유전적 프로그램을 이용한 함수 합성 알고리즘의 개선

An Improved Function Synthesis Algorithm Using Genetic Programming

정 남 채*

Nam-Chae Jung*

요 약

함수합성법은 주어진 입출력 데이터 쌍으로부터 입출력관계를 충족하는 함수를 예측하는 것으로, 특성을 알 수 없는 시스템을 제어할 때에 필수적이다. 일반적으로 시스템은 비선형인 성질을 갖는 경우가 많고, 함수 합성에 취급하는 변수, 정수, 제약 등으로 조합된 문제가 발생하기가 쉽다. 그 함수를 합성하는 방법 중 한 가지로 유전적 프로그래밍이 제안되고 있다. 이것은 함수를 트리구조로 표시한 함수 트리에 유전적 조작을 적용하여, 입출력 관계를 충족하는 함수 트리를 탐색하는 방법이다. 본 논문에서는 기존의 유전적 프로그래밍에 의한 함수 합성법의 문제점을 지적하고, 새로운 4종류의 개선법을 제안한다. 즉, 함수 트리를 탐색할 때에 함수가 복잡하게 되는 것을 방지하기 위하여 함수 트리의 성장 억제, 조기 수렴을 목표로 하는 국소 탐색법의 채택, 함수 트리 내의 필요 없이 길어지는 요소의 효과적인 삭제, 대상으로 하는 문제의 특성을 이용하는 방법이다. 이러한 개선법을 이용할 경우, 기존의 유전적 프로그래밍에 의한 함수 합성법보다도 짧은 시간에 우수한 구조의 함수 트리가 구해지는 것을 2-spirals 문제에 대하여 컴퓨터 시뮬레이션을 통하여 확인하였다.

Abstract

The method of function synthesis is essential when we control the systems not known their characteristic, by predicting the function to satisfy a relation between input and output from the given pairs of input-output data. In general the most systems operate non-linearly, it is easy to come about problem is composed with combinations of parameter, constant, condition, and so on. Genetic programming is proposed by one of function synthesis methods. This is a search method of function tree to satisfy a relation between input and output, with appling genetic operation to function tree to convert function into tree structure.

In this paper, we indicate problems of a function synthesis method by an existing genetic programming, propose four type of new improved method. In other words, there are control of function tree growth, selection of local search method for early convergence, effective elimination of redundancy in function tree, and utilization of problem characteristic of object, for preventing function from complicating when the function tree is searched.

In case of this improved method, we confirmed to obtain superior structure to function synthesis method by an existing genetic programming in a short period of time by means of computer simulation for the two-spirals problem.

Key words : function synthesis, genetic programming, two-spirals problem

Ⅰ.서 론

시스템의 제어와 최적화를 실현하기 위하여 시스템의 동작을 예측하는 것은 대단히 중요하다. 그러기 위해서는 어떤 시스템에 대하여 시행 실험 등에 주어진 입출력 데이터로부터 그 입출력간의 관계를 나타내는 함수를 올바르게 분류하는 것이 필요하다[1],[2]. 일반적으로 시스템은 비선형인 것이 많고, 취급하는 변수, 데이터, 제약 등의 조합이 폭발적으로 증가할 수 있기 때문에, 입출력 함수의 합성 또는 학습은 곤란하게 된다. 비선형 함수 근사에 근거한 대표적인 학습 방법으로는 뉴럴 네트워크, 방사상 기저 함수, 분할 통치법 등을 들 수 있다[3]~[5]. 이러한 방법의 공통점은 미리 적용시킨 기본 함수로 한정되어 있으므로, 국소 해에 빠지기 쉽다. 또한, 그 기본 함수로 복잡한 함수를 합성할 때에는 함수의 규모가 커지는 경향이 있다. 그러므로, 본 논문에서는 기본 함수에 자유도가 있으므로, 대역적으로 탐색하는 성질을 갖는 유전적 프로그래밍에 의한 함수 합성에 주목한다. 유전적 프로그래밍에 의한 함수 합성의 공학적 응용 예에는

* 초당대학교

투고 일자 : 2009. 9. 28 수정완료일자 : 2010. 1. 23
게재확정일자 : 2010. 1. 29

시계열 예측 문제, 패턴인식 문제, 부울 함수 합성 문제 등이 있다[6]. 유전적 프로그래밍은 교차나 돌연변이 등의 유전적 operator를 트리 구조에 적용하고, 목표로 하는 프로그램이나 개념 트리를 탐색하는 방법이다[6],[7]. 본 논문에서 대상으로 하는 함수 합성에 있어서는 트리 구조의 개체가 함수 합성에 대응하고, 각 함수 트리는 함수의 입력이 되는 변수나 정수, 연산자로 구성된다. 이러한 요소는 대상이 되는 시스템에 따라 다르다. 또한, 합성한 함수의 출력은 함수 트리를 잎부터 뿌리로 향하여 계산하고, 뿌리에 해당하는 연산자의 계산결과가 출력값이 된다. 함수를 합성하기 위한 조건은 얻어진 함수의 출력과 실제 출력 간의 오차가 최소이고, 그 함수 크기가 최소가 되는 것이다. 본 논문에서는 Korza가 제안한 가장 기본적인 골격을 갖는 유전적 프로그래밍[7]을 샘플 GP로 부르기로 한다. 샘플 GP는 계산량이 방대하기 때문에, 효율적인 탐색을 위한 몇 가지 방법이 제안되고 있다[8],[9],[11],[12],[15]. 그러나, 이러한 개선법을 적용할 수 있는 문제가 한정되어 있으므로 샘플 GP와 마찬가지로 우연성에 의존하는 경우가 많으므로, 본 논문에서는 2-spirals 문제에 대한 평가실험을 통하여, 샘플 GP의 문제점을 지적한 다음 기존의 개선법과 달리 제약을 받지 않고 여러 가지 문제에 적용할 수 있는 4가지 방법을 제안한다. 첫 번째 방법은 함수 트리의 성장에 관한 제어로 트리의 크기를 억제하면서 유전조작을 하는 것이다. 두 번째 방법은 조기 수렴을 목적으로 함수 트리에 나타난 정수에 대하여 국소 탐색법으로 최적화하는 것이다. 세 번째 방법은 함수 트리의 크기를 축소하기 위하여 함수 트리 내의 필요 없는 요소를 최적화 시간에 삭제하는 것이다. 마지막 네 번째 방법은 취급하는 문제의 특성을 함수 트리에 이용하는 것이다. 위에서 설명한 개선법을 적용한 GP를 평가하기 위하여 2-spirals 문제에 대한 평가 실험을 한다. 즉, 2중 나선이 분포하는 xy 평면 위의 좌표가 어느 쪽의 나선에 속하는지를 분류하는 문제이다. 이 두 가지의 실험결과에 의해서 제안하는 개선법의 유효성을 나타낸다. 다음 Ⅱ장에서는 샘플 GP를 이용한 함수의 합성과 기존의 개선법에 관해서 개략적으로 설명한다. Ⅲ장에서는 2-spirals 문제에 대하여 샘플 GP를 적용하고, 그 문제점을 분석한 다음 기존의 개선법의 한 가지인 MDL에 관해서도 평가한다. Ⅳ장에서는 GP의 개선법을 제안하고, Ⅴ장에서는 2-spirals의 문제에 대하여 개선된 GP를 적용하여 평가하고, 마지막 Ⅵ장에서는 결론을 맺는다.

Ⅱ. 샘플 GP에 의한 함수의 합성과 기존의 개선법

2.1 함수의 합성

함수의 합성은 어느 시스템 혹은 사물에 주어진 입출력 데이터 쌍을 이용하여, 입출력간의 관계를 충족하는 함수를 합성하는 것이다. 함수의 합성을 형식적으로 정의하면 다음과 같다. m개의 입력 변수 x_i와 한 개의 출력변수 y에 규정된 미지의 시스템이 있다고 하면, 미지의 시스템을 함수 f로 표현할 경우 식 (1)과 같이 표시할 수 있다.

$$y = f(x_1, x_2, \cdots, x_m) \tag{1}$$

함수의 합성은 시스템에 주어진 실제의 입출력 데이터 쌍을 이용하여 f의 근사함수 $\hat{f}$를 구하는 것이다. 여기서, 구해진 근사함수는 가장 단순한 구조로 되는 것이 중요하다. 본 논문에서는 이와 같은 근사함수 $\hat{f}$를 GP에 의하여 생성하는 것을 목적으로 한다.

2.2 샘플 GP의 구성

GP는 진화론적 방법으로 데이터를 조작하고, 최적화의 문제나 학습, 추론을 다루는 유전적 알고리즘을, 그 유전자형으로 하여 트리 구조를 취급하도록 확장한 방법이다[6],[7]. Korza가 제안한 가장 기본적인 구조를 갖는 GP를 샘플 GP라 하는데, 샘플 GP의 함수 합성에 관한 구조(유전자형), 적합도, 유전조작 및 알고리즘에 관하여 설명한다.

2.3 데이터의 구조

각 개체는 근사함수 $\hat{f}$에 대응하며 그 데이터 구조는 트리 구조이다. 이 함수 트리는 비종단 노드의 연산자와 종단 노드의 정수 또는 변수로 구성되어 있다. 연산자와 변수의 종류는 대상으로 하는 문제에 대응하여 미리 준비해둔다. $\hat{f}$의 출력치는 함수 트리의 잎부터 뿌리로 향하여 계산하며, 최종적으로 뿌리에 해당하는 연산자의 계산결과가 얻어진다. 함수 트리의 크기를 나타내는 지표로서 깊이와 노드가 이용된다. 깊이는 뿌리부터 가장 먼 잎까지의 거리이고, 노드 수는 트리를 구성하고 있는 연산자, 변수, 정수의 총수이다. 예를 들면, 연산자 '+, -, ×, / ', 변수 'x_1, x_2'로부터 생성된 함수 $\hat{f}(x_1, x_2) = x_1 \times 5.0 + x_2$의 함수 트리를 그림 1에 나타낸다. 이 함수 트리의 깊이, 노드 수는 각각 2, 5가 된다.

샘플 GP에는 함수 트리인 개체를 미리 설정한 개수만큼 준비하여 집단을 형성한다. 초기집단의 생성시, 함수 트리는 초기 깊이의 상한 내에 연산자, 변수, 정수를 랜덤하게 조합하여 생성된다.

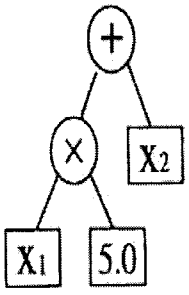

그림 1. 함수 트리의 예
Fig. 1. An example of a function tree.

2.4 적합도

샘플 GP에서 합성된 개체 ($\hat{f}$의 함수 트리)의 평가지표로 일반적으로 각 학습 데이터 $\hat{f}$의 출력치와 실제의 출력치의 2승 오차의 총합을 이용한다. 즉, i번째 데이터에 대하여 $\hat{f}$의 출력치를 $\hat{y_i}$, 실제의 출력치를 y_i로 하면 식 (2)가 되고 그 값이 최소가 되는 함수를 합성한다.

$$(\text{적합도}) = \sum_{i=1}^{data\text{수}} (\hat{y_i} - y_i)^2 \quad (2)$$

2.5 유전 조작

함수 트리의 구성을 진화시킨 유전 조작에는 교차와 돌연변이가 있다.

교차에는 우선 부모가 되는 함수 트리 2개에 대해 각각 랜덤하게 노드를 선택하여 교차점을 결정한다. 그리고, 그 교차점을 뿌리로 하는 부분 트리를 교체하고, 그림 2와 같이 새로운 자식이 되는 함수 트리 2개를 생성한다. 교차에서 생성한 트리의 깊이에는 상한이 설정되어 있으므로, 그 상한을 넘는 트리가 생성되는 경우에는 교차를 무효화한다.

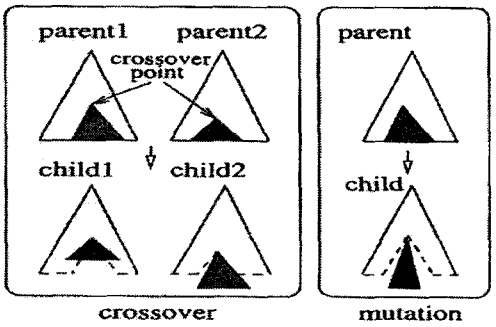

그림 2. 교차와 돌연변이

Fig. 2. Crossover and mutation.

돌연변이에는 한 개의 부모 함수 트리에 대하여 우선 랜덤하게 노드를 선택한다. 그리고, 그 노드를 뿌리로 간주했던 부분 트리를 삭제하여, 그림 2와 같이 새로운 부분 트리를 설정 한 깊이의 상한 내로 랜덤하게 생성한다.

이러한 유전조작을 할 때에는 집단으로부터 부모가 되는 함수 트리를 선택하지만, 그 선택방법은 tournament로 선택한다. Tournament 선택과 집단 내에서 설정한 개수(실험할 때에는 15개)의 개체(함수 트리)를 선택하여, 그 중에서 적합도가 가장 높은 개체를 부모로 하는 선택방법이다. 또한, 본 유전조작에 의해서 집단을 모두 교체하지만, 그 때 교차, 돌연변이 외에 부모의 구조를 그대로 복사한 조작도 실시한다. 실험에서는 이러한 조작 선택 확률을 교차, 돌연변이, 복사 순으로 0.8, 0.1, 0.1로 한다.

2.6 알고리즘 구성

샘플 GP의 알고리즘 구성은 다음과 같다.

(1) 초기집단의 생성

(2) 적합도의 계산

(3) 종료조건의 판정, 설정 세대수 또는 목적 개체를 얻을 수 있으면 종료

(4) 세대교체, 교차, 돌연변이에 의한 유전조작

(5) (2)로 회귀

2.7 샘플 GP의 기존 개선법

위에서 설명한 샘플 GP에 대한 기존의 개선법에 대한 개요와 그 문제점을 다음과 같이 나타낸다.

(1) MDL(Minimum Description Length)

단순한 함수 트리의 방법이 복잡한 구성의 함수 트리보다 우수하다는 것을 기초로 한 개념으로서, 적합도에 함수 트리를 구성하는 노드 수를 추가하여, 노드 수가 적은 함수가 생존하도록 탐색하는 방법이다.

여기서, MDL에는 트리가 커지는 만큼 그 정도가 개선되어 동시에 부분 트리의 정도만으로 정의할 수 있으므로 문제에 적용할 수 있으며 부분 문제로 분할할 수 있다[12].

(2) Adaptive Representation

유익한 부분구조인 빌딩 블록을 발견하여 그것에 대응하는 부분 트리를 파라미터화한 함수로 정의한다. 그리고 그것을 bottom up으로 쌓아 올려가는 것을 목적으로 하는 개체를 탐색하는 방법이다. 그러나, 그 빌딩 블록의 평가기준의 설정은 곤란하여 각 함수 트리 전체에 대하여 평가함수(적합도)와 같은 것을 사용하고 있다[9][10]. 즉, 본 방법은 부분 문제로 분할할 수 없는 문제에는 적용할 수 없다.

(3) ADF(Automatically Defined Functions)

부분 함수와 그 호출한 측의 함수의 집단을 준비하여 각각을 독립적으로 진화시켜 구조적인 규칙성(부분 함수)를 탐색하는 방법이다[11]. 그러나, 규칙성의 탐색도 랜덤한 유전조작을 하였기 때문에, 샘플 GP와 같이 우연성에 의존하는 경향이 있다.

(4) SSAC(Selective Self-Adaptive Crossover)

함수 트리의 각 노드에 선택 확률을 갖게 하여 어떤 노드에서 교차해야할 것인가를 확률적으로 결정하는 방법이다[15]. 그러나, 노드의 선택확률의 초기치는 랜덤하게 설정되어 있어서, 그 노드를 뿌리로 간주한 부분 트리의 평가가 확률적으로 반영되지 않고 샘플 GP와 마찬가지로 우연성에 의존하고 있다.

(5) STROGANOFF(STructured Representation On GA for NOn linear Function Fitting)

통계적 회귀분포 방법과 적응적 탐색 방법(GP)을 통합한 방법이지만, 회귀분석방법을 적용할 수 있는 부분 문제로 분할할 수 없게 된다[8],[13].

위에서 설명한 것처럼, 이러한 기존의 개선법은 적용할 수 있는 문제를 부분 문제로 분할 될 수 있는 문제로 한정되거나 샘플 GP와 같은 우연성에 의존한다는 점이다. 그러므로, 본 논문에서 제안한 개선법의 비교 대상으로는 샘플 GP와 부분 문제로 분할할 수 없어도 적용할 수 있도록 노드 수의 항을 추가하여 적합도를 유지하는 MDL로 한다.

Ⅲ. 샘플 GP의 평가

본 장에서는 2-spirals 문제[15]를 대상으로 하여 샘플 GP, 및 적합도에 함수 트리를 구성하는 노드 수의 항을 갖는 MDL 평가실험을 실시하였다.

3.1 2-spirals 문제

2-spirals 문제는 xy평면 위의 각 점이 그림 3에 표시된 나선 묶음 A, B의 어느 쪽에 속하는가를 분류하는 문제이다.

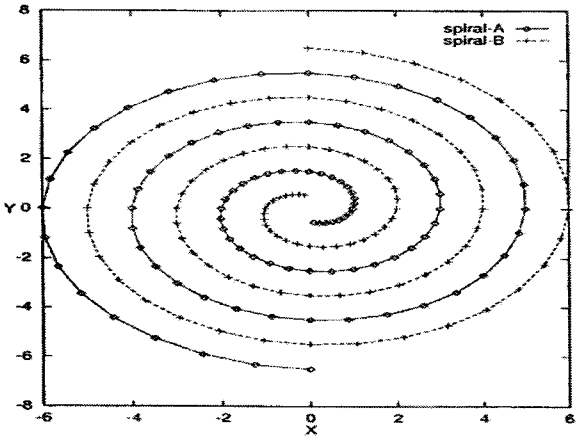

그림 3. 이중 나선의 문제
Fig. 3. Two-spirals problem.

즉, 합성한 함수는 좌표값 (x, y)이 주어졌을 때, "소용돌이 A에 속하거나 소용돌이 B에 속한다."로 분류하는 함수이다. 그러므로, 함수의 출력을 해석하여 출력값이 음수가 아니면 소용돌이 A, 음수이면 소용돌이 B로 정한다. 2-spirals 문제에 관하여 합성하는 함수의 종단, 비종단 노드는 다음과 같다[15].

- 종단 노드 : x 좌표, y좌표, 정수
- 비종단 노드 : '+, -, ×, /, sin, cos, IF'

'+, -, ×, /' 은 인수 2개의 연산자이고, 'sin, cos'는 인수 1개의 연산자이다. 단, '/'에서 제 2 인수가 0.0이 되는 경우는 1.0을 반환한다. 또한, 'IF'는 인수 4개의 연산자로 $IF(m\,n\,M\,N)$의 경우 $m < n$이면 M을 반환하고, 그렇지 않으면 N을 반환하는 연산자이다.

합성하는 함수 트리의 적합도는 올바르게 분류할 수 없는 수, 즉 잘못 분류된 수가 된다. xy 평면 위의 소용돌이 A, B의 합계는 194이기 때문에, 이 194점을 올바르게 분류할 수 있는 함수의 합성을 목적으로 한다.

3.2 샘플 GP의 평가 실험

GP의 실행에 해당하여 초기집단을 생성할 때의 트리 깊이의 상한을 6, 교차 후의 트리 깊이의 상한을 17, 그리고, 돌연변이에서 생성한 부분 트리의 깊이 상한을 4로 하고 있다. 이상의 설정에서 100세대까지 실행한 결과 얻어진 가장 좋은 개체(함수 트리)의 적응도, 즉 잘못 분류한 수는 20개였다. 이 100세대 동안에 생성된 함수 트리의 적합도와 크기의 관계를 나타내기 위하여, 세대마다 함수 트리의 깊이와 노드 수에 관하여 조사한 결과를 각각 그림 4, 5에 나타낸다.

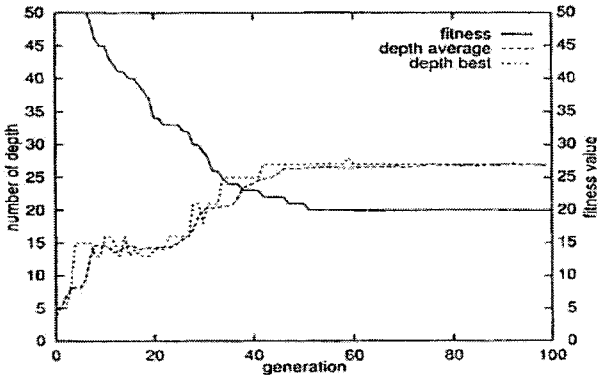

그림 4. 세대마다 트리의 깊이와 적합도
Fig. 4. Change of depth and fitness.

그림 4는 가장 양호한 개체와 총 개체의 평균 트리의 깊이와 가장 양호한 개체의 적합도를 세대마다 나타내고 있으며, 그림 5는 노드 수에 관하여 가장 양호한 개체와 총 개체의 평균을 세대마다 표시하고 있다. 그림 4에 나타낸 것처럼, 가장 양호한 개체는 51세대 이후는 개선되어 있지 않다. 또한, 그림 4, 5에서 세대가 진행함에 따라 노드 수, 트리의 깊이는 단조롭게 증가하고 있는 것을 알 수 있다. 그러나, 적합도가 수렴하는 51세대 이후에 주목하면 트리의 깊이에서 최대 27에 수렴하고 있지만, 노드 수는 계속 증가하는데, 이것은 적응도가 개선되지 않음에도 불구하고, 함수 트리의 크기가 단조롭게 증가함을 나타낸다. 이상으로부터 샘플 GP의 탐색법에는 유전조작을 랜덤하게 계속 실시하여 트리의 구조를 변화시켜 버려, 트리의 성장에 대하여 제어가 되지 않기 때문에 세대가 진행됨에 따라 폭발적으로 트리의 크기가 커진다고 생각할 수 있다.

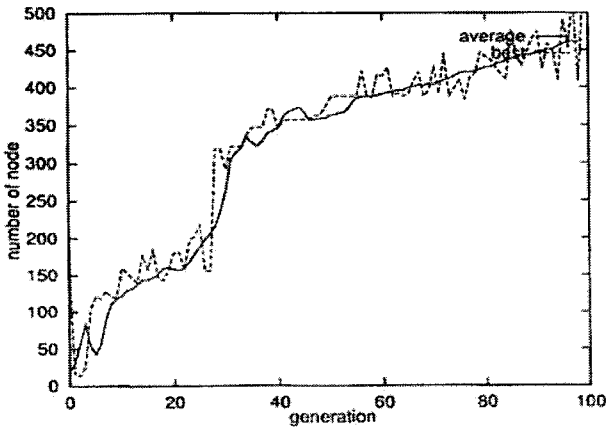

그림 5. 세대마다의 노드 수
Fig. 5. Change of node size

이러한 현상은 목적으로 하는 함수 트리를 탐색하는 과정에서 함수의 출력에 영향을 미치지 않는 부분 트리나 노드(intron)의 생성이 원인이다[14]. 즉, intron에 의해 구조적으로 의미가 있는 빌딩 블록을 교차, 돌연변이 등의 파괴적 작용으로부터 회피한 결과 intron은 GP 실행의 초기부터 중기까지는 유익하게 되지만, 종반에는 brot [bread]라고 불리고, 유전 operator에 의한 작은 intron에서 큰 intron으로 반복적인 치환이 일어나 트리의 노드 수가 지수함수적으로 증대한다.

3.3 MDL의 평가실험

기존의 개선법인 MDL 평가로 함수 트리를 구성하는 노드 수의 항을 더한 적합도를 식 (3)에 의하여 평가 실험한다.

$$(\text{적합도}) = (\text{오차}) + k \cdot (\text{트리의 노드 수}) \quad (3)$$

여기서, k의 값은 1, 0.8, 0.5, 0.1, 0.05로 하여 2-spirals 문제에 대하여 100세대까지 실행한 것 중에서 가장 양호한 함수의 오차와 노드 수를 표 1에 나타낸다. 표 내의 값은 난수를 변경하여 30번 시행한 평균치이다. 표 1에서 k의 값이 증가함에 따라 생성되는 함수 트리에는 오차(잘못 분류된 수)가 증가하고, 노드 수가 억제되어 있는 것을 알 수 있다. 본 결과에서 적합도에 노드 수의 항을 더한 것에 노드 수를 억제할 수 있지만, 그 결과 앞에서 기술한 intron의 작용도 억제되어 정도의 개선을 방해한다고 생각할 수 있다. 여기서 이상의 문제점을 고려한 개선법을 다음 장에서 제안한다.

표 1. 노드 수 부착 적합도의 실험결과

Table 1. Result of MDL

k	0.0	0.05	0.1	0.5	0.8	1.0
error	32.3	36.3	36.4	53.3	55.6	55.5
node 수	133.3	47.5	35.5	11.1	6.6	6.7

Ⅳ GP 개선법의 제안

샘플 GP의 개선법으로 기존의 개선법과 달리 적용문제에 제약을 받지 않는 방법을 4가지 제안한다.

4.1 함수 트리의 성장 억제

샘플 GP는 세대가 진행됨에 따라 함수 트리의 구조가 복잡하게 된다. 이것을 방지하기 위해서 트리의 성장을 억제한다. 구체적으로 우선 유전조작 후의 트리 깊이의 상한을 비교적 작은 값(초기 생성 때의 트리 깊이의 상한과 동일한 값)으로 설정하고, 그 범위 내에서 교차, 돌연변이를 시행하고, 이 상한을 넘는 경우는 그 유전조작을 무효로 한다. 그러나, 설정 상한 내에서 적합도가 일정 세대동안 연속하여 개선되지 않으면, 트리 깊이의 상한을 약간 늘린다. 실험에서는 5세대 동안 적합도가 개선된 개체를 탐색할 수 없는 경우, 유전조작 후의 트리 깊이의 상한을 +2로 하면 작은 크기로 함수 트리의 탐색이 가능하고, 샘플 GP에 관해서 유전조작 후의 깊이의 상한을 미리 적당한 값으로 설정한다고 하는 문제도 해결할 수 있다.

4.2 국소 탐색법에 의한 정수의 최적화

교차나 돌연변이에 의한 랜덤한 구조변경만으로는 우연에 의존하기 때문에 탐색하는 시간이 필요하다. 여기서 이 효율성을 높이기 위하여 국소 탐색법을 도입한다. 즉 함수 트리에 나타난 정수를 국소 탐색법으로 조정함으로써 그 구조에서 얻어진 가장 양호한 함수 트리를 탐색한다. 본 국소 탐색법의 알고리즘을 다음과 같다.

(1) 함수 트리에 나타난 총 정수 노드 중에서 랜덤하게 1개의 정수 노드를 선택한다.

(2) 선택한 정수에 최소 단위를 더한 함수 트리(함수 트리(+))와 유도된 함수 트리(함수 트리(-)) 2개를 생성한다. 여기서, 최소 단위는 다음의 식으로 산출된 값이다.

$$(\text{최소단위}) = (\text{정수값}) \times ([0.01, 0.1]\text{의 난수})$$

(3) 생성한 각 함수 트리의 적합도를 계산한다.

(4) 원래의 함수 트리, 함수 트리(+), 함수 트리(-) 중에서 가장 양호한 함수 트리를 선택하여 그 값으로 치환한다.

(1)로부터 (4)를 함수 트리에 나타난 정수의 개수만큼 반복한다. 국소 탐색법의 적용회수를 고정값이 아니고, 정수의 개수만큼 한 것은 함수 트리에 의하여 포함된 정수의 수가 다양하기 때문이다. 또한, 국소 탐색법은 유전조작 후의 모든 개체에 적용하고 있다.

예를 들면 국소 탐색법의 적용에 관하여 그림 6에 나타낸다. 이 함수 트리에는 정수 노드가 3개 포함되어 있는데, 그 중에서 3.2의 정수 노드가 선택되었다고 가정한다. 이때 최소 단위를 3.2×0.1=0.32로 하면 이것을 더한 함수 트리(+)와 유도된 함수 트리(-)가 생성된다. 이상의 3가지 적합도를 산출하여 그 중에서 가장 양호한 함수 트리로 치환한다.

4.3 필요 없는 요소의 삭제(compact화)의 적용법

compact화는 함수의 계산 과정에서 값의 변화가 없이 필요 없는 함수 트리의 요소를 삭제하는 처리이다. 이러한 개념은 기존에 Koza가 Edit operator에서 제안하고 있다[7]. 그러나 본 논문에서는 그 구체적 실현방법으로 다음의 2가지를 제안하며, 그 적용 시각에 관해서는 5.3에서 고찰한다.

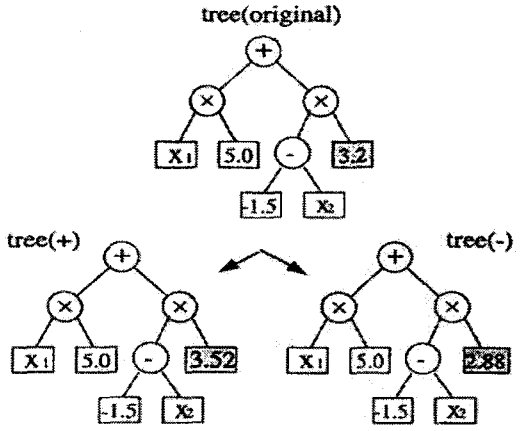

그림 6. 국소 탐색법의 적용 예

Fig. 6. An example of local search method.

(1) 부분 트리의 정수 치환

어느 비종단(연산자) 노드에 관한 계산 결과가 어떤 입력에 대하여도 변하지 않는 경우, 그 비종단 노드를 뿌리로 하는 부분 트리를 정수 노드로 치환한다. 이것에는 다음의 2가지 case가 존재한다.

case 1 : 정수의 계산

case 2 : 동일 변수의 감산, 제산

(2) *IF* 문의 삭제

IF 연산에서 조건 분기하는 부분 트리가 항상 같은 경우, *IF* 문과 기본으로 분기하지 않는 방향의 부분 트리를 삭제한다. 예를 들면, Lisp 형식의 부분 트리 '(*IF* 1.0, 2.0 부분 트리 A 부분 트리 B)'에는 항상 부분 트리 A 에서 분기하기 위하여 부분 B에서는 분기하지 않는다. 여기서 부분 트리 *IF* 를 부분 트리 A 에서만 치환한다.

4.4 대상문제 특성의 이용

대상 문제의 특성으로는 문제의 해(출력)의 분포부터 볼 수 있는 주기성 등의 성질이 있다. 이 성질을 표제로 함수 트리의 부분 함수로 사용함으로써 함수 탐색의 효율화와 구조를 간단하게 표시할 수 있다.

이상 각 방법의 GP에 관한 의의를 정리하면 함수 트리의 성장 억제와 국소 탐색법에 의한 정수 최적화를 병용함으로써 intron의 발생을 억제함과 동시에 building block의 형성을 촉진하고, 또한 compact화로 intron에 대한 억제를 높인다고 생각할 수 있다. 여기서 특성의 이용은 building block으로 주어지고 있다.

V 시뮬레이션 및 검토

제안한 개선법의 평가를 2-spirals 문제에 대하여 시뮬레이션한 결과를 다음과 같이 제시한다. 편의상 1. 함수 트리의 성장 억제, 2. 국소 탐색에 의한 정수 최적화, 3. compact화, 4. 문제의 특성을 이용한 제안법 1, 2, 3, 4로 나타낸다.

5.1 파라미터의 설정

기존법(샘플 GP)의 설정은 3의 실험과 마찬가지로 집단 수 1,500, 초기 생성 때의 트리 깊이의 상한 6, 교차 후의 트리 깊이 상한 17, 돌연변이로의 부분 트리의 깊이 상한 4로 한다. 또한, 제안한 개선법에 관해서는 집단 수를 1,000, 초기 생성 때의 트리 깊이의 상한(유전 조작 후의 트리 깊이의 상한)을 6으로 한다. 종료 조건은 소용돌이 A 와 소용돌이 B 를 형성하는 194점을 올바르게 함수 합성할 수 있는 경우, 또는 100세대 실행한 경우로 한다. 이러한 처리를 1번 시행하여 다른 난수를 이용하여 모두 30번 시행한다.

5.2 제안법의 1+2의 효과

트리의 성장을 억제하면서 국소 탐색법에서 조정하는 개선법(제안법 1+2)과 기존법의 실행 결과를 표 2에 나타낸다. 표 중에 값은 모든 시행의 평균값이다. 본 결과에서 기존법에서는 어떤 시행에 대해서도 100 세대 실행하여도 함수의 합성을 전혀 할 수 없었는데 반하여 제안법 1+2를 적용한 경우, 194점 모두를 올바르게 분류한 함수가 4회 시행한 중에 1회는 합성되어 있는 것을 알 수 있다. 또한, 합성에 실패한 경우에도 제안법 1+2의 방법이 양호한 적합도(잘못 분류한 수)로 되고 있다. 노드 수에 관해서는 기존법에 최악으로 500 정도까지 증대시켜도 함수를 합성할 수 없는 것에 대해서는 제안법 1+2에 의하여 평균 242 노드의 함수를 합성할 수 있다. 즉, 제안법 1+2에서 올바르게 분류하는 함수를 합성할 때 함수의 크기도 기존법에 비하여 작아지고 있다. 또한, 계산 시간에 있어서는 기존법은 몇 시간 실행하여도 목적으로 하는 함수를 합성할 수 없는데 반하여 제안법 1+2에 의해서 평균 7,263초에 합성할 수 있다.

표 2. 제안법 1+2의 효과(2-spirals 문제)

Table 2. Effects of the proposal method 1+2 on the two-spirals problem.

항 목	제안법 1+2	기존법
성공률	23.3%	0%
성공할 때의 합성 세대	69.1	-
실패할 때의 적합도	8.8	32.6

5.3 제안법 3의 효과

위에서 기술한 제안법 1+2에 더하여 compact화 즉 함수 트리 내의 필요 없는 요소를 삭제(제안법 3)하였다.

a) 예비 실험

우선, compact화의 최적인 적용 빈도와 대상 개체를 조사하기 위하여 제안법 1+2에서 함수의 합성이 성공한 5회 시행에 대하여 예비 실험을 하였다.

(1) 적용 빈도

1세대마다, 2세대에 1회, 4세대에 1회, 6세대에 1회, 8세대에 1회, 10세대에 1회의 6가지 방법의 적용 빈도에 대해, 각 세대에 얻어진 가장 양호한 함수 트리에 대하여 compact화를 실행하여 그 결과를 표 3에 나타내었다.

표 3. compact화에 의한 개선률(단위 %)

Table 3. Variation of effects by proposal method 3.

적용 빈도	성공률	노드		합성 세대	
10세대마다	80.0	38.3	(14.9)	16.7	(13.4)
8세대마다	80.0	9.8	(42.5)	21.6	(23.5)
6세대마다	100.0	23.7	(53.7)	2.4	(47.5)
4세대마다	100.0	33.7	(14.6)	35.5	(14.3)
2세대마다	40.0	21.1	(48.2)	43.5	(18.0)
1세대마다	20.0	40.1	(-)	52.1	(-)

표 3의 값은 제안법 1+2의 결과를 기준으로 하여 개선된 비율(%)이다. 또한 ( )내는 표준 편차를, "-"는 계측 불가능

을 나타낸다. 표 3에서 4세대마다, 6세대마다의 경우가 5회 시행 모두 함수를 합성할 수 있었다. 역으로 1세대, 2세대마다의 경우는 함수의 합성을 방해하고 있다고 생각할 수 있다. 여기서 적용 빈도로는 노드의 삭감률과 합성 세대의 개선률 모두에 효과가 있는 4세대마다를 적용하였다.

(2) 대상 개체

다음으로 compact화 적용 대상의 함수 트리를 증가시킨 경우에 대해서 실험하였다. 구체적으로 전집단 10%(실험에서는 100개)를 대상으로 compact화를 시행한 결과 함수의 합성을 완료한 세대가 지연되거나 합성을 할 수 없는 것으로 나타났다.

이상에서 compact화의 적용 빈도나 대상 개체를 많이 하면, 다음 세대에서 진화할 가능성이 있는 부분 트리나 필요한 intron까지 삭제되어, 함수의 탐색을 방해할 수 있다고 판단된다. 즉, 결과에 영향을 주지 않는 트리에도 실행 초기 및 중기의 단계에는 유용한 것과 일치하고 있다[15]. 따라서, 아래에서는 4세대마다 가장 양호한 개체를 대상으로 compact화를 시행하였다.

b) 전시행의 실험

예측 실험으로 설정한 compact화를 제안법 1+2로 조합한 제안법 1+2+3에 의하여 30회 시행한 결과를 표 4에 나타낸다. 표 4에서 제안법 1+2와 비교하여 함수의 합성이 성공할 비율이 개선되므로, 합성 세대의 계산시간도 단축되고 함수 트리의 크기도 작아진다. 따라서 compact화(제안법 3)에 의해서 함수의 탐색이 촉진되어 작은 크기화가 실현된다. 구해진 함수 트리에 대해서는 제안법 1+2+3이 기존법에 비하여 필요 없는 노드가 없고, compact한 함수가 되어 있다. 또한, 함수 트리의 뿌리에 해당하는 연산자가 "sin"인 것으로부터 문제의 특성을 햇빛에 노출시키지 않아도 주기성을 함수 트리에 반영하고 있다.

표 4. Compact화의 효과(2-spirals 문제)
Table 4. Effects of the proposal method 3 on the two-spirals problem.

항목	제안법 1+2+3		제안법 1+2	
성공률	30.3%		23.3%	
성공 때의 합성 세대	59.7	(26.4)	69.1	(19.9)
노드 수	174.2	(87.1)	242.0	(117.8)
깊이	11.1	(3.3)	11.7	(3.7)
계산 시간 (s)	4,189	(2,544.1)	7,263	(5,182.9)

5.4 제안법 4의 효과

제안법 4로 2-spirals 문제의 특성을 이용한다. 2개의 소용돌이는 점대칭에 가까운 분포를 하고 있으므로 극좌표계를 이용하는 것이 유용하며, 함수의 부분 함수로 xy 좌표계를 극좌표계로 변환한 식 (4), (5)를 도입하였다.

$$r = \sqrt{x^2 + y^2} \tag{4}$$

$$\theta = \tan^{-1}\frac{y}{x} \tag{5}$$

이 부분 함수를 종단 노드로 이용하고, 제안법 1+2+3을 적용한 GP에 실행하면, 30회 시행 중 모든 시행에 올바르게 분류한 함수를 합성하고, 평균 6.7세대로 깊이 6.0, 노드 수 12.3의 함수 트리가 구해진다. 구해진 함수의 한 가지 예를 식 (6)으로 표시하면, 제안법 4에 적용한 특성이 대단히 유효하다고 할 수 있다.

$$(\sin(-(+\ -2.88r)(-\ -0.52\theta))) \tag{6}$$

VI 결 론

본 논문에서는 우선 함수 합성법으로 샘플 GP를 이용할 때의 문제점을 지적하였다. 다음에는 그 문제점의 개선과 효율적인 탐색을 목적으로 함수 트리의 성장 억제, 국소 탐색법에 의한 정수의 최적화, compact화의 적용법, 대상 문제 특성의 부분함수로 이용하는 4가지의 개선법을 제안하였다. 그리고 2-spirals 문제에 대한 평가 실험에 의해서 본 개선법이 유효하다는 것을 나타내었다. 본 개선법은 GP를 적용할 수 있는 문제에도 적용할 수 있을 것으로 판단되지만, 실제의 문제에 적용한 실증은 후에 연구해야 할 과제이다.

또한, 태양의 흑점 수가 늘어나거나 줄어들면 지구가 받는 태양의 복사에너지의 세기나 연속성에서 분명한 변화가 있겠지만, 정확하게 물리학적으로 복사에너지의 변화를 시뮬레이션할 수 없는 것이 현실이다. 이러한 점에 착안하여 태양의 흑점 수를 예측하는 실험을 진행하고 있지만 아직 완료되지 않은 관계로 다음 발표로 미루고자 한다.

이번의 연구에서 샘플 GP만의 범위에서 문제를 해석하는 것이 대단히 곤란하다고 추정되었다. 탐색 능력, 풀이 능력을 향상시키기 위해서는 본 논문에서의 국소 탐색법이나 최급강하법 등으로 탐색에 방향성을 갖게 할 필요가 있다. 이와 같이 샘플 GP 외에 $+\alpha$의 탐색방법이 적용되어야만 문제를 해결할 수 있을 것이다.

참 고 문 헌

[1] 片山 徹, システム同定入門, システム制御情報ライブリ— 9, 2008.

[2] K. J. Astrom and P. Eykhoff, "System identification-A survey," Automatica, vol. 7, pp. 123-162, 1971.

[3] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," IEEE Trans. Neural Networks, vol. 1, no. 2, pp. 4-27, 1990.

[4] S. Chen, S. A. Billings, C. F. N. Cowan, and P. M. Grant, "Non-linear systems identification using radial basis function networks," Int. J. Systems Sci., vol. 21, no. 12, pp. 2513-2539, 1990.

[5] 八野知博, 高田 等, "自動抽出關數展開モデルによる非線形システムの同定—遺伝的アルゴリズムによるモデル構造の決定," システム制御情報學會論文誌, vo. 11, no. 3, pp. 127-135, 2008.
[6] 伊庭齊志, 遺伝的プログラミング, 東京電機大學出版局, 2006.
[7] J. Koza, Genetic Programming, On the Programming of Computers by Means of Natural Selection, MIT Press, 1992.
[8] 伊庭齊志, 佐藤泰介, "システム同定アプローチに基づく遺伝的プログラミング," 人工知能誌, vol. 10, no. 4, pp. 590-600, 2004.
[9] J. P. Rosca and D. H. Ballard, "Hierarchical self-organization in genetic programming," Proc. 11th Int. Conf. of Machine Learning, pp. 251-258, 1994.
[10] J. P. Rosca and D. H. Ballard, "Causality in genetic programming," Proc. 6th Int. Conf. of Genetic Algorithms, pp. 256-263, 1994.
[11] K. E. Kinnear, Jr., "Alternatives in automatic function definition: A comparison of performance," Advances in Genetic Programming, pp. 119-141, MIT Press, 1994.
[12] H. Iba, H. de Garis, and T. Sato, "Genetic programming using a minimum description length principle," Advances in Genetic Programming, pp. 265-284, MIT press, 1994.
[13] H. Iba, T. Kurita, H. de Garis, and T. Sato, "System identification using structured genetic algorithm," Proc. 5th. Conf. on Genetic Algorithm, pp. 279-286, 1993.
[14] 伊庭齊志, 進化論的計算の方法, 東京大學出版局, 2008.
[15] P. J. Angeline, "Two self-adaptive crossover operators for genetic programming," Advances in Genetic Programming, pp. 89-109, MIT Press, 1994.

정 남 채 (Nam-chae Jung)

1984년 2월 : 조선대학교 전자공학과 (공학사)
1987년 2월 : 조선대학교 전자공학과 (공학석사)
1992년 8월 : 조선대학교 전기공학과 전자전공(공학박사)
1996년 3월~현재 : 초당대학교 정보통신공학과 교수
관심분야 : Digital 신호처리, Robotics, 의용생체전자공학