한국퍼지 및 지능 시스템학회 논문지 1998, Vol. 8, No. 3. (1998) 1~8

강화학습에 의한 유전자 프로그래밍의 성능개선

Performance Improvement of Genetic Programming Based on Reinforcement Learning

전효병 · 이동욱 · 심귀보

Hyo-Byung Jun, Dong-Wook Lee and Kwee-Bo Sim

중앙대학교 전자전기공학부

요 약

본 논문에서는 유전자 프로그래밍의 성능을 향상시키기 위하여 강화학습법에 기반한 강화 유전자 프로그래밍을 제안한다. 트리구조의 프로그램을 염색체로 가지는 유전자 프로그래밍(GP)은 다른 진화 알고리즘에 비해 염색체의 크기에 제한이 없기 때문에 표현력에 융통성이 많다는 장점이 있다. 그러나 이러한 특징은 반대로 교차 및 돌연변이 연산에 있어서 수렴성을 떨어뜨리는 단점을 나타낸다. 따라서 유전자 프로그래밍은 다른 진화알고리즘에 비해 개체군의 크기 및 진화 세대수를 크게 잡는 것이 일반적이다. 본 논문에서는 유전자 프로그래밍의 이러한 성질을 개선하기 위해서 프로그램에 강화신호를 주어 이것의 보답/벌칙의 정도에 기반한 교차 및 돌연변이 연산을 실행하는 방법을 제안한다. 제안된 방법은 인공개미(Artificial Ant) 문제에 적용하여 그 유효성을 입증한다.

ABSTRACT

This paper proposes a reinforcement genetic programming based on the reinforcement learning method for the performance improvement of genetic programming. Genetic programming which has tree structure program has much flexibility of problem expression because it has no limitation in the size of chromosome compared to the other evolutionary algorithms. But worse results on the point of convergence associated with mutation and crossover operations are often due to this characteristic. Therefore the sizes of population and maximum generation are typically larger than those of the other evolutionary algorithms. This paper proposes a new method that executes crossover and mutation operations based on reinforcement and inhibition mechanism of reinforcement learning. The validity of the proposed method is evaluated by appling it to the artificial ant problem.

1. 서 론

지식정보처리로 대표되는 인공지능 기술은 공학적 문제영역을 중심으로 정보처리의 효율화 및 지능화가 필요한 많은 분야에 침투해 왔다. 그러나 논리 및 기호의 표현에 의존하는 기존의 인공지능 기술은 이상화된 조건과 정보의 완전성을 전제로 하는 곳에서는 훌륭한 기능을 발휘하지만 불확실성이 존재하는 동적인 환경에서의 처리는 완전하지 못하다. 최근 이에 대한 해결책을 제시하며 인공생명[1,2]이라는 분야가 나타났다. 인공생명은, 컴퓨터 기술의 발달로 복잡계 또는 가상적인 상황을 모의로 실험해볼 수 있게 됨에 따라 이와 같은 기술을 이용하여 복잡한 생명현상의 해명 및 그 원리의 공학적 응용을 목표로 한다. 인공생명은 기존의 인공지능 방법에 비하여 합성적인 접근방식을 취한다. 현재 인공생명의 가장 중심적인 역할을 하고 있는 것 중의 하나가 자연계의 진화이론에 바탕을 둔 진화 알고리즘(EA)[3]이다. 진화 알고리즘에는 대표적으로 유전자 알고리즘(GA : Genetic Algorithms), 유전자 프로그래밍(GP : Genetic Programming), 진화 전략(ES : Evolution Strategies), 진화 프로그래밍(EP : Evolutionary Programming) 등이 있으며, 이들 중 유전자 프로그래밍은 개체의 행동을 트리구조의 프로그램으로 표현할 수 있고 프로그램의 크기에 제한성이 없기 때문에 표현력에 융통성이 많다. 따라서 적용되는 분야도 다양하며, 로봇의 창발적인 행동의

※이 논문은 1997년 학술 진흥재단의 공모과제 연구비에 의하여 연구되었음

발현에도 좋은 결과를 기대할 수 있다[4-6].

유전자 프로그래밍은 1990년대 초반 Koza에 의해 제안되었으며, 현재 인공지능의 여러 가지 문제의 해결에 응용되고 있다. 염색체로서 LISP의 S-식(Symbolic Expression) 표현방법인 트리구조의 프로그램을 사용하고 다른 진화 알고리즘과 마찬가지로 선택, 교차, 돌연변이 등 유전 연산을 거친다. 그러나 염색체의 크기가 가변적이기 때문에 교차에 의한 형질유전성이 떨어진다. 따라서 유전자 프로그래밍에 있어서는 일반적으로 개체군의 크기를 크게 잡거나 세대수를 길게 하는 등의 방법을 사용한다. 이러한 점을 개선하기 위하여 Koza는 함수의 자동정의(ADF)를 이용하여 연산 회수의 감소를 가져왔고[5], Kinnear는 MA(module acquisition)방법과 기존의 ADF방법의 성능을 비교하였으며[6], D'haeseleer는 문맥유지 교차방법을 제안하였고[7], Maxwell은 병렬적으로 실행 가능한 결합루틴 실행 모델을 제안하였다[8]. 이들의 방법은 적용되는 문제에 따라서 기존의 유전자 프로그래밍에 비하여 진화의 성능을 향상 시켰다.

저자들은 다수 로봇의 협조행동의 자율 생성적 실현을 위하여 로봇의 프로그램을 유전자 프로그래밍을 사용하여 진화시키는 연구를 수행하고 있다[9,10]. 본 논문에서는 이 연구의 일환으로 로봇의 프로그램과 같이 하향식으로 수행되는 프로그램의 진화성능을 개선하기 위한 강화학습의 융합방법을 제안한다. 강화학습은 행동의 옳고 그름의 판단만으로 올바른 행동을 획득해 나가는 행동학습법이다. 유전자 프로그래밍의 염색체인 트리구조의 프로그램은 로봇의 행동결과에 대하여 강화신호를 받고 이 값을 기반으로 각 노드의 돌연변이 및 교차 확률을 결정함으로서 성능의 개선을 실현하였다. 본 알고리즘은 Artificial Ant 문제에 적용하여 그 유효성을 입증한다.

2. 유전자 프로그래밍

인간 프로그래머에 의해서만 생성되던 컴퓨터의 프로그램을 컴퓨터에 의해 자동생성 되도록 하기 위한 방법으로 다윈의 진화이론을 도입한 것이 유전자 프로그래밍이다. 유전자 알고리즘, 진화 전략, 진화 프로그래밍 등이 60~70년대에 개발된 것에 비해 이것은 90년대 초 Stanford 대학의 John Koza에 의해 개발, 소개되어 연구가 진행중이며 아직 개발의 여지가 많이 남아있다.

유전자 프로그래밍의 염색체 표현은 LISP언어의 S-식 표현방법인 트리구조로 되어있으며 다른 진화 연산 방법과 마찬가지로 트리구조의 프로그램을 임의로 여러 개를 생성한 후 이것의 수행능력에 따라 경쟁적으로 선택, 도태되어 진화시킴으로서 원하는 프로그램을 얻어낸다. 이는 기존의 유전자 알고리즘과 유사한 점이다. 그런데 프로그램을 진화시킬 때 원하는 해가 계층적이고 해의 크기와 모양을 미리 알 수 없는 어떤 문제들에 있어서는 고정 길이 염색체 문자열로써는 그 해를 구하기 어려운 점이 있다. 따라서 유전자 프로그래밍에서는 기존의 유전자 알고리즘을 확장하여 그 구조를 고정 길이 염색체 문자열이 아닌 트리 구조라는 새로운 형태를 도입하였다. LISP 프로그래밍 언어의 Symbolic Expression은 이러한 트리구조를 갖는 유전자 프로그래밍의 함수와 말단 기호의 구성을 생성하고 조작하는데 특별히 편리한 방법이다. 유전자 프로그래밍의 각 개체는 각 문제에 대하여 그 문제의 영역에 알맞게 정해진 함수와 말단 기호들로 구성된 모든 결합체들의 무제한의 공간이다. 이 점이 고정길이 염색체 문자열을 갖는 유전자 알고리즘과 대조되는 유전자 프로그래밍의 장점이라고 할 수 있다. 유전자 알고리즘에서 스트링으로 표시되던 염색체를 트리구조의 프로그램으로 확장하면서 학습, 추론, 문제해결 등 적용분야도 넓어졌다.

프로그램 진화를 위한 기본적인 조작 방법은 다음의 세 가지가 있다.

(1) 돌연변이(mutation) - 노드의 임의적인 변경
(2) 교차(crossover) - 다른 트리와 부분트리(subtree)의 교환
(3) 역위(inversion, permutation) - 한 트리 내에서 노드 간의 위치 바꿈

이외에도 편집(editing), 캡슐화(encapsulation), 소거(decimation) 등 문제에 따라 여러 가지 연산자를 사용한다. 유전자 프로그래밍으로 문제를 해결하기 위해서는 우선 다음의 5단계가 필요하다.

(1) 종단기호의 집합 결정
(2) 함수의 집합 결정
(3) 적합도 산정방법의 결정
(4) 알고리즘의 파라메타 결정
(5) 종결조건 결정

프로그램을 진화함으로써 얻는 가장 큰 이점은 프로그래머가 고려해 주기 힘든 상황이 존재하는 프로그램을 선택 및 적응이라는 메커니즘으로 작성이 가능하다는데 있다. 유전자 프로그래밍은 로봇의 프로그램 생성, 게임 프로그램, 화상인식, 인공지능의 여러 가지 문제해결이나 학습 등에 응용되고 있으며 알고리즘의 유효성도 이미 증명이 된 바 있다. 특히 본 연구에서는 군의 협조행동을 달성하기 위한 로봇의 프로그램을 만들어내고 이에 따른 로봇들의 행동에

의해 군 목적의 수행정도를 측정하여 선택, 도태시켜 목적에 맞는 로봇의 프로그램을 얻어내기 위한 방법으로 유전자 프로그래밍을 적용한다.

3. 강화 학습

강화학습은 실험 심리학에서 동물의 학습방법의 연구에서 비롯되었으나, 최근에는 공학에서 여러 다른 인공지능 분야들과 합성하여 학습알고리즘을 향상시키는데 많이 이용되고 있다. 가장 간단한 강화학습 방법은 어떤 행동을 했을 때 수행하고 있는 작업의 방향으로 향상된 상태를 가져오면 그 행동을 유발하기 위하여 그 행동에 강화를 준다는 일반적인 상식의 생각에 기초한다.

강화학습은 일반적으로 제어기 또는 에이전트의 행동에 대한 보상을 최대화하는 상태-행동 규칙이나 행동 발생 전략을 찾는 것이다. 그러나 많은 실세계의 경우에 있어서 목표상태에 도달할 때까지는 중간 단계의 행동에 대한 즉각적인 보상이 주어지지 않는다. 이러한 경우 외부로부터의 강화 신호가 없기 때문에 학습이 일어나지 않게 된다. 그러한 경우에도 목표상태에 도달하기 위해서는 계속적인 학습이 이루어져야 하므로 일시적인 신뢰 할당이 이루어져야한다. 이것을 신뢰 할당 문제(credit-assignment problem)라고 하며 강화 학습에 있어서 가장 중요한 문제라고 할 수 있으며, 이 문제에 대한 가장 일반적인 접근 방법은 강화 신호를 생성하는 외부 평가 함수보다 더 자세한 정보를 얻을 수 있는 내부 평가 함수를 구현하는 것이다.

한편, 강화학습법은 현재까지 Sutton의 TD method에 의한 Actor-critic 구조와 Watkins의 Q-learning 등이 개발되어 있다[11-13]. 그러나 이것은 마코비언 환경(markovian environment)의 환경동정형(exploration oriented) 학습방법으로 앞으로는 비마코비언 환경의 경험강화형(exploitation oriented)의 학습방식으로 발전해갈 전망이다. 최근의 연구에서는 진화연산과 학습과의 상호 보완을 시도한 진화적 강화학습(Evolutionary Reinforcement Learning)[14]도 연구되어 있다. 이것은 인공생명 연구의 입장에서 학습능력의 진화나 학습능력이 진화의 과정에 미치는 효과에 대한 연구이다. 또한 강화신호에 대해서도 진화적인 번식에 좋은 평가기준을 획득할 수 있는 것이 시뮬레이션에 의해 확인되어져 있다[15]. 본 논문에서는 진화적 방법에 의한 프로그램 생성의 효율성을 증대시키기 위한 강화학습의 도입방식을 제안한다.

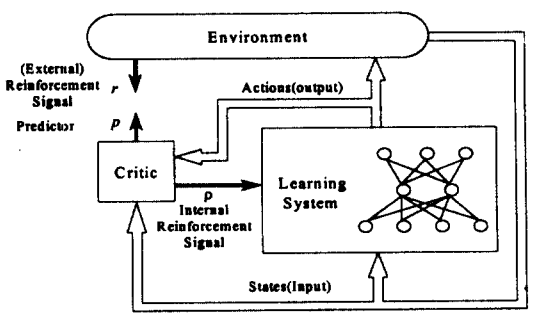
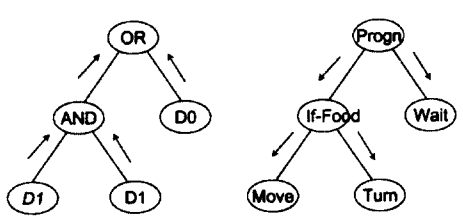


그림 1. 강화학습의 기본 구조.
Fig. 1. Basic structure of reinforcement learning.

4. 강화 유전자 프로그래밍

일반적인 유전자 프로그래밍에서는 각 노드의 교차 및 돌연변이의 확률이 일정하다. 따라서 모든 노드는 랜덤하게 선택되어 교차 및 돌연변이의 과정을 거친다. 그러나 이것은 유용한 스키마(빌딩 블록: building block)에 대하여도 똑같은 비율로 연산이 적용되므로 유용한 스키마가 손실될 가능성도 많으며 이것은 성능의 저하로 이어진다. 본 논문에서는 이와 같은 유용한 스키마가 손실될 가능성을 줄이기 위하여 강화신호를 이용해 돌연변이 지점과 교차 지점을 결정하는 방법을 제안한다. 그러나 이러한 방법은 현재로서는 로봇의 제어와 같은 하향식으로 실행되는 문제에만 적용할 수 있으며 함수근사나 패리티 문제와 같은 상향식으로 실행되는 문제에 대하여는 적용하기 어렵다. 본 논문에서는 로봇의 프로그램을 진화시키는 것을 목적으로 하향식으로 실행 문제에 대하여만 생각한다. 상향식이나 하향식으로 실행되는 프로그램의 예는 그림 2와 같다.

로봇의 제어와 같은 문제의 종단기호집합(terminal set)은 행동(action) 또는 센서 입력값(sensor input)이 된다. 또한 행동은 환경에 대하여 상호작용 하는 출력



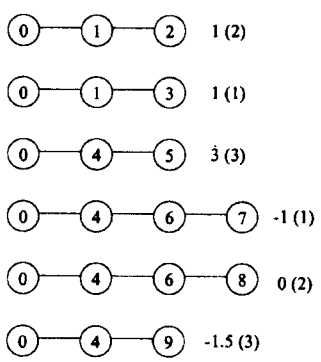
(a) 상향식 실행 방법 (b) 하향식 실행방법
(a) Upward execution method (b) Downward execution method

그림 2. 트리구조 프로그램의 실행방법.
Fig. 2. The execution types of program which is tree structure.

으로서 행동에 대한 보답을 구할 수 있다. 따라서 종단기호집합 중 행동에 대한 강화신호를 계산할 수 있으며 이 값을 근거로 부모노드(함수 집합 : function set)에 대한 강화값을 유추할 수 있다. 각 종단기호가 각각 N_i회씩 실행되고 누적된 강화값이 그림 3-(a)와 같이 주어졌을 때 돌연변이 확률 계산을 위한 각 노드에 대한 강화값(rm)의 계산 식은 (1)식과 같으며, 그 결과는 그림 3-(b)와 같다.

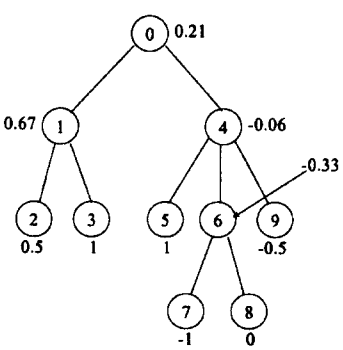
$$rm_i = \frac{\sum_{n \subset T} ram_n}{\sum_{j \subset T} N_j} \tag{1}$$

단, T는 자식트리(subtree)에 속하는 종단기호의 노드 집합, N_j는 노드 j의 실행 회수, ram_n은 노드 n의 누적된 강화값 rm_i는 노드 i의 강화값이다.



(a) 각 종단기호의 누적된 강화값 및 실행된 회수

(a) Accmulated reinforce value and execution number of each terminal node



(b) 각 노드의 강화값

(b) Reinforce value of each node

그림 3. 돌연변이 확률 계산을 위한 노드의 강화값(rm).
Fig. 3. Reinforce value for calculating mutation probability(rm).

예를 들어 4번 노드의 강화값은 (1)식에 의해 다음과 같이 계산된다.

$$rm_4 = \frac{3+(-1)+0+(-0.5)}{3+1+2+3} = -0.06$$

또한 본 논문에서는 강화신호값이 클수록[보상 : reward] 돌연변이 확률을 작게 하고 값이 작을수록[벌칙 : penalty] 돌연변이 확률을 크게 하기 위하여 다음과 같은 식을 사용하였다.

$$pm_i = e^{-2rm_i} pm \tag{2}$$

단, pm_i 는 노드 i의 돌연변이율, pm은 돌연변이율의 기본값이다.

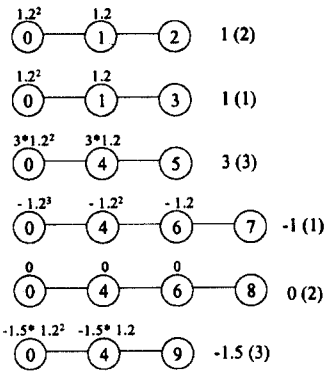
위 (2)식에서 노드 i의 강화값 rm_i가 0일 때 기본적인 pm을 적용받으며 -1일 때 e^2pm, 1일 때 $e^{-2}pm$의 돌연변이율을 적용받는다.

교차확률의 계산은 돌연변이율에 비하여 약간 복잡하다. 그 이유는 교차연산에 의하여 훌륭한 스키마(building block)의 파괴를 최소화하고 나쁜 부트리를 제거하기 위함인데 그 방법은 다음과 같다.

한 노드의 강화값이 크다는 것은 자식트리에 속하는 종단기호가 모두 좋은 강화신호를 받았다는 것이 된다. 또 반대로 강화값이 작을 경우는 자식트리의 종단기호가 모두 안 좋은 강화신호를 받은 것이 된다. 따라서 좋은 부트리의 손실을 최소화하고 안 좋은 트리를 다른 것과 바꾸기 위하여 노드의 깊이(depth)를 고려해 교차를 위한 강화신호(rc)를 구하였다. 그림 4는 각 종단기호가 한 번씩 실행되었을 경우 각 노드의 교차점 계산을 위한 강화값을 구하는 방법을 보인 예제이며 그 식은 (3)식과 같다.

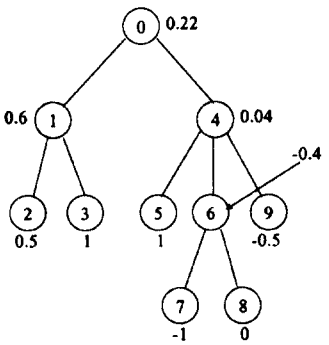
교차를 위한 강화값의 설정에 있어서는 노드의 깊이(depth)에 따른 강화상수 $\alpha(\alpha>1)$를 도입하였다. 그 이유는 일반적으로 부모의 노드는 자식노드의 강화값의 대략적인 평균값을 갖기 때문에 보상과 벌칙을 받은 자식노드가 함께 존재할 경우 부모노드의 강화값이 0에 가깝게 된다. 따라서 교차점은 주로 종단노드의 벌칙을 많이 받은 노드에서 실행될 가능성이 높다. 이것은 교차에 의해 새로 생성되는 프로그램의 길이를 극단적으로 짧게 하거나 길게 하기 때문에 교차의 성능을 떨어뜨린다. 따라서 부모의 노드는 작은 값에서도 교차지점으로 선택될 확률이 높은 것이 바람직하다.

$$rc_i = \frac{\sum_{n \in T} \alpha^{(d_n - d_i)} \times rac_n}{\sum_{j \in T} N_j} \tag{3}$$



(a) 각 종단기호의 누적된 강화값 및 실행된 회수

(a) Accmulated reinforce value and execution number of each terminal node



(b) 각 노드의 강화값

(b) Reinforce value of each node

그림 4. 교차점 계산을 위한 노드의 강화값(rc)(α=1.2).
Fig. 4. Reinforce value for calculating crossover point(rc) (α=1.2).

단, rac_n은 노드 n의 누적된 강화값, rc_n은 노드 n의 강화값, N_j는 노드 j의 실행 회수, α는 강화상수, d_i , d_n 은 각각 노드 i와 n의 깊이(depth)이다.

이때, 교차점의 선택을 위한 비율값은 강화값에 대한 시그모이드 함수의 형태로 (4)식과 같이 정하였다.

$$cv_i = \frac{1}{2}(1-\tanh(2rc_i)) \tag{4}$$

단, cv_i 는 교차점 선택비율값이다. 이때, 임의의 노드 i가 교차점으로 선택될 확률($P(s_i)$)은 cv_i값에 비례하는 (5)식과 같이 구해진다.

$$P(s_i) = \frac{cv_i}{\sum_{n=0}^{N-1} cv_n} \tag{5}$$

단, N은 총 노드의 개수이다. 역위의 경우 교차의 강화값(rc_i)을 그대로 이용해 (6)식과 같이 구할 수 있다.

$$pi_i = (1-\tanh(2rc_i))pi \tag{6}$$

단, pi_i는 노드 i의 역위확률, pi는 역위확률의 기본값이다.

한편, 종단노드의 강화신호 계산은 간단한 문제일 경우 주어진 상황을 고려해 계산해줄 수 있으나, 문제에 따라 보상과 벌칙의 값이 몇 개의 값으로 주어지지 않고 복잡할 경우 퍼지추론에 의해 계산하는 방법도 있다[16].

4. Artificial Ant Problem

4.1 문제의 설정

본 논문에서는 제안한 방법의 평가를 위하여 그림 5와 같은 Artificial Ant 문제를 위한 Santa Fe trail을 이용한다. 이 문제는 32×32의 격자상에서 주어진 먹이를 모두 획득하는 프로그램을 찾는 것을 목표로 한다. 주어진 격자는 Jefferson에 의해 설계된 Trail로서 회전, 1칸의 틈, 1칸의 틈 이후 회전, 2칸의 틈, 2칸의 틈 이후 회전, 3칸의 틈, 3칸의 틈 이후 회전 등 인공개미가 처할 수 있는 다양한 상황에 대하여 고려되어 있다. 참고로 그림 5에서 숫자는 경로상에 존재하는 먹이의 개수를 나타낸 것이다.

4.2 문제 해결을 위한 준비

문제를 유전 프로그래밍에 적용하기 위해 위 2절에

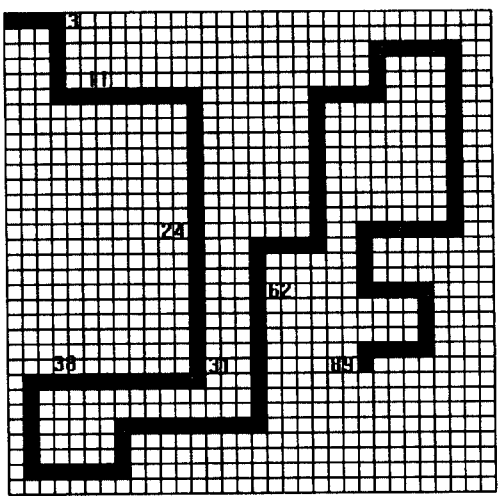


그림 5. 인공개미 문제를 위한 Santa Fe Trail.
Fig. 5. Santa Fe Trail for Artificial Ant Problem.

서 언급한 다섯 단계의 준비가 필요하다. 이 다섯 단계는 Koza가 사용한 방법을 그대로 이용하였다.

(1) 종단기호의 집합(terminal set) 결정

Left, Right, Move

(2) 함수의 집합(function set) 결정

If-Food-Ahead, Progn2, Progn3

(3) 적합도 함수(fitness function)의 결정

적합도 함수는 프로그램의 실행동안 개미가 획득한 먹이의 개수에 비례하며, 이때 최대 프로그램의 실행 시간은 500 스텝으로 하였다. 선택방법은 적합도 비례선택을 사용하였다.

(4) 알고리즘 파라메타의 결정

알고리즘의 파라메타로 개체군(population)의 크기(M), 교차율(pc : probability of crossover), 돌연변이율(pm : probability of mutation), 역위율(pi : probability of inversion) 등이 있으며 다음과 같이 정하였다.

M=400, pc=0.8, pm=0.05, pi=0.02

(5) 종결조건의 결정

주어진 먹이(89개)를 모두 획득한 개체가 발생하면 진화를 멈춘다. 그러나 100세대가 되어도 우수한 프로그램을 찾지 못하면 개체군을 새로 구성하여 진화를 시작한다.

4.3 강화신호의 계산

트리(프로그램)의 각 노드에 강화신호를 할당하기 위하여 환경과 상호작용을 하는 행동(종단기호)에 강화신호를 할당한다. 일반적으로 강화학습을 적용하는 문제에 있어서 즉각적인 보상 보다는 지연된 보상이 주어지는 것이 보통이다. 본 문제에 대하여 다음과 같은 방법으로 보상과 벌칙을 주었다.

- 다음의 경우에 보상(reward)의 강화신호를 받는다.
  - 개미가 먹이를 취한 경우(1)
  - 앞에 먹이가 없을 때 행동 후 앞에 먹이가 생긴 경우(1)
  - 강화신호를 받은 후 이전의 행동(4스텝 이전의 행동 까지)에 대하여 지연된 보상을 준다.(1)
- 다음의 경우에 벌칙(penalty)의 강화신호를 받는다.
  - 앞에 먹이가 있는데도 불구하고 다른 방향으로 움직여 먹이가 사라진 경우(-1)
  - 아무 성과도 없는 경우(-0.5)

4.4 시뮬레이션 결과

다음 그림 6의 (a), (b), (c), 와 (d)는 각각 개체군의 크기가 400, 200, 100, 50일 경우 강화학습을 적용한 경우와 일반적인 경우의 적합도 변화를 비교한 그림이다. 각 그래프는 50회 실행의 평균값으로 하였다.

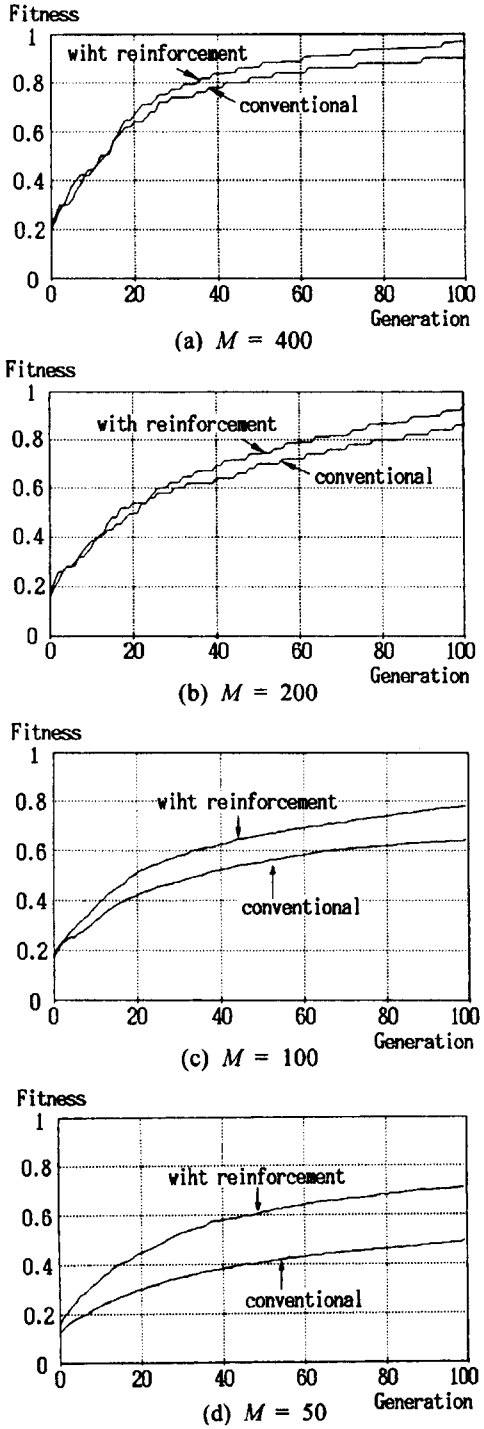


그림 6. 기존의 방법과 강화 유전자 프로그래밍의 성능 비교 (세대수에 따른 적합도의 변화량, 50회 평균).

Fig. 6. Performance comparison beteeen conventional GP and reinforcement GP (Change of fitness vs generation, average of 50 iteration).

```
(IF_FOOD_AHEAD (MOVE)
               (PROGN3(LEFT)
                      (IF_FOOD_AHEAD (MOVE)
                                     (RIGHT))
                      (PROGN3(RIGHT)
                             (IF_FOOD_AHEAD (MOVE)
                                            (LEFT))
                             (MOVE)))).
```

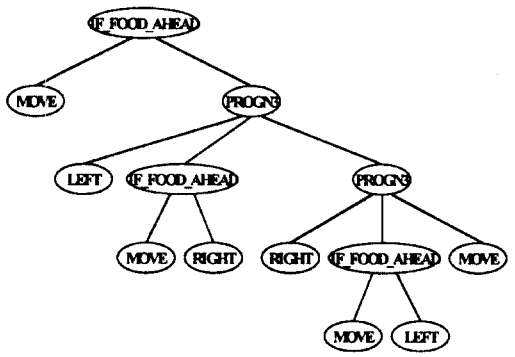


그림 7. Artificial ant 문제의 최적프로그램의 예.
Fig. 7. Example of best individual for artificial ant problem.

개체군의 크기가 400일 경우 100세대 동안 모든 먹이를 취하는 개미를 찾은 경우는 일반적인 경우와 강화학습을 사용한 경우 각각 26회, 34회였다. 즉 강화신호의 사용으로 완벽한 해를 찾을 확률을 높인 것이다. 또한 강화학습은 그림 6에서와 같이 개체군의 수가 적을 경우 일반적인 방법에 비해 더욱 효과적이다. 이 결과는 강화학습을 도입한 경우 계산시간, 수렴시간, 메모리 등의 자원을 절약하고도 같은 성능을 얻어낼 수 있음을 의미한다. 그림 7은 찾아낸 해의 한 예를 나타낸다.

5. 결론 및 추후 과제

본 논문에서는 유전자 프로그래밍의 교차 및 돌연변이의 성능을 개선하기 위하여 강화학습 방법에 기반한 강화 유전자 프로그래밍을 제안하였다. 강화학습을 도입함으로써 기존의 유전자 프로그래밍이 교차 및 돌연변이 연산시 수렴성이 떨어지는 문제점을 개선하였다. 제안한 방법은 Artificial Ant의 문제에 적용하여 그 우수성을 입증하였다. 그러나 제안한 방법은 하향식의 실행을 하는 프로그램에만 적용될 수 있는 것으로, 모든 경우에 대하여 일반적으로 적용가능한 방법의 연구가 더 필요할 것으로 생각된다.

저자들은 인공생명의 합성적인 방법을 이용하여 동적인 환경의 변화가 존재하는 다수의 로봇으로 구성된 시스템을 설계하는 연구를 수행하고 있다. 유전자 프로그래밍은 프로그램의 자동설계의 개념을 사용한 것으로, 이와 같이 복잡한 협조시스템을 구축하는데 훌륭한 도구가 될 수 있다고 생각된다.

참고문헌

[1] C. Langton, C. Taylor, J. Farmer and S. Rasmussen, *Artificial Life II*, Addison-Wesley, 1989.

[2] 上田 完次, 下原 勝憲, 伊庭 齋志, 人工生命の方法, 工業調査會, 1995.

[3] 장병탁, "유전 알고리즘의 이론 및 응용," 대한전자공학회지, **22**(11), pp. 1290-1300, 1995.

[4] John R. Koza, Genetic Programming : *On The Programming of Computers by Means of Natural Selection*, The MIT Press, 1992.

[5] John R. Koza, *Genetic Programming II : Automatic Discovery of Reusable Programs*, The MIT Press, 1994.

[6] Kenneth E. Kinnear, Jr., "Alternatives in Automatic Function Definition : A Comparison of Performance," *in Advances in Genetic Programming*, The MIT Press, pp. 119-141, 1994.

[7] P. D'haeseleer, "Context Preserving Crossover in Genetic Programming," *Proc. of The First IEEE Int. Conf. on Evolutionary Computation*, **1**, pp. 256-261, 1994.

[8] Sidney R. Maxwell III, "Experiments with a Coroutine Execution Model for Genetic Programming," *Proc. of The First IEEE Int. Conf. on Evolutionary Computation*, **1**, pp. 413-417, 1994.

[9] 이동욱, 심귀보, "유전 프로그래밍에 의한 자율이동로봇군의 협조행동 및 제어," *제11회 한국자동제어학술회의 논문집*, **2**, pp. 1177-1180, 1996.

[10] H. B. Jun, D. W. Lee, K. B. Sim, "Cooperative Behavior and Control of Collective Autonomous Mobile Robots using Genetic Programming," *Proc. of 2nd Asian Control Conference*, **2**, pp. 619-622, 1997.

[11] R. S. Sutton, "Learning to Predict by the Methods of Temporal Differences," *Machine Learning*, **8**, pp. 9-44, 1992.

[12] C. J. C. H Watkins, P. Dayan, "Technical Note:Q-Learning," *Machine Learning*, **8**, pp. 279-292, 1992.

[13] J. S. R. Jang, C. T. Sun, E. Mizutani, *Neuro-Fuzzy and Soft Computing*, Prentice Hall, 1997.

[14] H. B. Jun, K. B. Sim, "Behavior Learning and Evolu- tion of Collective Autonomous Mobile Robots based on Reinforcement Learning and Distributed Genetic Algorithm," *Proc. of 6th IEEE International workshop on RO-MAN*, pp. 248-253, 1997.

[15] 이동욱, 전효병, 이준섭, 심귀보, "강화학습에 의한 유전자 프로그래밍의 성능개선," *한국정보과학회*

가을학술발표 논문집(II), pp. 405-408, 1997.

[16] H. B. Jun, D. W. Lee, D. J. Kim and K. B. Sim, "Fuzzy Inference Based Reinforcement Learning of Recurrent Neural Networks," *Proc. of SICE Annual Conf.(Japan, International Session)*, pp. 1083-1088, 1997.

전 효 병(Hyo-Byung Jun) 준회원

제 7 권 제 5 호 참조

현 재 : 중앙대학교 대학원 제어계측학과, 석사과정(로보틱스 및 지능정보 시스템)

이 동 욱(Dong-Wook Lee) 준회원

제 6 권 제 4 호 참조

현 재 : 중앙대학교 대학원 제어계측학과, 박사과정(로보틱스 및 지능정보 시스템)

심 귀 보(Kwee-Bo Sim) 종신회원

제 7 권 제 5 호 참조

현 재 : 중앙대학교 전자전기공학부 교수