 있다.

1. 비례 선택법(proportionate selection) : 개체 i가 $f_i/\sum_j f_j$의 확률로 적합도 값에 비례하여 선택된다.
2. 순위 선택법(ranking selection) : 적합도 값 자체가 아니라 적합도 f_i에 기반한 순위에 따라 선택된다.
3. 토너먼트 선택법(tournament selection) : 무작위로 선발된 k개의 개체 중 가장 적합도가 높은 개체가 선택된다.

비례 선택법은 가장 우수한 개체도 우연에 의해 다음 세대에 선택되지 않을 수 있고 또 가장 열등한 개체라고 자식을 복제할 수 있는 가능성을 전혀 배제하지는 않는 보수적인 선택 방법이다. 이에 반해서 순위 선택법은 최상위의 개체는 항상 다음 세대에 유전자를 전달하고 최하위의 개체는 도태되는 비교적 강한 선택 메커니즘이다. 이들 두 방법은 개체군 전체가 한꺼번에 교체되는 "generational 진화"에 사용되는데 반해 토너먼트 선택법은 일부의 개체만의 개체군에서 교체되는 "steady-state 진화" 방식에 채택된다.

2.3 교차 연산자

만약 선택 연산자만이 사용된다면 유전 알고리즘은 초기화된 개체군내에 존재하는 개체 외에는 새로운 개체를 생성할 수 없다. 다른 구조를 갖는 개체들을 탐구하기 위한 연산자로서 교차 연산자가 사용된다. 교차는 세 단계로 수행된다. 먼저 선택 연산자에 의해 개체군에서 두 개의 부모 개체가 선택된다. 다음으로, 이 둘이 염색체 교환을 일으키기 위한 교차 위치가 무작위로 결정된다. 마지막으로, 교차 위치를 기준으로 두 염색체간에 유전자 값이 교환되어 새로운 두 개의 자식 개체가 생성된다. 예를 들어, 선택된 두 개의 개체가

$$i_1=(11111111),\ i_2=(00000000)$$

인 경우 교차 위치로 $x=5$가 선택되었다면 이로부터 생성되는 두 개의 자식 개체는

$$j_1=(11111000),\ j_2=(00000111)$$

가 된다.

교차는 이 연산자가 적용되는 확률을 나타내는 파라메터인 p_c에 의하여 그 적용 빈도 수가 제어된다. 이 외에도 여러 가지 교차 연산자가 제안되어 있으나 가장 중요한 것은 필요한 부분 염색체들이 적합도를 증진하는 방향으로 교환되어야 한다는 것이다.

2.4 돌연변이 연산자

돌연변이 연산자는, 이진 스트링을 염색체로 사용하는 진화 알고리즘의 경우, 각각의 비트에 대해 돌연변이 확률 p_m에 따라 0은 1로 1은 0으로 변경하는 연산자이다. 스트링에 대한 돌연변이는 스트링에 있는 각각의 비트에 대한 돌연변이를 독립적으로 p_m의 확률로 적용함으로써 수행된다.

예로서, 다음 스트링에

$$j_1=(11111000)$$

이 연산자를 적용하면 두 번째 비트가 돌연변이를 일으킴으로써

$$j_{1'}=(10111000)$$

가 될 수 있다. 이 경우 8개의 비트 중 1개의 비트만이 돌연변이를 일으켰으며 effective 돌연변이율은 1/8이다. 돌연변이 연산자는 개체군의 다양성을 유지하는데 중요한 역할을 한다. 만약 개체군에 존재하는 모든 염색체들의 i번째 유전자가 모두 같은 값만을 자진 경우, 염색체의 교차만에 의해서는 이 위치에 새로운 유전 형질을 부여할 수 없으나 돌연변이 연산자는 이것이 가능하도록 해 준다.

3. 진화 알고리즘의 동작 원리

여기서는 간단한 응용 문제를 예로 들어 이 문제에 실제로 진화 알고리즘을 적용하는데에 있어서 고려할 점들을 기술한다. 또한 계산의 수행 과정을 추적함으로써 그 작동 원리를 이해하고자 하는데 도움을 주고자 한다.

3.1 제과업자의 문제

한 제과업자가 직면한 문제를 생각해 보자(이 예는 「Winston 1992」로부터 변형된 것임). 이 제과업자는 가장 양질의 과자를 만들 수 있는 밀가루와 설탕의 최적 배합을 찾으려 한다. 이 문제는 그림 1에 표시된 바와 같은 이차원 공간상에서의 탐색 문제로 볼 수 있다. 여기서 x축은 밀가루의 무게를 y축은 설탕의 무게를 나타내며 생산되는 과자의 품질은 이 두 재료에 대한 함수로 나타난다. 이 문제의 경우 물론 제과업자가 모든 가능한 조합을 다 시험해 보더라도 81가지의 가능성만이 존재한다. 그러나 조합할 수 있는 경우의 수가 증가하거나 재료의 수가 2개에서 3개로 한 단계 증가한다면 탐색 공간의 크기는 점점 커져서 맹목적인 탐색 방법으로서는 더 이상 해결하기 어렵게 된다.

설탕 \ 밀가루	1	2	3	4	5	6	7	8	9
9	1	2	3	4	5	4	3	2	1
8	2	3	4	5	6	5	4	3	2
7	3	4	5	6	7	6	5	4	3
6	4	5	6	7	8	7	6	5	4
5	5	6	7	8	9	8	7	6	5
4	4	5	6	7	8	7	6	5	4
3	3	4	5	6	7	6	5	4	3
2	2	3	4	5	6	5	4	3	2
1	1	2	3	4	5	4	3	2	1

그림 1. 밀가루와 설탕의 결합에 대한 제과의 질.

3.2 진화 알고리즘의 적용

이 제과업자의 문제를 진화 알고리즘을 사용하여 풀기 위

해서는 앞 절에서 살펴본 바와 같이 먼저 다음과 같은 결정을 내려야 한다.

- 개체의 표현 방법
- 적합 함수의 정의
- 개체 선택 방법
- 염색체 교환 방법
- 돌연변이 연산자

각각의 개체가 하나의 특별한 제과법을 나타내도록 표현하도록 한다. 즉 하나의 개체는 (2, 7)과 같은 형태의 염색체 모양을 하며 여기서 첫 번째 유전자는 사용할 밀가루의 양을 그리고 두 번째 유전자는 설탕의 양을 표시하는 1에서 9까지의 숫자로 이루어진다.

적합 함수 f_i는 i번째 개체에 의해 만들어진 과자의 품질 q_i에 비례하도록 다음과 같이 정의된다.

$$f_i=\frac{q_i}{\sum_i q_i}$$

염색체가 표현하는 제과법에 의해 만들어진 과자의 품질이 1에서 9까지의 정수 값을 갖는데 반해, 개체 i의 적합도 값은 0에서 1 사이의 실수 값을 갖도록 조정되었다. 염색체 교환을 위해서는 염색체에 있는 두 개의 유전자 사이를 잘라 분리한 다음 두 부분을 다른 염색체의 각각의 염색체 조각과 교차시킨다. 예를 들어, 두 개의 부모 염색체 (6,9)와 (2,5)의 교차에 의해 새로운 자식 염색체 (6,5)와 (2,9)가 다음과 같이 생성된다.

염색체	q_i	교 차	염색체	q_i
(6,9)	4		(6,5)	8
(2,5)	6	==>	(2,9)	2

여기서 자식 염색체 (6,5)의 품질 또는 성능은 증가하였고 다른 자식 염색체 (2,9)의 질은 감소하였다.

이 문제의 경우 돌연변이 연산자는 염색체의 두 유전자 중 하나를 무작위로 선택하여 1을 더하거나 뺌으로써 무작위 변형을 시키고 그 값이 1에서 9까지의 범위 내에 머물도록 정의한다. 다음의 예와 같이, 돌연변이에 의해 (6,5)가 (5,5)로 변형된다면 이 제과법에 의한 과자의 품질이 즉 개체의 성능이 8에서 9로 향상된다.

염색체	q_i	교 차	염색체	q_i
(6,5)	8	==>	(5,5)	9

또한 진화 알고리즘을 실제적으로 적용하는데 있어서는 다음과 같은 알고리즘 제어 파라메터의 값을 정해야 한다.

- 개체군의 크기 M을 얼마로 할 것인가?
- 돌연변이율 p_m의 값을 얼마로 할 것인가?
- 염색체의 교차율 p_c를 얼마로 할 것인가?

만약 개체군의 크기 M이 너무 작다면 모든 염색체들은 곧바로 동일한 유전자 값을 지니게 될 것이고 그러면 염색체의 교환에 의해 새로운 개체들은 생성할 수 없게 된다. 반대로 개체들의 수가 너무 많다면 불필요한 계산 시간을 많이 사용하게 된다. 돌연변이의 발생 확률 p_m이 너무 작다면 새로운 특성들은 개체군내에 너무 느리게 받아들여지게 되고, 반대로 너무 크면 각 세대들은 이전 세대들과 관련이 없게 된다. 너무 작은 교차율 p_c의 값을 사용하면 개체간에 좋은 형질이 교환될 기회가 부족하여 좋은 해에 빨리 도달하기 어렵게 된다. 반면 염색체의 교환이 너무 빈번히 행해지면 좋은 해를 파괴할 확률이 증가한다. 가장 좋은 해를 빨리 찾을 수 있는 파라메터의 값은 문제의 성격과 다른 제어 파라메터들의 값에 따라 다르다. 아래의 시뮬레이션에서는 다음과 같은 파라메터 값을 사용하기로 한다.

$$M=4,\ p_m=0.2,\ p_c=0.9$$

3.3 알고리즘 수행 과정

알고리즘의 동작을 이해하기 위하여 4개의 개체를 가진 제 0세대의 개체군을 다음과 같이 초기화되었다고 하자.

$$P(0)=\{(1,8),\ (2,3),\ (1,1),\ (9,1)\}$$

이들 염색체가 나타내는 제과법은 각각 2, 4, 1, 1의 품질을 가진 과자를 제조하며 그 총합은 8이다. 이로부터 표준화된 적합도의 값이 표 1에 제시되어 있다.

적합도 비례 선택에 의해 염색체가 먼저 복제되는데 복제 횟수 r_i는 개체군의 평균 적합도 값 $\bar{f}$에 대한 개체 i의 적합도 값 f_i에 의해 결정된다.

$$r_i \propto \frac{f_i}{\bar{f}}$$

이 예의 경우 개체들이 각각 1, 2, 0, 1 개의 복제를 하였다면 그 결과는 표 1에서 새로운 개체군 P′(0)와 같다. 여기에 (1,8)과 (2,3) 그리고 (2,3)과 (9,1) 사이에 각각 염색체 교차가 일어나면 결과는 P″(0)가 된다. 다시 염색체 (1, 3)이 돌연변이를 일으켜 (1,4)로 변형되면 최종적으로 (1, 4), (2,8), (2,1), (9,3)의 개체를 갖는 개체군 P‴(0)이 생성된다.

표 1. 제 0세대에서 제 1세대로의 진화 과정의 추적.

$P(0)$	q_i	f_i	$f_i/\bar{f}$	r_i	$P'(0)$	cross	$P''(0)$	mut	$P'''(0)$	q_i
(1,8)	2	2/8	1	1	(1,8)	x1	(1,3)	mut	(1,4)	4
(2,3)	4	4/8	2	2	(2,3) (2,3)	x1 x2	(2,8)	no mut	(2,8)	3
(1,1)	1	1/8	0.5	0			(2,1)	no mut	(2,1)	2
(9,1)	1	1/8	0.5	1	(9,1)	x2	(9,3)	no mut	(9,3)	3
sum	8	1.00		4						12
avg	2	0.25								

이제 $P'''(0)$은 다음 세대 즉 제 1 세대의 개체군 $P(1)$으로 초기화된다.

$$P(1)=\{(1,4), (2,8), (2,1), (9,3)\}$$

$P(1)$의 개체들은 각각 4, 3, 2, 3의 품질을 가지며 그 총합은 12로서 이는 제 0 세대에서의 개체들의 질에 대한 총합 8보다 향상된 값이다. $P(1)$은 다시 $P(0)$에서와 같은 선택 복제, 염색체 교환, 돌연변이 연산자의 적용에 의해 $P'(1)$, $P''(1)$ 등으로 변형되며 그 결과들이 표 2에 제시되어 있다.

표 2. 제 1세대에서 제 2세대로의 진화 과정의 추적.

$P(1)$	q_i	f_i	$f_i/\bar{f}$	r_i	$P'(1)$	cross	$P''(1)$	mut	$P'''(1)$	q_i
(1,4)	4	4/12	1.3	2	(1,4) (1,4)	x1 x2	(1,1)	mut	(1,2)	2
(2,8)	3	3/12	1.0	0			(2,4)	no mut	(2,4)	5
(2,1)	2	2/12	0.7	1	(2,1)	x1	(1,3)	no mut	(1,3)	3
(9,3)	3	3/12	1.0	1	(9,3)	x2	(9,4)	mut	(8,4)	5
sum	12	1.00		4						15
avg	3	0.25								

최종 결과로서는 $P'''(1)$가 얻어지며 이는 다시 제 2세대의 개체군으로 초기화된다.

$$P(2)=\{(1,2), (2,4), (1,3), (8,4)\}$$

$P(2)$의 각 개체들은 2, 5, 3, 5의 품질을 가지며 그 총합 15는 바로 전 세대의 개체군에서의 값보다 3만큼 증가한 것이다. 이와 같이 세대 교체를 반복하며 (5,5)의 결합을 가진 염색체가 발견되면 이 문제에 대한 최적의 해를 찾은 것으로서 진화를 멈추게 된다.

4. 진화 계산 이론

실제 응용에 있어서의 많은 성공에도 불구하고 진화 알고리즘의 이론적인 연구에 대한 발전은 상대적으로 느린편이다. 한 가지 이유는, 진화 계산이 비교적 단순한 연산자들로 구성되어 있으나 이들이 확률적으로 적용되기 때문에 그 결합 효과를 분석하기가 그리 쉬운 일이 아니기 때문이다. 그러나 몇몇 단순한 진화 알고리즘 모델에 대해서는 어느정도 이론적인 결과가 정립되어 있다. 여기에서는 협의의 유전 알고리즘 즉 GA의 이론적 기반이 되는 스키마 정리에 대하여 고찰해 보기로 한다. 이를 위해 먼저 스키마와 빌딩 블록에 대한 개념을 알아보기로 한다.

4.1 스키마와 빌딩 블록

스키마(schema)는 정해진 스트링 위치에 같은 비트 값을 가진 모든 가능한 스트링들의 부분집합이다. 스키마는 기호 0 또는 1의 어떤 값도 취할 수 있는 don't care 기호 *를 사용하여 표시한다. 가령 4개의 비트를 가진 스트링에서 스키마 $s=1**0$은 첫 번째 유전자 위치에 1의 값을 그리고 4번 째 유전자의 위치에 0의 값을 가지는 모든 스트링들을 나타낸다.

$$s=1**0=\{1000, 1010, 1100, 1110\}$$

스키마에 의해 표현되는 각각의 스트링들은 스키마의 인스턴스라 한다. 스키마에서 0 또는 1의 값을 가진 유전자의 위치를 고정 위치(fixed position)라 한다. 고정 위치의 개수는 그 스키마의 오더(order)로 정의한다. 그리고 스트링

의 가장 왼쪽 고정 위치와 가장 오른쪽 고정 위치 사이의 거리를 스키마의 길이(defining length)라 한다. 위 예의 스키마 $s=1**0$은 오더가 2이고 길이가 3이다.

스키마도 각각의 스트링처럼 적합도를 가지며, 스키마의 적합도는 이를 구성하는 각각의 요소 스트링들의 적합도의 평균값이다. 즉 g세대에 있어서 스키마 s의 적합도 $f(s, g)$는 다음과 같이 정의된다.

$$f(s, g)=\frac{\sum f(i, g)}{N(s, g)}$$

여기서 Is는 스키마 s에 속하는 개체들의 인덱스 집합이고 $f(i, g)$는 개체 i의 적합도이다. $N(s, g)$는 스키마 s의 인스턴스 개수이다.

진화 계산 이론에 있어서 또 하나 중요한 개념은 소위 빌딩 블록(building blocks)이다. 빌딩 블록은 스키마 중에서 적합도가 높고 스키마의 길이가 짧은 것을 가리킨다. 유전 알고리즘에 의한 최적의 스트링을 탐색하는 과정은 여러 스키마들간의 경쟁 과정으로 볼 수 있으며 결국 최적의 스트링을 인스턴스로 가진 스키마 s가 다른 스키마들에 비해 더욱 많은 자식을 복제하고 선택 과정에서 살아남음으로써 개체군에서 s의 인스턴스의 개수가 늘어나는 과정으로 볼 수 있다. 그런데 만약 최적의 스트링이 높은 적합도의 짧은 스키마들 즉 빌딩 블록들을 일렬로 나열하여 얻을 수 있다면, 개개의 빌딩 블록들이 경쟁 관계에서 이겨내기만 한다면 이들의 결합으로 최적의 스트링을 구축할 수 있다. 이와 같이 높은 적합도를 가진 스트링들이 높은 적합도를 가진 빌딩 블록들을 찾아내어 이들을 결합함으로써 만들어질 수 있다는 전제를 빌딩 블록 가설(building blocks hypothesis)이라고 한다.

4.2 스키마 정리

유전 연산자의 적용에 의해 스키마의 인스턴스의 개수는 변한다. 먼저 복제(reproduction) 연산자에 의한 스키마 인스턴스 개수의 변화를 살펴본 후 교차와 돌연변이의 영향을 분석한다.

분석의 편의상 복제되는 자식의 개수가 부모 개체의 적합도에 정비례한다고 가정한다. 즉 i번째 스트링에 의해 정확히 $f(i, g)/\bar{f}(g)$개 만큼의 스트링이 복제된다고 하자. 세대 g에 있어서의 스키마 s의 인스턴스의 개수를 $N(s, g)$로 표기하면 다음 세대 $g+1$에서의 s의 인스턴스의 개수는 다음과 같다.

$$N(s, g+1)=\sum_{i\in I_s}\frac{f(i, g)}{\bar{f}(g)}=\frac{\sum_{i\in I_s} f(i, g)}{\bar{f}(g)}$$

여기서 Is는 스키마 s의 인스턴스에 대한 인덱스 집합이다. $f(i, g)$는 인스턴스 i의 세대 g에서의 적합도이고, $\bar{f}(g)$는 g세대 개체군의 평균 적합도이다. 또한 $f(s, g)=\frac{\sum_{i\in I_s} f(i, g)}{N(s, g)}$로부터 다음의 식이 성립된다.

$$\sum_{i\in I_s} f(i, g)=N(s, g)\, f(s, g)$$

따라서 선택만에 의한 스키마의 인스턴스의 개수는 세대 교체를 함에 따라 다음과 같이 변경된다.

$$N(s, g+1)=\frac{\sum_{i\in I_s} f(i, g)}{\bar{f}(g)}=\frac{f(s, g)}{\bar{f}(g)}N(s, g)$$

이 식에서 만약 $\frac{f(s, g)}{\bar{f}(g)}=k$ 이면 $N(s, g)=k^g N(s,0)$가 되므로, 개체군의 평균 적합도보다 높은 적합도의 값을 가진 스키마의 개수가 지수 함수적으로 증가한다는 것을 알 수 있다.

다음에는 교차와 돌연변이가 스키마의 인스턴스의 개수에 미치는 영향을 생각하기로 한다. 위에서 살펴본 바와 같이 교차 연산자는 스트링의 임의의 한 부분을 분할하여 다른 스트링과 교환한다. 따라서 스키마의 길이 $\delta(s)$가 길면, 교차 위치가 스키마의 고정 위치 사이로 선택되어 인스턴스가 파괴될 가능성이 높아진다. 즉 l이 스트링에 있는 비트의 개수를 나타낸다면, 스키마 s의 인스턴스가 p_c의 확률로 적용되는 교차에 의해 파괴될 확률은

$$p_c\frac{\delta(s)}{l-1}$$

에 비례한다. 비슷하게, 한 스키마의 인스턴스의 개수는 스키마의 오더 $o(s)$가 클수록 돌연변이에 의하여 파괴될 확률이 높다. 즉 스키마의 인스턴스의 개수는 돌연변이 확률 p_m과 스키마의 오더 $o(s)$의 곱

$$p_m o(s)$$

에 비례하여 감소한다.

위의 결과를 종합하여 선택과 교차 및 돌연변이 연산자에

의한 영향을 모두 고려하면, 다음세대 $g+1$에서의 s의 인스턴스의 개수는 다음과 같이 변경된다.

$$N(s, g+1) \geq N(s, g)\frac{f(s, g)}{\overline{f}(s)} \left[1-p_c\frac{\delta(s)}{l-1}-p_m o(s)\right]$$

이 관계식은 GA에 대한 기본 정리를 제공하는데 흔히 스키마 정리라 불리운다. 이 정리가 의미하는 바는 짧고 오더가 낮으며 평균치 이상의 적합도를 가진 스키마의 개수는 지수함수적으로 증가한다는 것이다.

스키마 정리는 몇가지 제한적인 상황하에서 유도된 것임에 주의하여야 한다. 가령 $f(s, g)/\overline{f}(g)$가 $f(s)/\overline{f}$에 대한 정확한 추정치라고 가정하지만 실제 상황에서는 개체군의 크기가 제한되어 있고 샘플링시의 바이어스로 인하여 오차가 발생한다. 또한 교차 연산자가 기존의 스키마를 파괴적인 연산자로 취급되나, 이는 교차 연산자가 탐색하는데 있어서 실제로는 유용한 연산자라는 실험적인 결과와 배치된다. 이와 같은 몇가지 제한점에도 불구하고 스키마 정리는 유전 알고리즘의 동작에 대한 하나의 이론적인 근거를 제공한다는 점에서 큰 의의를 지닌다.

5. 진화 알고리즘의 응용

진화 알고리즘에 의한 탐색은 탐색공간에 대한 제약, 예를 들어 공간의 연속성과 같은 제약조건을 갖지 않기 때문에 여러가지 문제에 적용이 가능하고 또 주어진 상황에 적응적으로 대처해서 탐색을 한다는 특징을 가지고 있다. 특히 multimodal 공간에서의 탐색, 최적화 및 학습 문제에 탁월한 성능을 보일 수 있음이 여러 연구에 의하여 확인되었다. 이러한 특성으로 인하여 진화 알고리즘은 여러 산업 및 경제 분야에 적용되고 있다. 몇 가지 대표적인 응용의 예가 표 3에 열거되어 있다.

5.1 퍼지 로직과 신경망에의 응용

특히 최근 들어서 퍼지시스템이나 신경망과 결합하여 응용되는 예가 늘어나고 있다. 퍼지시스템과의 결합은 세 가지로 대별할 수 있는데

1. 퍼지 멤버쉽 함수의 최적화
2. 퍼지 규칙 집합의 학습
3. 퍼지 멤버쉽 함수와 퍼지 규칙의 동시 학습

등이 있다 [Carse et al. 1995]. 진화 알고리즘을 신경망 연구에 사용하는 방법도 여러가지가 제안되었는데 이는 다시 세부류로 나누어 볼 수 있다 [Schaffer et al. 1992, Zhang 1992].

1. 신경망 학습에 필요한 데이터의 최적화
2. 신경망의 연결강도를 학습하는 알고리즘으로 사용
3. 신경망 구조의 설계에 이용

첫번째의 경우는 신경망 학습의 전처리로서 유전 알고리즘을 사용하는 것이다. 가령 학습 데이터의 입력 변수의 개수를 줄이거나 유효한 트레이닝 집합을 선택하는데 진화 계산 개념이 적용된 바 있다 [Zhang 1994]. 두 번째의 방법은 기존의 신경망 학습 알고리즘 대신 진화 알고리즘을 사용하는 것인데 대개는 기존의 방법과 결합하여 사용된다. 특히 오차에 대한 미분값이 존재하지 않을 경우에 진화알고리즘은 아주 유용한 학습 알고리즘이다. 신경망 구조의 최적화를 위해서는 많은 방법들이 제안되었는데 이들은 대부분 스트링이나 매트릭스와 같은 형태로 망의 구조를 코딩하고 이에 유전 연산자를 적용함으로써 최적화를 시도한다.

표 3. 진화 알고리즘의 응용 분야.

응용 분야	응 용 사 례
최적화	수치적 함수의 최적화, 가스 파이프라인의 최적화, 전력 송전망의 최적화, 컴퓨터 자판의 최적 배정 문제, 항공기 승무원 배정 문제
설계	VLSI 회로 설계, 비행기 날개의 공기역학적 설계, 엔진 노즐의 설계, 컴퓨터 통신망의 최적 설계, 심장박동기의 설계
인공지능	LISP 프로그램의 자동 생성, 문제해결 규칙의 자동 습득, 신경망 합성 및 학습, 패턴 인식, 자연언어 처리, 멀티에이전트 시스템
시스템 분석 및 예측	시스템 동정, 케이오틱 시계열의 예측, 환률 변화 예측, 단백질 구조 분석, 재정 및 경제 분야에의 예측 및 분석 문제
제어 및 로보틱스	broom balancing, truck backer upper problem, 이동 로봇의 경로 계획, 신경망 및 퍼지로직과 유전알고리즘의 결합에 의한 제어

이에 반해 [Zhang & Muehlenbein 1994]에서 제안된 방법에서는 유전 프로그래밍(GP)에서 착안하여 신경 트리라 명명한 트리 구조의 신경망을 사용하는데 그 크기가 동적으로 변하는 점이 특징이다. 또한 신경망에 응용된 기존의 진화 알고리즘은 대부분이 망의 구조나 연결 강도의 최적화 중 하나를 목표로한 것에 반해, 신경 트리에 의한 방법은 망의 구조와 연결 강도 뿐만 아니라 각 유닛의 종류까지 문제에 적합하게 적응시킬 수 있게 확장되었으며, 특히 이들이 상호 유기적으로 그리고 동적으로 유전 연산자에 의해 최적화되도록 설계되었다. 다음절에서는 신경 트리의 진화적 합성과 이의 응용에 대해 좀 더 자세히 살펴보기로 한다.

5.2 신경 트리의 유전 프로그래밍

신경 트리는 인공 신경들이 가변적인 연결선을 통해 트리 형태로 결합된 구조이다. 트리의 잎은 외부 입력 x_i를 나타내며, n개의 변수로 구성된 터미널 집합 X의 원소이다. 트리의 루트 노드는 신경 트리의 출력을 나타낸다. 각 내부 노드는 유닛 타입 u_i, 전이 함수 f_i, 리셉티브 필드 $R(i)$, 그리고 웨이트 벡터 w_i를 가진다. $R(i)$는 i번째 유닛에 입력을 제공하는 유닛들의 인수 집합을 나타낸다. 신경 트리는 하나의 망 구조에 서로 다른 종류의 신경 유닛을 포함할 수 있다. 가능한 신경 유닛은 응용 분야에 따라 정의될 수 있으며 구체적인 실현은 진화적 학습에 의해 결정된다. 예를 들어, 시그마 유닛과 파이 유닛을 함께 사용함으로써 여러 개의 은닉층을 사용하지 않고도 복잡한 함수 구조를 효과적으로 학습할 수 있다.

주어진 문제에 적합한 신경망 모델을 구축하기 위해 M개의 신경 트리로 구성된 개체군 A를 유지한다. 처음에는 트리가 임의의 크기와 모양으로 초기화된다. 각각의 신경 트리는 아래에 기술되는 평가 함수에 의해 적합도가 계산되며 상위의 τ%가 메이팅풀 B에 선택된다. 다음 세대의 M개의 개체는 기존의 신경 트리의 일부를 다른 신경 트리와 치환함으로써(crossover) 신경 트리의 구조와 크기를 변화시킨다. 신경 유닛의 유형과 입력 유닛의 인덱스는 돌연변이(mutation) 연산자에 의해서 변형된다. 선택 때 가장 좋은 성능을 가진 개체는 항상 다음 세대에 살아남도록 보장하는 엘리트 선택법(elitist strategy)을 사용한다. 그 이유는 세대 교체의 궁극적인 목적이 가장 좋은 성능을 가진 개체를 찾아내는데 있는데 반해 각각의 연산자는 국부적인 성능의 저하를 가져올 수 있기 때문에 이로 인해 세대 교체가 일어남에 따라 그 세대까지의 최적의 해가 소실되는 것을 방지하기 위해서이다.

새로운 구조의 신경 트리가 유전 연산자에 의해 생성되면 각각의 트리는 웨이트의 값들이 조정된다. 이 국부적 학습은 BGA 「Muehlenbein 1993」에 의해 다음과 같이 웨이트의 값을 변경한다.

$$\Delta w = R \cdot 2^{-\varepsilon \cdot K}$$

여기서 K는 상수이며 ε은 0과 1사이의 임의의 값이다. 이 방법은 다양한 범위의 최적화 문제에 있어서 효과가 있음이 입증되었다. 각 신경 트리의 적합도는 (최소화 문제의 경우) 다음과 같은 비용 함수로 평가된다.

$$F_i = F(D/A_i) = \frac{E(D/A_i)}{m \cdot N} + \frac{C(A_i)}{N \cdot C_{max}}$$

위 식에서 m은 출력 유닛의 갯수이고 N은 학습에 사용된 예의 갯수이다. 첫번째 항은 학습 집합 D에 대한 오차 E를 반영하고 두번째 항은 트리의 복잡도 C를 반영한다. 이 적합 함수는 단순한 트리를 복잡한 트리에 대해 더 높이 평가함(적은 Fi 값)으로써 같은 오차를 가진 신경망의 경우 단순한 구조를 가진 신경망이 우선권을 가지고 선택될 가능성을 높여준다. 이렇게 함으로써 학습 데이터에 대한 오차를 줄이기 위해 트리의 크기가 점점 커지려는 경향을 줄여주어 학습된 모델의 일반화 능력을 향상시킬 수 있다 [Zhang & Muehlenbein 1995].

신경 트리의 진화에 의한 신경망 구조의 설계 및 학습법은 여러 가지 예측 및 진단 시스템의 자동 설계 및 학습에 사용되었다 [Zhang & Muehlenbein 1996]. 한 가지 응용예는 환경 오염도의 예측 문제인데 학습 데이터는 일리노이의 상가몬강에서 1970년 1월 1일부터 1971년 12월 31일까지의 기간 동안에 강물에 포함된 질산의 함량 변화에 대한 시계열 자료이다. 학습의 목적은 지난 몇 주 동안의 함량으로부터 그 다음 주의 질산의 양을 예측하는 것이다. 예측된 농도는 수실의 오염도를 모니터링하고 제어하는데 사용된다. 트레이닝 데이터는 함량 변화에 대한 시계열 자료로부터 최근 4주간의 변화를 입력해서 다음주의 값을 예상하는 것으로 생각하여 생성되었다. 이 예측 문제를 풀기 위해 초기의 신경 트리들은 각 유닛의 최대 입력의 갯수가 4개, 그리고 트리의 최대 깊이가 4로 무작위로 생성되었다. 신경 유닛의 50%는 시그마뉴런 나머지 50%는 파이뉴런으로 초기화되었다. 각 웨이트의 값은 「−10, +10」의 범위 내에서 조정되었으며, 300번의 세대 교체 후의 가장 좋은 결과는 3개의 은닉층에 10개의 은닉 신경유닛을 가진 신경망에 의해 얻을 수 있었다. 트레이닝 집합에 대한 학습 결과 그 평균자승오차는 0.0104였다. 진화에 의해 학습된 신경 트리모델의 실제적인 예측 능력을 검토하기 위해 학습 때 알려주지 않은 1972년의 자료를 사용하여 그 성능을 평가하였다. 그 결과 전통적인 방법인 GMDH에 의해 얻은 예측치보다 보다 더 정확한 값을 얻을 수 있었다.

6. 결 어

본고에서는 진화 알고리즘의 기본 구성과 동작 원리를 간단한 예제를 통하여 살펴 보았다. 또한 진화 계산의 이론적 기반이 되는 스키마 정리에 대하여 고찰하였다. 응용 사례로서는 진화 알고리즘을 사용하여서 신경망의 구조를 최적화하고 학습시키는 방법에 대하여 기술하였다.

진화 알고리즘은 다양한 분야에 적용되고 있는데 이와 같은 많은 응용의 배경에는 몇 가지 이유가 있다. 그 한 가지는 알고리즘의 단순성과 일반성에서 찾을 수 있다. 아무리 복잡한 문제라도 일단 염색체 형태로 표현이 되면, 염색체의 복제와 재결합 및 적합도의 평가 등과 같은 비교적 단순한 연산 과정의 반복을 통해 계산이 수행된다. 또 하나의 이유는 기존의 문제 해결 방법과 결합하여 사용하기가 쉽다는 것이다. 이것은 진화 알고리즘이 문제 해결에 특수한 정보를 많이 사용하지 않고 또한 문제에 대한 배경지식이 있으면 이를 쉽게 수용할 수 있기 때문이다. 예를 들어 문제 P에 대한 기존의 경험적 알고리즘 A가 존재한다면, 먼저 A를 사용하여 여러 개의 가능한 해들을 생성한 다음 이를 초기의 개체군으로 생각하고 그 위에 진화 알고리즘을 적용함으로써 적어도 기존의 알고리즘 A의 결과 보다는 더 우수한 결과를 얻을 수 있다. 진화 알고리즘이 흥미를 끄는 또 하나의 이유는 계산 모델이 자연 현상에 기반을 두고 있다는 것이다. 인간이 오랜동안 풀려고 했던 많은 실세계의 문제들을 자연은 실제로 유연하게 해결하고 있다.

진화 알고리즘이 기존의 방법들보다 더 효과적으로 적용될 수 있는 응용 영역의 성격을 정확히 규정하는 문제는 아직 더 연구 되어야 할 과제로 남아 있으나, 적어도 다음과 같은 특성을 갖는 문제에 대해서는 진화 계산 방법이 다른 탐색 방법보다 더 효과적으로 적용된다는 사실이 많은 응용 결과에 의하여 확인되었다.

- 여러 개의 국부적 해가 존재하는 문제(multi-modal)
- 어느 정도의 규칙성을 가지고 있는 문제(regularity)
- 문제 영역의 규칙성이 어느 정도 염색체로 표현가능한 문제
- 부분적인 해들 간에 상대적인 우위관계가(epistatis) 존재하는 문제

현재 추세로 간다면 유전 알고리즘의 응용 범위는 점점 더 늘어날 것이며 이론적인 발전도 가속화되리라 예상된다. 특히 진화 계산 방식은 신경망과 퍼지로직과 결합되어 적응 학습 및 최적화 문제의 새로운 해결법으로 많은 새로운 결과들이 나올 것으로 예상된다.

참 고 문 헌

[1] 「Baeck & Schwefel 1993」 T. Baeck and H.-P. Schwefel, An overview of evolutionary algorithms for parameter optimization, *Evolutionary Computation*, 1(1) : 1-23, 1993.

[2] 「Carse et al. 1995」 B. Carse, T. C. Fogarty and A. Munro, Adaptive distributed routing using evolutionary fuzzy control, in *Proc. of 6th Int. Conf. on Genetic Algorithms*, L. J. Eshelman (ed.), Morgan Kaufmann, San Francisco, C.A., 1995, pp. 389-396.

[3] 「Fogel 1966」 D.B. Fogel, *Artificial Intelligence through Simulated Evolution*, John Wiley, N.Y., 1966.

[4] 「Fogel 1995」 D.B. Fogel, *Evolutionary Computation : Toward a New Philosophy of Machine Intelligence*, IEEE Press, 1995.

[5] 「Goldberg 1989」 D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[6] 「Goldberg 1994」 D. E. Goldberg, Genetic and evolutionary algorithms come of age, *Communications of the ACM*, 39(3) : 113-119, 1994.

[7] 「Holland 1975」 J.H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, M.I., 1975.

[8] 「Koza 1992」 J. R. Koza. *Genetic Programming : On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.

[9] 「Muehlenbein 1993」 H. Muehlenbein and D. Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I : Continuous parameter optimization, *Evolutionary Computation*, 1(1) : 25-49, 1993.

[10] 「Rechenberg 1973」 I. Rechenberg, *Evolutionsstrategie : Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*, Frommann-Holzberg Verlag, Stuttgart, 1973.

[11] 「Schaffer et al. 1992」 J. D. Schaffer, D. Whitley and L. J. Eshelman, Combinations of genetic algorithms and neural networks : A survey of the state of the art, in *Proc. Int. Workshop on Combinations of Genetic Algorithms and Neural Networks*, IEEE, 1992, pp. 1-37.

[12] 「Srinivas & Patnaik 1994」 M. Srinivas and L. M. Patnaik, Genetic algorithms : a survey, *IEEE Computer, June* 1994, pp. 17-26.

[13] 「Winston 1992」 P. H. Winston, *Artificial Intelli-*

gence, Addison-Wesley, 1992.

[14] 「Zhang 1992」 B. T. Zhang, *Lernen durch Genetisch -Neuronale Evolution*, DISKI Vol. 16, ISBN 3-929037-16-5, Infix-Verlag, Bonn/St. Augustin, 1992.

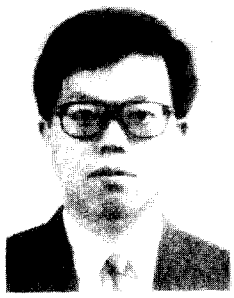
[15] 「Zhang & Muehlenbein 1993」 B. T. Zhang and H. Muehlenbein, Evolving optimal neural networks using genetic algorithms with Occam's razor, *Complex Systems*, 7(3) : 199-220, 1993.

[16] 「Zhang 1994」 B. T. Zhang, Accelerated learning by active example selection, *International Journal of Neural Systems*, 5(1) : 67-75, 1994.

[17] 「Zhang & Muehlenbein 1995」 B. T. Zhang and H. Muehlenbein, Balancing accuracy and parsimony in genetic programming, *Evolutionary Computation*, 3 (1) : 17-38, 1995.

[18] 「Zhang & Muehlenbein 1996」 B. T. Zhang and H. Muehlenbein, Adaptive fitness functions for dynamic growing/pruning of program trees, *Advances in Genetic Programming 2*, MIT Press, Cambridge, M. A., Chapter 13, 1996.

저 자 소 개



장 병 탁

1963년 7월 11일생

1986년 서울 대학교 컴퓨터공학과(학사)

1988년 서울대학교 대학원 컴퓨터공학과(석사)

1992년 독일 Bonn대학교 전산학과(박사)

경력 1988-1992 : Universitaet Bonn, AI Lab. 연구원

1992-1995 : 독일국립전산학 연구소(GMD) 연구원

1995-현재 : 건국 대학교 컴퓨터공학과 조교수

주 관심 분야 : 진화계산, 신경망시스템, 기계학습, 인공지능

서울 관악구 남현동 602-250

TEL) (02)450-3510 / FAX) (02)444-9044