문법 코딩에 기반한 유전적 퍼지 시스템의 설계 및 응용

(Design and Application of Genetic-Fuzzy Systems based on Grammatical Encoding)

길 준 민 † 고 명 숙 †† 황 종 선 †††
(Joon-Min Gil)(Myung-Sook Ko)(Chong-Sun Hwang)

요 약 퍼지 시스템의 설계시, 퍼지 시스템의 성능 저하 없이 최적의 퍼지 규칙 선택과 퍼지 소속 함수의 단순한 정의는 매우 중요하다. 이러한 목적을 이루기 위해서, 본 논문에서는 입력 공간에 강한 영향을 보이는 퍼지 규칙만을 퍼지 규칙으로 선택함으로써 입력 공간의 증가에 유연하게 대처할 수 있는 퍼지 규칙 구조를 제안한다. 또한, 유전자 알고리즘의 진화 탐색을 통하여 퍼지 시스템의 최적화된 구조를 얻기 위해서 퍼지 시스템의 구조를 생성시키는 문법 규칙을 해개체로 코딩하는 문법 코딩을 이용한 유전적 퍼지 시스템을 제안한다. 문법 규칙은 퍼지 규칙의 복잡한 구조를 단순한 모듈 구조로 표현하므로 문법 규칙의 코딩은 유전자 알고리즘의 빠른 수렴과 효율적인 탐색을 보장한다. 아울러, 제안하는 방법을 많은 입력 공간을 갖는 아이리스 데이타(Iris data) 문제와 시간열 예측(time series prediction) 문제에 적용함으로써 제안하는 방법의 응용성을 보이고 성능을 분석한다. 실험 결과, 제안하는 방법이 직접 코딩을 사용한 다른 설계 방법보다 더 좋은 성능을 보여 주었다.

Abstract In designing fuzzy systems, it is important to select an optimal subset of fuzzy rules and to define fuzzy membership functions simply without degrading the performance of fuzzy systems. In this paper, in order to achieve such objectives, we propose a structure of fuzzy rules that can flexibly deal with the increase of input space by selecting those fuzzy rules that significantly affect input space among the whole set of fuzzy rules. To find the optimal structure of fuzzy systems through the evolutionary search of genetic algorithms, we also propose genetic-fuzzy systems using a grammatical encoding, in which the grammatical rules generating the structure of fuzzy systems are encoded into an individual. Since grammatical rules represent a complex structure of fuzzy rules as a simple module structure, the encoding of them can guarantee fast convergence and efficient search to genetic algorithms. Moreover, we report on the application and performance analysis of the proposed method by applying to a Iris data problem and a time series prediction problem with much input space. The experimental results show that the proposed method has a better performance than other design methods using the direct encoding.

· 이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의해서 연구되었음.

† 학생회원 : 고려대학교 컴퓨터학과
jmgil@disys.korea.ac.kr
†† 비 회 원 : 고려대학교 정보통신기술연구소 연구조교수
kms@disys.korea.ac.kr
††† 종신회원 : 고려대학교 컴퓨터학과 교수
hwang@disys.korea.ac.kr

논문접수 : 1999년 3월 25일
심사완료 : 2000년 11월 9일

1. 서 론

퍼지 시스템(fuzzy system)은 시스템에 대한 상태를 인간의 사고와 언어의 기술 등을 이용하여 쉽게 표현할 수 있는 장점 때문에 입출력 관계의 정량적인 분석이 어려운 제어 시스템에 성공적으로 적용되어 왔다[1, 2]. 그러나, 퍼지 시스템의 조직적인 설계를 위해서는 다음과 같은 고려 사항이 요구된다[1-6]: ① 퍼지 소속 함수의 구조 결정, ② 퍼지 규칙의 개수 결정, ③ 퍼지 규

칙의 구조 결정.

최근 들어, 퍼지 시스템의 자동적인 설계를 위하여 신경망의 학습 알고리즘(*learning algorithm*) 또는 유전자 알고리즘(genetic algorithm)을 퍼지 시스템의 설계 과정에 도입하여 위의 세 가지 고려 사항을 해결하려는 시도가 활발히 모색되어 왔다[7-15]. 그러나, 이러한 방법론들은 위의 세 가지 고려 사항을 각기 독립적 혹은 순차적으로 고려하였다. 최근의 연구 방향은 퍼지 시스템의 설계 시 매개변수의 조정(parameter tuning: ①)과 구조식별(structure identification: ②, ③)의 과정을 동시에 고려한 연구가 이루어지고 있다. 즉, 퍼지 소속 함수에 대한 매개변수의 최적화 뿐 아니라, 퍼지 규칙의 개수와 구조를 동시에 결정하는데, 이는 퍼지 소속 함수와 퍼지 규칙의 상호 연관성으로 인하여 상호 독립적 혹은 순차적으로 수행되는 퍼지 시스템의 설계 방법은 부분적인 최적화만을 고려한 설계 방법이라는 문제점의 인식에 기초를 두고 있다[5, 6, 16-18].

이러한 문제점의 인식 하에 최근에 제안된 자동적인 퍼지 시스템의 설계 방법론은 전역적 탐색의 장점을 갖는 유전자 알고리즘이 퍼지 소속 함수와 퍼지 규칙의 구조를 동시에 최적화 시키기 위해 융합되고, 유전자 알고리즘의 진화 탐색 과정을 통하여 얻어진 퍼지 규칙과 퍼지 소속 함수의 미세 조정(fine tuning)을 위해서 학습 알고리즘을 적용함으로 최적화된 퍼지 시스템을 구축한다[4-6, 17, 18]. 그러나, 이들 방법론 또한 퍼지 시스템의 설계 과정에서 부딪치게 되는 문제점들을 완전히 해결하지 못하였고, 더욱 심각한 문제점은 퍼지 시스템의 매개변수들이 직접 유전인자(gene)로 코딩(encoding)되어 퍼지 소속 함수와 퍼지 규칙의 개수가 증가됨에 따라서 해개체(individual)의 길이 또한 증가되는 현상을 보이고 있다. 이는 유전자 알고리즘이 최적의 퍼지 시스템을 찾기 위한 탐색 공간의 폭발적인 증가로 귀결된다(유전자 알고리즘이 최적의 퍼지 소속 함수와 퍼지 규칙의 구조를 탐색하기 위해서 이들 매개변수 자체를 직접 해개체로 코딩하는 기법을 직접 코딩(direct encoding)이라고 명명한다). 이를 좀 더 구체적으로 설명하면, 유전자 알고리즘이 찾고자 하는 해는 퍼지 소속 함수와 퍼지 규칙의 구조인데, 일반적으로 모든 입력 변수에 대한 퍼지 소속 함수의 조합으로 표현되는 규칙 테이블을 코딩한다. 입력 공간이 2차원인 경우에는 퍼지 규칙의 구조를 2차원 규칙 테이블 형태로 쉽게 표현할 수 있고, 직접 코딩 기법에 의해서 규칙 테이블을 코딩할 때 유전자 알고리즘은 다루기 쉬운 탐색 공간을 갖는다. 그러나, 입력 공간이 증가함에 따라서 퍼지 규칙의 개수는 폭발적으로 증가하게 된다. 퍼지 규칙의 증가는 유전자 알고리즘의 탐색 공간의 증가로 귀결됨으로써 유전 연산의 수행과 해개체의 평가를 위해서 많은 계산 복잡도와 진화 탐색 시간을 요구한다. 따라서, 입력 공간의 증가에 적절히 대처할 수 있는 퍼지 규칙의 표현 방법이 요구된다.

많은 응용 사례에서 입출력 관계를 표현하는 퍼지 규칙은 모든 입력 공간을 포함하지 않으며 입력 공간에 강한 영향을 갖는 퍼지 규칙만이 필요함을 보여 주었듯이[1, 4-6, 19], 입력 공간을 대표하는 퍼지 규칙의 전제부가 퍼지 규칙의 결론부로 전부 대응되지 않고 입출력 공간에 강하게 의존하는 퍼지 규칙만을 선택적으로 생성하여 퍼지 규칙 베이스를 구성하는 것이 바람직하다.

국내외적으로 이러한 문제점을 해결하기 위해서 [4-6]에서는 퍼지 규칙을 많은 탐색 공간을 야기하는 규칙 테이블의 형태로 표현하기보다는 입출력 관계를 적절히 표현하는 방법들이 제안되어 왔다. [4]에서는 퍼지 소속 함수와 퍼지 규칙의 구조를 입출력 변수의 관계를 갖는 계층적 구조(hierarchical structure)로 표현하였다. [5, 6]에서는 각각 수목 구조(*tree structure*)와 그래프 구조(graph structure)를 이용하여 입력 공간의 특징을 적절히 반영하는 퍼지 규칙의 표현 방법을 제안하였다. 그러나, 이들 방법들은 입력 공간의 효율적인 감소를 위한 기틀을 제공함에도 불구하고, 유전자 알고리즘이 퍼지 시스템을 최적화할 때 직접 코딩 기법을 이용함으로써 탐색 공간의 크기는 여전히 입력 공간의 증가에 좌우되고 이로 인하여 많은 계산 복잡도를 요구한다. 이러한 문제점을 해결하기 위해서 본 논문에서는 퍼지 소속 함수와 퍼지 규칙의 구조 자체를 직접 코딩하여 유전자 알고리즘으로 최적화시키는 직접 코딩 기법에 기반한 퍼지 시스템의 설계 방법 대신에, 퍼지 소속 함수와 퍼지 규칙의 구조를 생성시키는 문법 규칙(grammatical rule)을 해개체로 코딩하는 문법 코딩(grammatical encoding) 기법을 이용하여 퍼지 시스템 내의 매개변수를 유전자 알고리즘으로 최적화하는 방법으로 해결책을 모색하고자 한다. 최근에 임의의 구조를 생성시키는 규칙을 유전자 알고리즘을 이용하여 탐색하는 방법들이 제안되어 왔다[20-23]. 문법(*grammar*)을 최적화하기 위한 방법들이 [20, 21]에서 제안되었으며, 주로 문맥자유 문법(context-free grammar)의 자동 생성에 초점이 맞추어져 있다. 또한, [22, 23]에서는 최적화된 신경망 구조를 찾기 위해서 문법을 진화시키는 유전자 알고리즘이 제안되었다. 이들 방법과 달리, 본 논문에서는

문법 규칙을 위한 유전자 알고리즘을 퍼지 시스템의 설계에 초점을 맞추었다. 본 논문에서의 문법 코딩 기법은 퍼지 소속 함수와 퍼지 규칙의 상호 연관적인 규칙을 유전형(genotype)을 갖는 문법 규칙으로 표현하고 문법 규칙들의 확장에 의해서 표현형(phenotype)을 갖는 퍼지 소속 함수와 퍼지 규칙의 구조를 생성한다.

문법 코딩에 기반한 퍼지 시스템의 설계 방법은 퍼지 규칙의 반복적 패턴(repeated pattern) 혹은 재귀적(recursive) 특성을 표현할 수 있는 문법 규칙을 해개체로 코딩하므로, 불필요한 탐색 공간에서의 비효율적인 탐색이 직접 코딩보다 감소된다. 아울러, 직접 코딩 기법에 기반한 퍼지 시스템의 설계 방법보다 생성되는 퍼지 소속 함수와 퍼지 규칙의 구조를 이해하고 분석하기 쉬운 형태로 표현되고 입력 변수의 증가에 따라서 야기되는 탐색 공간의 증가에 적절히 대처함으로써 유전자 알고리즘의 빠른 수렴과 많은 입력 공간을 갖는 문제에 보다 강건함을 보여 줄 것이다.

본 논문의 구성은 다음과 같다: 2장에서는 자동적인 퍼지시스템의 설계 방법에서 사용하는 퍼지 시스템과 유전자 알고리즘에 대해서 소개한다. 3장에서는 입출력 관계를 적절히 표현할 수 있는 퍼지 규칙 베이스를 구성하기 위하여 퍼지 소속 함수의 개수와 모양 그리고 퍼지 규칙의 구조를 그래프 구조로 표현하는 방법에 대해서 설명한다. 4장에서는 최적화될 퍼지 시스템 내의 매개변수들의 문법 코딩 방법을 기술한다. 5장에서는 유전자 알고리즘에 의한 퍼지 시스템의 최적화 방법이 제시된다. 6장에서는 학습 알고리즘에 대해서 기술한다. 7장에서는 본 논문에서 제안하는 방법을 아이리스 데이타 문제와 시간열 예측 문제에 적용한 실험 및 실험 결과에 대해서 기술한다. 마지막으로 8장에서는 본 논문의 결론 및 향후 연구과제를 제시한다.

2. 퍼지 시스템과 유전자 알고리즘

본 논문의 자동적인 퍼지 시스템의 설계 방법에서 사용할 퍼지 시스템과 유전자 알고리즘에 대해서 기술한다.

2.1 퍼지 시스템

본 논문에서 사용할 퍼지 추론 모델(fuzzy inference model)은 현재 널리 사용되는 단순 퍼지 추론 모델(simplified fuzzy inference model)이다[1, 4-6, 10, 11]. 단순 퍼지 추론 모델은 다음과 같은 퍼지 규칙의 형태를 갖는다:

Rule i: If x_1 is A_1^i and, ..., and x_d is A_d^i then y is w_i (1)

여기서, $A_1^i, A_2^i, \ldots, A_d^i$는 i번째 퍼지 규칙에서 전제부의 퍼지 소속 함수들을 나타내며, w_i는 i번째 퍼지 규칙에서 결론부의 실수값을 나타낸다($i = 1, 2, \ldots, n$). 단순 퍼지 추론 모델에 기반한 퍼지 시스템을 실수 벡터 입력 $\vec{a} = (x_1, x_2, \ldots, x_d)$에서 실수 출력 b로 사상하는 함수 $f: \vec{a} \mapsto b$로 표현하면 다음과 같다:

$$f(\vec{a}) = \frac{\sum_{i=1}^{n} \prod_{j=1}^{d} A_j^i(x_j) \cdot w_i}{\sum_{i=1}^{n} \prod_{j=1}^{d} A_j^i(x_j)} \tag{2}$$

(2)에서 가변인 매개변수는 각 입력 변수 당 할당되는 퍼지 소속 함수의 개수, 퍼지 소속 함수를 위한 매개변수들, 퍼지 규칙의 개수, 퍼지 규칙의 구조, 그리고 퍼지 규칙의 결론부의 실수값이다. 이러한 매개변수들 중 퍼지 규칙의 결론부의 실수값인 w_i는 다음의 LMS(Least Mean Square) 규칙[1]을 이용하여 학습 오차(learning error), $e = \frac{1}{2}\sum_p (y^p - y^{rp})^2$가 최소가 되도록 조정한다:

$$\begin{aligned} w_i(t+1) &= w_i(t) - \eta \cdot \frac{\partial e}{\partial w_i} \\ &= w_i(t) - \eta \cdot \frac{\mu_i^p}{\sum_{i=1}^{n} \mu_i^p}(y^p - y^{rp}) \end{aligned} \tag{3}$$

여기서, y^{rp}는 p번째 입출력 패턴에 대한 실제 출력값이고, y^p는 p번째 입출력 패턴에 대한 퍼지 시스템의 출력값이다. μ_i^p는 p번째 입출력 패턴에 대한 i번째 퍼지 규칙의 신뢰도(support value)이며, η는 학습률(learning rate)이다. (3)의 학습 규칙은 최대 학습 반복 횟수만큼 또는 학습 오차의 변화량에 대한 임계값(threshold) δ에 대해서 $|\Delta e(t)| < \delta$ 를 만족할 때까지 반복 수행된다.

2.2 유전자 알고리즘

유전자 알고리즘(genetic algorithm)은 자연 생태계의 생물학적 자연 진화와 적자 생존의 원리를 모방한 알고리즘으로서 원하는 해를 찾는 탐색 메커니즘이다[24]. 유전자 알고리즘은 세대 t 마다 임의의 문제에 대한 후보 해를 표현하는 해개체(individual)들로 구성된 해집단(population), $P(t) = \{x_1^t, x_2^t, \ldots, x_p^t\}$을 유지한다. 각 해개체들은 미리 정의된 평가함수(evaluation function)에 의해서 해개체의 적합도(fitness)가 설정된다. 해개체의 평가 후 생성되는 적합도 값에 비례하여 현재 세대에서 우수한 해개체는 선택 연산(selection operation)에 의해서 선택되고 다음 세대의 해집단을 구성한다. 새로 생성된 해집단에서 몇몇 해개체들은 좀더 우수한 해개

체를 생성하기 위해서 유전 연산(genetic operation)이 수행된다. 일정 세대 반복 후, 유전자 알고리즘은 점진적으로 해집합을 개선시키면서 원하는 해로 수렴한다.

3. 선택적 퍼지 규칙에 기반한 그래프 구조 퍼지 시스템

퍼지 소속 함수의 구조는 퍼지 규칙의 형태에 많은 영향을 끼칠 뿐 아니라, 퍼지 소속 함수의 모양과 형태에 따라서 퍼지 시스템의 성능을 크게 좌우한다. 입력 공간의 특징을 적절하게 반영하도록 하기 위해서 본 논문에서 사용하는 퍼지 소속 함수는 다음과 같이 정의한다:

$$A(x, c_l, \sigma_l, c_r, \sigma_r) = \begin{cases} \exp\left(-\frac{1}{2}\left(\frac{x-c_l}{\sigma_l}\right)^2\right) & \text{if } x < c_l \\ 1.0 & \text{if } c_l \le x \le c_r \\ \exp\left(-\frac{1}{2}\left(\frac{x-c_r}{\sigma_r}\right)^2\right) & \text{if } x > c_r \end{cases} \quad (4)$$

여기서, c_l과 c_r은 퍼지 소속 함수 $A(x, c_l, \sigma_l, c_r, \sigma_r)$에서 왼쪽 중점과 오른쪽 중점을 나타내고, σ_l과 σ_r은 왼쪽 중점과 오른쪽 중점에서 퍼진 정도를 결정한다.

(4)의 퍼지 소속 함수는 매개변수(c_l, σ_l, c_r, σ_r)를 적절히 조정함으로써 지역적 공간에 반응하는 가우시안 퍼지 소속 함수(gaussian fuzzy membership function)와 전역적 공간에 반응하는 사다리꼴 퍼지 소속 함수(trapezoidal fuzzy membership function)를 동시에 표현할 수 있다.

퍼지 규칙은 시스템의 상태를 기술하는 역할을 수행한다. 현재 주로 사용되는 퍼지 규칙 베이스의 표현 형태는 퍼지 규칙을 모든 입력 변수에 대한 퍼지 소속 함수의 조합으로 만들어지는 규칙 테이블이다.

실제 퍼지 시스템을 설계할 때 적용되는 문제에 따라서 모든 입력 공간에 대응되는 퍼지 규칙 대신에 입력 공간에 강한 반응을 보이는 퍼지 규칙만을 선택하는 경우가 있다. 퍼지 규칙을 규칙 테이블의 형태로 표현하는 것이 입출력 관계를 쉽게 추출하여 퍼지 규칙의 구성을 용이하게 할지라도 규칙 테이블로 표현된 퍼지 규칙은 모든 입력 공간을 완전히 포함하므로 불필요한 입력 변수와 퍼지 소속 함수도 모두 포함됨으로써 입출력 관계를 최적으로 표현한 퍼지 규칙의 구조라고 볼 수 없다. 또한, 규칙 테이블 표현 방법은 입력 공간이 증가할수록 입출력 관계의 복잡도가 증가되므로 입력 공간이 고차원인 경우 최적화된 퍼지 규칙의 추출이 쉽지 않다.

이러한 인식 하에 본 논문에서는 퍼지 규칙의 구조를 다음과 같이 정의하는 그래프(graph)[1]로서 설명한다[6]. 본 논문의 그래프 구조는 퍼지 소속 함수의 개수에 따라서 정의되는 노드(node)와 퍼지 규칙의 구조를 결정하는 에지(edge)들로 구성된다. 노드는 하나의 루트 노드(root node)와 각 입력 변수에 대한 퍼지 소속 함수를 나타내는 언어항 노드(verbal node)들로 구성된다. 루트 노드와 각 언어항 노드 사이의 관계는 방향성을 갖는 에지로 표현되며, 각 변수에 대한 언어항 노드들 사이의 에지는 상위 변수에 대한 언어항 노드로부터 하위 변수에 대한 언어항 노드로 방향성을 갖는다.

설명의 편의를 위해 입력 변수가 5개이고, 첫 번째 입력 변수에서 다섯 번째 입력 변수에 대한 퍼지 소속 함수의 개수가 각각 3개, 2개, 3개, 3개, 4개로 구성되는 경우를 생각해 본다. 그림 1은 이러한 경우에 대응하는 그래프 구조를 나타낸다.

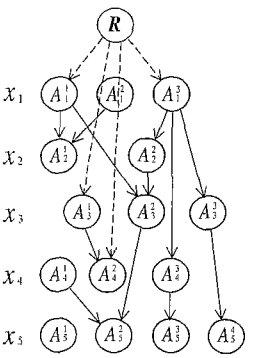


그림 1 그래프 구조의 표현

그림 1에서 R은 루트 노드를 나타내며, A_i^j는 i번째 입력 변수에 대한 j번째 퍼지 소속 함수를 의미하는 언어항 노드를 나타낸다. 루트 노드에서 언어항 노드로의 에지는 점선으로 표현되며, 실선은 언어항 노드들 사이의 에지를 나타낸다.

그래프 구조에 대응되는 퍼지 규칙 구조로의 해석 방법은 루트 노드에서 최하위 입력 변수에 대한 언어항 노드들까지의 경로(path)를 찾는 것이다. 이와 같은 방식으로 그림 1의 그래프 구조에 대응하는 퍼지 규칙은 그림 2와 같이 7개가 생성된다(자세한 설명은 [6]을 참조). 그림 2와 같은 형태를 갖는 퍼지 규칙은 입력 공간 위에서 불필요한 입력 변수와 퍼지 소속 함수의 제거가 가능하다. 즉, 퍼지 규칙을 각 입력 변수에 대한 모든 퍼지 소속 함수들의 조합으로 구성하는 형태가 아닌 선택적 조합의 형태로 나타냄으로써 주어진 문제에 최적의 퍼지 규칙을 생성할 수 있는 기틀을 제공한다.

1) 본 논문에서의 그래프는 DAG(Directed Acyclic Graph)이다.

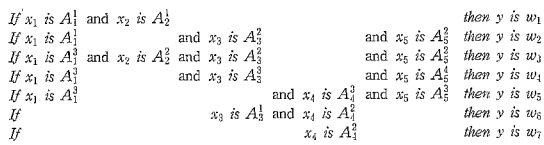


그림 2 그림 1의 그래프 구조에 대응하는 퍼지 규칙

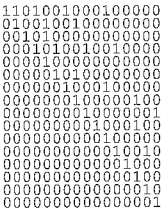
4. 그래프 구조 퍼지 시스템을 위한 코딩 기법

앞서 설명한 그래프 구조로 표현된 퍼지 시스템은 입력 공간에 적절히 반응하도록 퍼지 규칙의 구조를 모든 입력 변수에 대한 퍼지 소속 함수의 조합 형태가 아닌 불필요한 입력 변수와 퍼지 소속 함수가 제거된 형태로 해석하여 퍼지 규칙 베이스를 그래프 구조로서 표현하였다. 그래프 구조하에서, 입력 공간에 대한 퍼지 규칙으로의 적절한 분할을 생성하기 위해서 퍼지 소속 함수를 나타내는 노드와 입력 변수간의 관계를 나타내는 에지는 입력 변수가 두 개 이상이거나 시스템의 입출력 관계가 복잡한 경우 전문가로부터 최적화된 노드와 에지를 추출하기 어렵다. 따라서, 자동적인 최적화 과정이 필요하며, 본 논문에서는 이를 유전자 알고리즘을 이용하여 최적화하고자 한다.

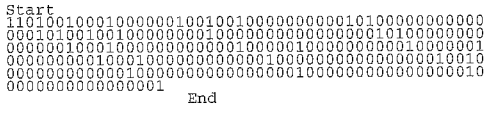
4.1 그래프 구조의 코딩

유전자 알고리즘을 이용하여 최적의 노드와 에지를 탐색하고자 할 때, 가장 먼저 고려해야 할 사항은 노드와 에지의 코딩(encoding) 방법이다. 원하는 해를 어떻게 코딩하는지에 따라서 유전자 알고리즘의 탐색 시간과 탐색 공간의 크기가 좌우된다. 주어진 문제에 대한 효율적이지 못한 코딩은 주어진 문제의 특성을 해개체로 올바르게 반영하지 못하기 때문에 유전자 알고리즘이 수렴할지라도 얻어진 해의 성능은 현저히 저하된다.

[6]에서 제안된 그래프 구조의 코딩 방법은 그래프 구조의 노드와 에지를 연결성 행렬(connectivity matrix)로 표현하고 이를 해개체로 직접 코딩하는 직접 코딩(direct encoding) 방법을 사용하였다. 그림 3은 연결성 행렬로 표현된 그래프 구조를 직접 코딩 기법으로 표현한 예이다. 그림 3(a)의 연결성 행렬은 그림 1의 그래프 구조에 대한 노드와 에지가 행렬로 표현된 형태를 보여준다. 연결성 행렬로부터 그래프 구조의 변환은 다음과 같다: 연결성 행렬의 인덱스가 $i=j$인 경우에는 노드의 존재를 나타낸다. 이 경우 연결성 행렬의 값이 '1'이면 노드가 존재하는 것이고 '0'이면 노드가 존재하지 않는다. 연결성 행렬의 인덱스가 $i\neq j$인 경우에는 연결성 행렬에서 i번째 행과 j번째 열의 값의 해석에 의해서 그래프 구조에서 에지의 유무를 결정한다. 이 경우 연결성 행렬의 값이 '1'이면 i번째 노드에서 j번째 노드로 에지가 존재함을 나타내고 '0'이면 에지가 존재하지 않음을 의미한다. 그림 3(b)의 해개체는 그림 3(a)의 연결성 행렬을 직접 코딩 기법에 의해서 표현한 형태이다. 직접 코딩 기법에 의해서 2차원 행렬로 표현되는 연결성 행렬의 해개체로의 코딩은 노드 수의 증가에 따라서 탐색 공간의 빠른 증가를 야기하고, 유전자 알고리즘이 각 해개체를 평가하고 유전 연산자를 수행하는데 있어서 많은 계산 복잡도를 요구한다. 이러한 인식 하에 본 논문에서는 그래프 구조 자체를 해개체로 직접 코딩하는 직접 코딩 방법 대신 그래프 구조를 생성시키는 문법 규칙을 코딩하는 문법 코딩 방법으로의 접근을 시도한다.



(a) 연결성 행렬



(b) 직접 코딩

그림 3 직접 코딩 기법에 의한 그림 1의 그래프 구조의 코딩

4.2 그래프 생성 시스템

문법 코딩 기법을 설명하기 앞서 그래프 구조를 생성시키는 그래프 생성 시스템(graph generation system)에 대해서 기술한다. 그래프 생성 시스템은 미리 정의된 심볼과 문법 규칙을 이용하여 초기 상태에서 최종 상태로 문법 규칙의 연속적인 확장을 수행함으로써 주어진 문제에 적합한 그래프를 생성한다[25]. 본 논문에서는 스트링-재작성(string-rewriting) 메커니즘의 형태를 도입한 L-system[25]을 사용하여 앞서 설명한 그래프 구조를 다음과 같이 정의하는 그래프 생성 시스템에 의해서 생성한다:

$$G = (NRL, NRR, RL, RR, PL, PR, P, S, w) \quad (5)$$

여기서, RL(recursive left)은 RR에 대응되는 비종결 심볼(nonterminal symbol)의 집합, RR(recursive right)은 비종결 심볼로 구성되는 2×2 행렬의 집합,

NRL(nonrecursive left)은 *NRR*에 대응되는 비종결 심볼의 집합, *NRR*(nonrecursive right)는 *PL*의 원소이면서 종결 심볼(terminal symbol)로 구성되는 2×2 행렬의 집합, *PL*(primitive left)은 *PR*에 대응되는 종결 심볼의 집합, *PR*(primitive right)는 '0'과 '1'로 구성되는 2×2 행렬의 집합, *S*는 *RL*의 한 원소이면서 시작 심볼(start symbol), $(\alpha, \chi) \in P$는 문법 규칙이며 $\alpha \rightarrow \chi$로 표기된다(여기서, α는 *NRL*, *RL*, *PL*의 한 원소이며, χ는 *NRR*, *RR*, *PR*의 한 원소이다). 그리고, w는 확장의 개수이다. 그림 4는 그림 1의 그래프 구조를 위한 그래프 생성 시스템을 보여주며, 그림 5는 그림 1의 그래프 구조를 생성시킬 수 있는 문법 규칙들을 보여준다.

$$G = (NRL, NRR, RL, RR, PL, PR, P, S, w)$$

$$RL = \{S, A, B, C, D\}$$

$$RR = \left\{ \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \begin{bmatrix} E & F \\ H & I \end{bmatrix}, \begin{bmatrix} G & H \\ J & K \end{bmatrix}, \begin{bmatrix} H & H \\ H & H \end{bmatrix}, \begin{bmatrix} L & M \\ H & L \end{bmatrix} \right\}$$

$$NRL = \{E, F, G, H, I, J, K, L, M\}$$

$$NRR = \left\{ \begin{bmatrix} n & c \\ a & f \end{bmatrix}, \begin{bmatrix} d & f \\ f & a \end{bmatrix}, \begin{bmatrix} a & b \\ d & e \end{bmatrix}, \begin{bmatrix} a & a \\ a & a \end{bmatrix}, \begin{bmatrix} f & e \\ a & f \end{bmatrix}, \begin{bmatrix} a & a \\ a & b \end{bmatrix}, \begin{bmatrix} a & a \\ e & a \end{bmatrix}, \begin{bmatrix} f & a \\ a & f \end{bmatrix}, \begin{bmatrix} e & c \\ a & d \end{bmatrix} \right\}$$

$$PL = \{a, b, c, d, e, f, \cdots, n, o, p\}$$

$$PR = \left\{ \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \cdots \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \right\}$$

$$w = 4$$

그림 4 그림 1의 그래프 구조를 위한 그래프 생성 시스템

$$S \rightarrow \begin{bmatrix} A & B \\ C & D \end{bmatrix} \quad A \rightarrow \begin{bmatrix} E & F \\ H & I \end{bmatrix} \quad B \rightarrow \begin{bmatrix} G & H \\ J & K \end{bmatrix} \quad C \rightarrow \begin{bmatrix} H & H \\ H & H \end{bmatrix} \quad D \rightarrow \begin{bmatrix} H & M \\ H & L \end{bmatrix}$$

(a) 재귀적 문법 규칙

$$E \rightarrow \begin{bmatrix} n & c \\ a & f \end{bmatrix} \quad F \rightarrow \begin{bmatrix} d & f \\ f & a \end{bmatrix} \quad G \rightarrow \begin{bmatrix} a & b \\ d & e \end{bmatrix} \quad H \rightarrow \begin{bmatrix} a & a \\ a & a \end{bmatrix} \quad I \rightarrow \begin{bmatrix} f & e \\ a & f \end{bmatrix}$$

$$J \rightarrow \begin{bmatrix} a & a \\ a & b \end{bmatrix} \quad K \rightarrow \begin{bmatrix} a & a \\ e & a \end{bmatrix} \quad L \rightarrow \begin{bmatrix} f & c \\ a & f \end{bmatrix} \quad M \rightarrow \begin{bmatrix} e & c \\ a & d \end{bmatrix}$$

(b) 비재귀적 문법 규칙

$$a \rightarrow \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad b \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad c \rightarrow \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad d \rightarrow \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \quad e \rightarrow \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad f \rightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdots$$

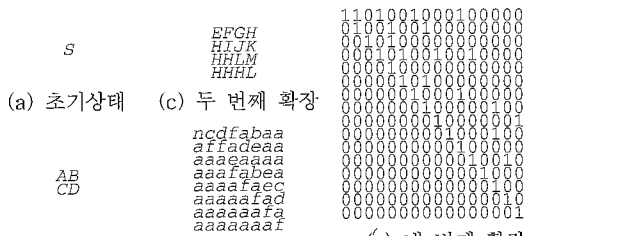
$$n \rightarrow \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad o \rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \quad p \rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

(c) 원시 문법 규칙

그림 5 그림 1의 그래프 구조를 위한 문법 규칙들

그래프 구조의 생성을 위한 문법 규칙은 재귀적 문법 규칙(recursive grammatical rule), 비재귀적 문법 규칙(nonrecursive grammatical rule), 그리고 원시 문법 규칙(primitive grammatical rule)으로 나누어진다. 재귀적 문법 규칙은 문법 규칙의 왼쪽 부분(LHS: left hand side)과 오른쪽 부분(RHS: right hand side)이 각각 하나의 비종결 심볼과 비종결 심볼의 2×2 행렬로 구성된다. 이 문법 규칙은 재귀적 확장을 허용하는 문법 규칙으로서 적은 수의 문법 규칙으로도 복잡한 구조의 그래프 구조를 쉽게 표현한다. 비재귀적 문법 규칙은 문법 규칙의 왼쪽 부분은 비종결 심볼로 구성되고, 오른쪽 부분은 종결 심볼 중 원시 문법 규칙의 왼쪽 부분에 해당되는 심볼('*a*'~'*p*')의 2×2 행렬로 구성된다. 원시 문법 규칙은 그래프 구조 생성을 위한 마지막 확장에 필요한 문법 규칙으로서 문법 규칙의 왼쪽 부분은 종결 심볼로 구성되고 오른쪽 부분은 '0'과 '1'의 2×2 행렬로 구성된다. 그림 1의 그래프 구조인 경우, 재귀적 문법 규칙은 5개, 비재귀적 문법 규칙은 9개이며, 원시 문법 규칙은 '*a*'에서 '*p*'까지 '0'과 '1'의 조합인 16가지 형태를 갖는다.

그림 6은 그림 1의 그래프 구조에 대응되는 연결성 행렬을 얻기 위해서 그림 4의 그래프 구조의 생성 시스템을 이용하여 그림 5의 문법 규칙의 확장 과정을 보여준다.



(a) 초기상태 (c) 두 번째 확장

(b) 첫 번째 확장 (d) 세 번째 확장 (e) 네 번째 확장 (최종상태)

그림 6 문법 규칙의 확장 과정

문법 규칙으로부터 연결성 행렬을 생성하기 위하여 그래프 생성 시스템은 시작 심볼인 '*S*'로부터 시작한다(그림 6(a)). 첫 번째 확장 과정에서 초기 그래프는 그림 5(a)의 첫 번째 규칙의 오른쪽 부분으로 대치되면서 재작성된다(그림 6(b)). 두 번째 확장 과정에서도 각 심볼들은 해당 심볼에 대응되는 문법 규칙들의 오른쪽 부분으로 대치되면서 재작성되고 네 번째 확장 과정을 거치면서 최종 상태인 16×16의 연결성 행렬을 얻게 된다(그림 6(c)~(e)).

4.3 퍼지 소속 함수와 문법 규칙의 코딩

해집단 내의 해개체는 퍼지 소속 함수의 형태를 표현하는 부해개체와 문법 규칙을 표현하는 부해개체로 구성된다.

퍼지 소속 함수를 위한 부해개체는 이진 문자열(binary string)에 따라서 그래프 구조에서 노드의 생성

즉, 퍼지 소속 함수의 개수와 모양을 결정한다. 그림 7은 이진 문자열로 코딩되는 퍼지 소속 함수의 예를 보여준다. 그림 7과 같은 퍼지 소속 함수의 코딩 방법은 (4)에서 정의한 퍼지 소속 함수에 기초하여 유전인자의 값에 의해서 퍼지 소속 함수의 개수 및 모양이 결정된다. 즉, 유전인자가 연속적으로 '1'인 부분에서 제일 처음과 제일 나중에 대응되는 점이 각각 퍼지 소속 함수의 c_l과 c_r이 된다. 퍼지 소속 함수의 폭은 이웃 함수의 중점값과의 관계에 의해서 결정되는데, 유전인자가 연속적으로 '1'인 부분들 사이에 위치한 '0'의 개수에 의해서 표현된다. 그림 7에 의해서 표현되는 퍼지 소속 함수는 유전인자의 값에 따라서 지역적 특성을 반영하는 가우시안 함수와 전역적 특성을 반영하는 사다리꼴 형태를 갖게 된다.

그래프 구조에 대응되는 연결성 행렬을 얻기 위해서 본 논문에서는 직접 코딩 기법 대신에 그래프 생성 시스템에서 필요한 문법 규칙을 코딩하는 문법 코딩 기법을 유전자 알고리즘의 코딩 방법으로 도입하여 이를 최적화시킨다. 문법 규칙은 재귀적 문법 규칙, 비재귀적 문법 규칙, 그리고 원시 문법 규칙으로 분류되는데 원시 문법 규칙은 16가지의 '0'과 '1'의 조합으로 구성되기 때문에 해개체로 코딩되지 않고, 재귀적 문법 규칙과 비재귀적 문법 규칙만이 유전자 알고리즘의 해개체로 코딩된다. 문법 규칙을 위한 부해개체는 그림 8에서 보는 바와 같이 문법 규칙에서 왼쪽 부분의 한 심볼(LHS)과 오른쪽 부분의 네 심볼(RHS)이 순차적으로 연결되어 구성된다.

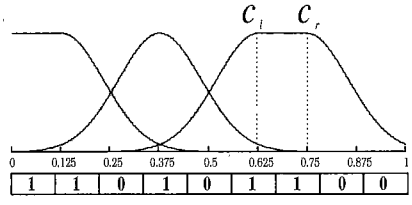
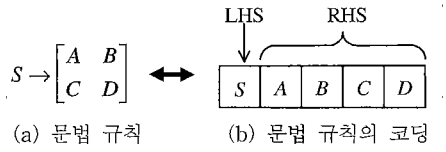


그림 7 퍼지 소속 함수를 위한 부해개체의 코딩



(a) 문법 규칙 (b) 문법 규칙의 코딩

그림 8 문법 규칙을 위한 부해개체의 코딩

유전자 알고리즘은 그래프 구조에 대응되는 연결성 행렬을 얻기 위해서 퍼지 소속 함수와 문법 규칙을 해개체로 코딩하여 최적의 해를 탐색한다. 최적화 대상이 되는 것은 그래프 구조이지만 유전자 알고리즘이 실제로 유전적 진화 탐색을 수행하는 구조는 그래프 구조를 생성시키는 규칙이다. 따라서, 퍼지 소속 함수를 위한 부해개체와 문법 규칙을 위한 부해개체는 유전형(genotype)을 갖는다. 실제 해개체에 대응되는 그래프 구조는 유전형으로부터 만들어지는 연결성 행렬의 해석에 의해서 생성된다. 따라서, 그래프 구조는 표현형(phenotype)의 형태를 갖는다. 연결성 행렬로부터 그래프 구조로의 변환은 이들을 나타내는 유전형과 표현형의 변환이다. 유전형의 최적화는 특별한 형태의 그래프 구조, 예를 들어 모듈화된 구조 등을 다루는데 있어서 적은 문법 규칙으로도 충분히 표현되므로 표현형을 직접 최적화하는 것보다 효율적이다. 또한, 그래프 구조에 대한 진화 과정의 이론적 분석을 위해서도 복잡한 표현형을 직접 분석하는 것보다 단순한 표현 방법을 갖는 유전형이 더 많은 효율성을 보일 것으로 기대된다.

4.4 직접 코딩 기법과 문법 코딩 기법의 이론적 분석

퍼지 시스템 내의 매개 변수를 직접 코딩 기법과 문법 코딩 기법에 의해서 해개체로 표현할 때 요구되는 해개체의 길이를 살펴보자. 입력 공간이 d차원이고 각 입력 공간이 N개의 퍼지 소속 함수로 분할되었다고 가정하자. 각 입력 공간에 대한 퍼지 소속 함수를 위해서 l개의 이진 문자열이 필요하다고 가정할 때 이에 대한 해개체의 길이는 $d \cdot l$이다. 퍼지 소속 함수에 대한 해개체의 길이는 두 방법에 대해서 동일하게 요구되기 때문에 전체 해개체의 길이 비교에서 퍼지 소속 함수를 위한 부해개체의 길이는 고려하지 않는다.

우선 퍼지 규칙의 구조를 종래의 방법인 규칙 테이블의 형태로 표현하였을 때 해개체의 길이를 살펴보자. 규칙 테이블은 모든 입력 공간이 완전히 포함되는 형태를 갖기 때문에 N^d개의 퍼지 규칙이 필요하다. 따라서, 규칙 테이블을 직접 코딩 기법에 의해서 해개체로 표현할 때 요구되는 해개체의 길이는 N^d이다.

다음은 그래프 구조를 직접 코딩 기법과 문법 코딩 기법에 의해서 해개체로 표현할 때 요구되는 해개체의 길이를 살펴보자. 직접 코딩 기법은 연결성 행렬을 직접 코딩하기 때문에 해개체의 길이는 $(d \cdot N+1)^2$이다. 반면, 문법 코딩 기법은 문법 규칙의 개수에 의해서 해개체의 길이가 결정된다. 문법 규칙에 대한 확장의 개수가 1이면 1개의 문법 규칙이 필요하다. 문법 규칙에 대한 확장의 개수가 2이면 문법 규칙의 개수는 첫 번째 확장을 위한 한 개의 문법 규칙과 두 번째 확장을 위한 4개의 문법 규칙이 필요하다. 문법 규칙에 대한 확장의 개

수가 3이면 첫 번째와 두 번째 확장을 위해서 5개의 문법 규칙이 필요하며 세 번째 확장을 위해서는 최악의 경우 16개의 문법 규칙이 필요하다(여기서, 최악의 경우는 재귀적 패턴이 없는 경우이다. 만약 재귀적 패턴이 존재하면 문법 규칙의 개수는 16개 미만이다). i번째 확장에 의한 문법 규칙의 개수는 i-1번째 확장에 필요한 문법 규칙의 개수에 기껏해야 16개의 문법 규칙만이 추가된다($i \geq 3$). 따라서, 문법 규칙의 개수는 $1+4+16(i-2)$ 이다. 하나의 문법 규칙은 5개의 유전 인자가 필요하며 확장에 의한 퍼지 소속 함수의 개수는 $(d \cdot N+1) \leq 2^i$ 이므로 문법 규칙을 해개체로 코딩할 때 요구되는 해개체 길이는 $5 \cdot (16 \cdot \lceil \log_2(d \cdot N+1) \rceil - 27)$이다. 표 1은 해개체 길이에 대한 직접 코딩과 문법 코딩 기법의 복잡도를 보여준다.

표 1 직접 코딩과 문법 코딩 기법의 복잡도

	직접 코딩 기법		문법 코딩 기법
	규칙 테이블	그래프 구조	그래프 구조
해개체의 길이	$O(N^n)$	$O((d \cdot N)^2)$	$O(\lceil \log_2(d \cdot N) \rceil)$

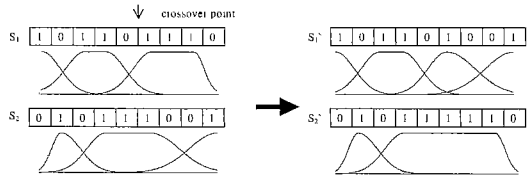
그래프 구조를 위한 문법 코딩 기법은 규칙 테이블 및 그래프 구조를 위한 직접 코딩 기법보다 동일한 퍼지 규칙의 구조를 표현하면서 훨씬 적은 해개체의 길이를 요구한다. 이는 두 코딩 방법이 동일한 탐색 공간을 갖지만 문법 코딩은 입력 공간이 증가함에 따라 직접 코딩보다 적은 해개체를 요구하므로, 입력 공간이 고차원일지라도 비교적 일정한 길이의 해개체를 유지하도록 한다. 또한, 앞 절에서 설명하였듯이 문법 코딩은 모듈화된 구조를 표현할 수 있으므로, 직접 코딩보다 작은 길이의 해개체를 유지하면서도 효율적인 탐색이 가능하다.

5. 유전자 알고리즘에 의한 퍼지 시스템의 최적화

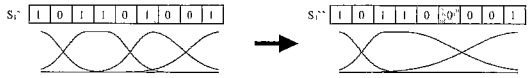
5.1 유전 연산

유전 연산자로는 교배와 돌연변이 연산자를 사용한다. [6]에서 제안된 직접 코딩 기법에 의한 그래프 구조 퍼지 시스템의 설계 방법에서는 그래프 구조의 노드와 에지를 해개체로 코딩할 때 퍼지 소속 함수의 개수에 의해서 정의되는 노드가 가변적이기 때문에 교배 연산자의 적용이 불가능하다. 본 논문에서는 그래프 구조의 노드와 에지를 직접 해개체로 코딩하지 않고 이들을 생성시키는 문법 규칙을 코딩하기 때문에 해개체의 길이가 일정하며 이는 교배 연산자의 적용을 가능하게 한다.

퍼지 소속 함수를 위한 부해개체에 대해서 교배와 돌연변이 연산자의 수행은 새로운 형태를 갖는 퍼지 소속 함수를 탐색하도록 한다. 그림 9는 퍼지 소속 함수를 위한 부해개체의 유전 연산자의 수행 과정을 보여준다.

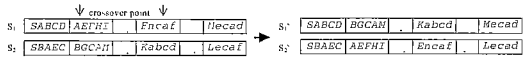


(a) 퍼지 소속 함수를 위한 부해개체의 교배 연산

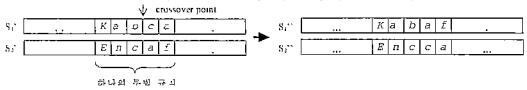


(b) 퍼지 소속 함수를 위한 부해개체의 돌연변이 연산

그림 9 퍼지 소속 함수를 위한 부해개체의 유전 연산



(a) 문법 규칙을 위한 부해개체의 교배(유전형교배)



(b) 문법 규칙을 위한 부해개체의 교배(셀 교배)

(c) 문법 규칙을 위한 부해개체의 돌연변이

그림 10 문법 규칙을 위한 부해개체의 유전 연산

문법 규칙을 위한 부해개체는 하나의 문법 규칙에 대응되는 부해개체 단위로 수행되는 교배 연산(유전형 교배)과 하나의 문법 규칙 내에서 수행되는 교배 연산(셀 교배)이 적용된다. 유전형 교배는 적용 대상이 문법 규칙의 단위로 수행되므로 중요한 문법 규칙을 탐색하는 역할을 수행하고 셀 교배는 문법 규칙 내에서 셀 단위로 교배 연산이 수행되므로 탐색 공간 내에서 특정 문법을 좀 더 세밀하게 탐색하는 역할을 수행한다. 문법 규칙을 위한 부해개체에 적용되는 돌연변이 연산자는 종결 심볼과 비종결 심볼을 무작위로 바꾸어 줌으로써 탐색 공간 내에서 전역적 탐색을 수행한다. 그림 10은 문법 규칙을 위한 부해개체의 유전 연산자의 수행 과정을 보여준다.

문법 규칙에 대한 부해개체는 유전 연산자의 수행에 따라서 같은 문법 규칙이 해개체 내에 여러 개가 존재할 수 있다. 중복된 문법 규칙은 연결성 행렬을 생성하는데 어떤 영향을 끼치지 않는다. 이 경우 본 논문에서

는 해개체 내에 처음으로 존재하는 문법 규칙만이 유효한 것으로 간주한다. 중복된 문법 규칙은 연결성 행렬을 생성하는데 아무런 영향을 끼치지 않을지라도 진화 과정을 거치면서 유전 연산의 적용으로 인하여 다음 세대에서는 다른 문법 규칙으로 바뀔 수 있으며, 유전 연산자의 적용에 의해서 중복된 문법 규칙 중 하나가 파괴될지라도 중복된 문법 규칙 중 남아있는 문법 규칙이 여전히 현 세대에 유지될 수 있으므로 유전 연산에 의해서 우수한 해개체가 파괴되는 현상을 방지할 수 있다.

본 논문에서 사용하는 교차와 돌연변이 연산은 유효하지 않는 그래프 구조의 생성 가능성을 가지고 있다. 이러한 가능성은 퍼지 소속 함수를 위한 부해개체의 유전인자의 값들이 모두 '0'일 때, 재귀적 문법 규칙 혹은 비재귀적 문법 규칙의 오른쪽 부분 모두가 원시 문법 규칙 'a'로 구성됨으로써 이러한 문법 규칙의 확장에 의해서 생성된 연결성 행렬의 원소들이 모두 '0'일 때, 그리고 확장 과정을 통하여 생성된 연결성 행렬 중에서 존재 비트가 모두 '0'인 경우에 발생할 수 있다. 본 연구에서는 유효하지 않는 그래프 구조의 생성 가능성을 배제하기 위하여 유전 연산에 의해서 유효하지 않는 그래프 구조가 생성된 경우 유전 연산의 수행 전 상태로 돌아가서 다시 유전 연산을 수행하는 방법을 사용한다.

5.2 평가 함수

각 해개체에 대한 평가는 유전형이 표현형으로 변환된 그래프 구조에서 수행된다. 퍼지 시스템을 설계할 때 요구되는 사항은 학습 오차가 최소가 되면서 퍼지 규칙의 개수 또한 최소로 만드는 것이다. 본 논문에서는 각 해개체에 대한 평가 함수를 다음과 같이 정의한다:

$$f(x_i) = \frac{1}{\alpha_E \cdot \text{학습 오차} + \alpha_R \cdot \text{퍼지 규칙의 개수}} \quad (6)$$

여기서, $f(x_i)$은 해개체 x_i에 대한 적합도(fitness)를 나타내며, α_E와 α_R은 각각 학습 오차와 퍼지 규칙의 개수에 대한 가중치이다.

5.3 선택 연산

다음 세대에 우수한 해개체들로 구성된 해집단을 얻기 위해서 본 논문에서는 [24]에서 소개한 확률적 균등표본 선택(stochastic universal sampling selection) 연산을 사용한다. 확률적 균등표본 선택 연산자는 해집단의 개체를 부모개체(parent individual), 소멸개체(dead individual), 나머지 개체(remaining individual)로 나눈다. 부모개체는 재생산되어질 개체이고 소멸개체는 낮은 적합도 값을 갖기 때문에 다음 세대로 선택되지 않고 도태되는 개체이다. 그리고, 나머지 개체는 유전 연산을 수행하지 않고 다음 세대의 해집단으로 복제된다. 따라서, 부모 개체와 나머지 개체는 우수한 개체로 진화될 좋은 기회를 갖는다. 확률적 균등표본 선택 연산은 해집단 내의 해개체들의 통계 값에 의존하여 다음 세대를 위한 해집단을 구성하기 때문에 룰렛휠(roulette wheel) 선택 연산에서 발생되는 문제점인 확률에 의존하여 낮은 적합도를 갖는 해개체가 복제되는 현상을 방지할 수 있다. 아울러, 본 논문에서는 다음 세대에 우수한 해개체를 확보하기 위해서 엘리트(elitist) 선택 연산자를 사용한다.

6. 학습 알고리즘

이상에서 설명한 퍼지 시스템의 자동적인 설계 방법은 종래의 퍼지 시스템의 설계에서 부딪치게 되는 문제점의 하나인 모든 입출력 공간을 무조건적으로 받아들이는 퍼지 규칙 대신에 입출력 관계를 적절히 표현하는 선택적 퍼지 규칙으로 구성된 퍼지 규칙 베이스에 기반을 한다. 이를 위해서 퍼지 소속 함수의 개수와 모양 그리고 퍼지 규칙의 구조를 기존의 규칙 테이블이 아닌 그래프 구조로 표현한다. 유전자 알고리즘은 그래프 구조를 최적화하기 위해서 사용되는데 최적화될 퍼지 시스템 내의 매개변수들을 문법 코딩 기법을 이용하여 코딩함으로써 입력 변수의 증가에 따른 탐색 공간의 빠른 증가에 적절히 대처하고 탐색 과정의 효율화를 기하고자 한다. 아울러, 본 논문에서 수행하게 될 자동적인 퍼지 시스템의 설계 방법은 매개변수의 조정과 구조식별의 과정을 동시에 고려함으로서 종래의 방법론보다 체계적이고 효율적인 퍼지 시스템을 생성하도록 할 것이다. 본 논문에서 수행하는 자동적인 퍼지 시스템의 설계 방법에 대한 학습 알고리즘은 다음과 같다:

[단계 1] 모든 해개체 $x_i^t(i = 1, 2, ..., p)$을 랜덤값에 의해 초기화하여 초기 해집단 $P(t) = \{x_1^t, x_2^t, ..., x_p^t\}$를 구성한다($t = 0$).

[단계 2] 퍼지 소속 함수를 위한 부해개체로부터 각 변수에 대한 퍼지 소속 함수의 개수와 모양을 결정한다.

[단계 3] 문법 규칙에 대한 부해개체로부터 변환된 그래프 구조로부터 퍼지 규칙의 전제부를 구성한다. 그리고, 퍼지 규칙의 결론부의 실수값 w_i를 랜덤값으로 초기화한다.

[단계 4] 학습 데이타 ($\vec{a}$, b)를 이용하여 w_i를 LMS 규칙으로 최적화한다.

[단계 5] 해집단 내의 모든 해개체를 평가한다.

[단계 6] 해개체의 적합도에 비례하여 해집단 내의 모든 해개체의 선택 확률을 구한다.

[단계 7] 확률적 균등표본 선택 방법을 이용하여 교배와 돌연변이 연산을 수행할 해개체를 선택한다.

[단계 8] 자식 해개체 x_i^t를 얻기 위해서 해개체 x_i^t에 대해서 교배와 돌연변이 연산을 수행한다.

[단계 9] 자식 해개체와 현 세대의 최우수 개체를 이용하여 다음 세대를 위한 해집단을 구성한다.

[단계 10] 유전자 알고리즘이 수렴할 때까지 또는 최대 세대 반복 횟수만큼 단계 2에서 단계 9를 반복한다.

7. 실험 및 실험 결과

본 논문에서 제안한 문법 코딩을 이용한 퍼지 시스템 설계 방법의 유용성을 보이기 위해서, 제안한 방법을 아이리스 데이타(Iris data) 문제와 시간열 예측(chaotic time series prediction) 문제에 적용하여 제안한 방법의 응용 가능성을 보이고, 직접 코딩을 이용한 퍼지 시스템의 설계 방법과 비교, 분석한다.

7.1 아이리스 데이타 문제

본 실험에서는 제안한 방법의 유용성을 보이기 위해서, 제안한 방법을 고차원 분류 문제로 알려진 아이리스 데이타 문제에 적용하여 응용성을 살펴본다. 아이리스 데이타는 세 개의 클래스(class)로 분류되며 각 클래스는 50개의 데이타로 구성되어 있으며 각 데이타는 사차원의 특징 벡터(feature vector)를 갖는다[1]. 따라서, 4개의 입력 변수와 3개의 출력 변수를 갖는다. 본 실험에서는 150개의 데이타 중 앞의 75개의 데이타는 학습 데이타로서 나머지 75개의 데이타는 시험 데이타로서 사용한다. 각 입력 변수에 대한 퍼지 소속 함수는 [0.0, 7.0]에서 정의되도록 하였으며, 퍼지 소속 함수에 대한 부해개체의 길이는 각 입력 변수당 9개의 유전인자가 할당되었다. 문법 규칙을 위한 부해개체는 1개의 초기 문법 규칙과 각각 20개의 재귀적 및 비재귀적 문법 규칙을 유전인자로 할당하였다. 실험에 사용된 각 매개변수의 값은 표 2로 요약된다. 유사 연구와 비교를 위해서 본 논문에서 제안한 문법 코딩에 기반한 퍼지 시스템의 설계 방법과 [6]에서 제안된 직접 코딩 기법에 기반한 퍼지 시스템의 설계 방법에 대해서 표 2에서 제시된 매개변수를 동일하게 사용하여 실험을 수행하였다.

표 2 아이리스 데이타 문제를 위한 매개변수

매개변수	값
최대 세대 반복 횟수	100
해집단의 크기	50
부모개체의 개수	40
교배 연산자의 확률	0.3
돌연변이 연산자의 확률	0.1
학습률(η)	0.5
LMS 규칙을 위한 최대 학습 반복 횟수	100
학습 오차에 대한 변화량의 임계값(δ)	0.1×10^{-5}
학습 오차의 가중치(α_E)	0.167
퍼지 규칙 개수의 가중치(α_R)	0.1

100번의 실험이 본 논문에서 제안한 문법 코딩에 기반한 퍼지 시스템과 [6]에서 제안된 직접 코딩에 기반한 퍼지 시스템에 대해서 각기 수행되었다. 그림 11과 그림 12는 각각 평균 적합도 곡선과 학습데이타에 대한 평균 분류 적중률 곡선을 보여준다. 그림 11과 그림 12에서 볼 수 있듯이 문법 코딩에 기반한 퍼지 시스템이 세대수가 증가함에 따라서 직접 코딩에 기반한 퍼지 시스템보다 빠른 수렴 속도를 보여준다.

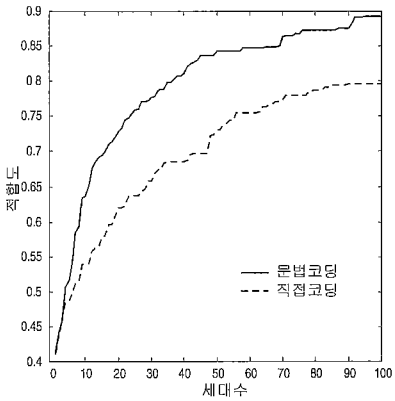


그림 11 아이리스 데이타 문제에 대한 평균 적합도 곡선

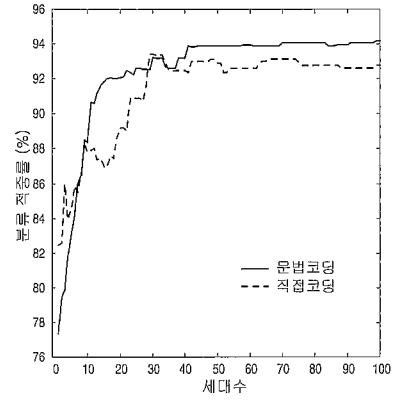


그림 12 아이리스 데이타 문제에 대한 학습 데이타의 평균 분류 적중률 곡선

표 3 아이리스 데이타 문제에 대한 결과 비교

		직접 코딩	문법 코딩
퍼지 규칙		4 (4.71)	3 (3.12)
퍼지 소속 함수		8 (9.18)	5 (4.94)
분류 적중률	학습 데이타	93.33 % (92.78 %)	96.00 % (94.19 %)
	시험 데이타	97.33 % (96.16 %)	98.67 % (96.55 %)

표 3은 두 방법의 결과 비교를 보여주며, 100번의 실험 중 가장 좋은 성능을 보이는 퍼지 시스템의 결과이다(표 3에서 괄호 안의 수치는 100번의 수행에 대한 평균값이다). 표 3에서 볼 수 있듯이, 문법 코딩에 기반한 퍼지 시스템은 직접 코딩에 기반한 퍼지 시스템보다 적은 수의 퍼지 소속 함수와 퍼지 규칙을 생성하였음에도 불구하고 학습 및 시험 데이타에 대해서 더 정확한 분류 적중률을 보여준다.

문법 코딩에 기반한 퍼지 시스템은 100번의 수행 중 가장 좋은 성능을 갖는 퍼지 시스템의 경우 그림 13과 같이 9개의 퍼지 소속 함수를 생성하였으며, 그림 14는 유전적 진화 과정을 통하여 생성된 문법 규칙들을 보여준다. 그림 14의 문법 규칙에서 동일한 문법 규칙이 연결성 행렬의 구성을 위해서 각 문법 규칙의 오른쪽 부분에 중복되어 사용되었다. 문법 규칙의 오른쪽 부분에서 중복된 심볼들은 연결성 행렬에서 이들 심볼로부터 확장되는 구조가 중복되어 표현된다. 따라서, 중복된 심볼들은 모듈화된 구조를 나타낼 수 있으며, 연결성 행렬 자체를 해개체로 코딩하는 것보다 단순하게 그 구조를 표현할 수 있다. 예를 들어, 그림 14에서 재귀적 문법 규칙 "G B B F F"는 심볼 'B'와 'F'를 각각 비재귀적 문법 규칙 "B l e j p"와 "F a f o i"의 오른쪽 부분으로 대치하는 문법 규칙으로서, 심볼 'B'와 'F'로부터 확장된 구조는 최종적으로 생성되는 연결성 행렬에서 서로 같은 구조로 표현된다. 직접 코딩 기법에 의한 연결성 행렬의 코딩에서는 이 부분에 대한 코딩을 구조에 대한 연관성 없이 그 자체를 해개체로 코딩한다. 따라서, 중복된 구조일지라도 해개체에 서로 다른 유전인자로 코딩됨으로써 규칙성을 갖는 퍼지 규칙의 구조를 중복적으로 탐색하여 유전자 알고리즘의 탐색 과정을 복잡하게 만든다. 반면, 문법 코딩에서는 모듈화된 구조를 나타내는 부분을 재귀적 문법 규칙으로 단순하게 표현함으로써, 불필요한 공간에서의 탐색을 줄일 수 있다.

그림 15는 그림 14의 문법 규칙의 확장에 의해서 생성되는 그래프 구조를 보여준다. 그림 15의 그래프 구조로부터 대응되는 퍼지 규칙은 그림 16과 같이 3개가 생성된다. 본 논문에서 제안한 문법 코딩에 기반한 퍼지

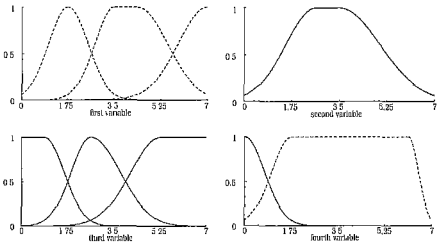


그림 13 아이리스 데이타 문제를 위한 퍼지 소속 함수

초기 문법 규칙 :
S A D I G
재귀적 문법 규칙 :
A B I C H, D J G J I, G B B F F,
I E I D J
비재귀적 문법 규칙 :
B l e j p, C d c l p, D n k k o,
E b m b o, F a f o i, G a p h a,
H p e j p, I b l b k, J d n b o

그림 14 아이리스 데이타 문제를 위한 문법규칙

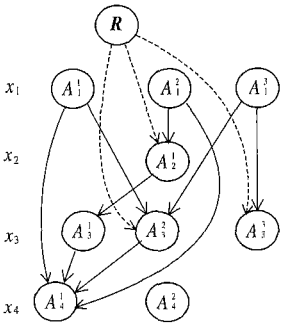


그림 15 그림 14의 문법 규칙의 확장에 의해서 생성된 그래프 구조

If x_2 is A_2^1 and x_3 is A_3^1 and x_4 is A_4^1
then $y_1 = 0.9245, y_2 = -0.0713, y_3 = 0.0042$
If x_3 is A_3^2 and x_4 is A_4^1
then $y_1 = 0.0038, y_2 = 0.9141, y_3 = -0.0820$
If x_3 is A_3^3
then $y_1 = -0.0104, y_2 = -0.0355, y_3 = 0.9542$

그림 16 그림 15의 그래프 구조에 대응되는 퍼지 규칙

시스템은 9개의 퍼지 소속 함수와 3개의 퍼지 규칙을 생성하였다. 제안한 방법은 9개의 퍼지 소속 함수가 생

성될지라도 불필요한 입력 변수와 퍼지 소속 함수가 유전자 알고리즘의 진화 과정을 통하여 도태되어 입력 공간에 강한 영향을 미치는 퍼지 규칙만을 선택적으로 생성한다. 그림 15의 그래프 구조에서 첫 번째 입력 변수에 대한 모든 퍼지 소속 함수(A_1^1, A_1^2, A_1^3)와 네 번째 입력 변수에 대한 두 번째 퍼지 소속 함수(A_4^2)는 입력 공간에 영향을 끼치지 못하므로 퍼지 규칙의 생성시 선택되지 않았다. 이들 퍼지 소속 함수들 중 A_1^1과 A_4^2는 그림 14의 문법 규칙의 확장에 의해서 생성된 연결성 행렬에서 노드의 존재를 나타내는 비트가 '0'으로 생성되어 퍼지 규칙에 포함되지 않는다. 또한, A_1^2와 A_1^3는 연결성 행렬로부터 퍼지 규칙으로의 해석시 루트 노드로부터 에지가 존재하지 않기 때문에 선택되지 못하였다. 따라서, 초기에 생성된 퍼지 소속 함수의 수는 9개이지만, 문법 규칙의 확장과 그래프 구조의 해석에 의해서 최종적으로 선택되는 퍼지 소속 함수의 수는 5개이다. 그림 13의 퍼지 소속 함수에서 점선으로 표시된 퍼지 소속 함수는 퍼지 규칙의 구성 시 선택되지 않는 퍼지 소속 함수를 나타낸다.

7.2 시간열 예측 문제

본 실험에서 시간열 예측 문제로서 사용되는 Mackey-Glass chaotic time series 문제는 다음과 같이 정의된 지연 미분 방정식(delay differential equation)에 의해서 생성된다[1, 8]:

$$\frac{dx(t)}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \qquad (7)$$

시간열 예측 문제는 과거의 값들, 즉 $\{x(t), x(t-\Delta t), \cdots, x(t-(n-1)\Delta t)\}$로부터 미래의 값, 즉 $x(t+P)$를 예측하는 문제이다. P에 대한 적당한 Δt와 n의 선택에 따라서 입력 변수의 개수와 문제의 성질이 결정되는데, 본 실험에서는 $P=\Delta t=6$이고 $n=6$인 시간열 예측 문제를 다룬다. 각 점에서 시간열을 얻기 위한 방법으로, 4차 Runge-Kutta 방법을 이용하여, (7)의 수치적 해를 발견하였고, 시간 단계는 0.1, 초기 데이타($x(0)$)는 0.8, 그리고, τ는 30을 사용하여 시간 t가 130부터 1129까지 1000개의 데이타를 추출하였다. 이 데이타 중 500개는 학습 데이타, 나머지 500개의 데이타는 시험 데이타로서 사용하였다. 각 입력 변수에 대한 퍼지 소속 함수는 [1.2, 1.4]에서 정의되도록 하였으며, 퍼지 소속 함수에 대한 부해개체의 길이는 각 변수당 9개의 유전인자가 할당되었다. 문법 규칙을 위한 부해개체는 1개의 초기 문법 규칙과 각각 20개의 재귀적 및 비재귀적 문법 규칙을 유전인자로 할당하였다. 실험에 사용된 각 매개변수의 값은 표 2와 같으나, 학습 오차의 가중치(α_E)와 퍼지 소속 함수의 가중치(α_R)로서 각각 15.0과 0.03을 사용하였다. 성능 평가 기준으로서 본 논문에서는 평균 제곱근 오차(root mean squared error)를 데이타의 표준편차로 나눈 값인 NMSE(Normalized Mean Square Error)를 사용한다.

100번의 실험이 본 논문에서 제안한 문법 코딩에 기반한 퍼지 시스템과 직접 코딩에 기반한 퍼지 시스템에 대해서 각각 수행되었다. 그림 17과 그림 18은 각각 시간열 예측 문제에 대한 평균 적합도 곡선과 학습 데이타의 NMSE 곡선을 보여준다. 그림 17은 문법 코딩에 기반한 퍼지 시스템이 세대수가 증가함에 따라서 직접 코딩에 기반한 퍼지 시스템보다 빠른 수렴 속도를 보여준다. 그림 18은 문법 코딩에 기반한 퍼지 시스템이 진화 초기에는 탐색 공간을 광범위하게 탐색하여 직접 코딩에 기반한 퍼지 시스템보다 느린 수렴 속도를 보인다. 그러나, 세대수가 증가함에 따라서 문법 규칙의 재귀적

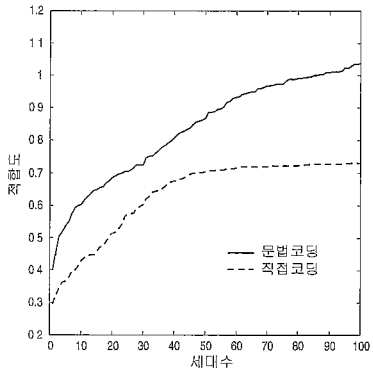


그림 17 시간열 예측 문제에 대한 평균 적합도 곡선

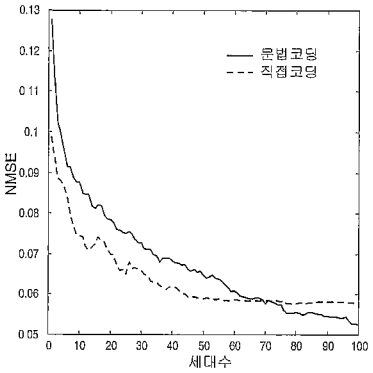


그림 18 시간열 예측 문제에 대한 학습 데이타의 NMSE 곡선

표 4 시간열 예측 문제에 대한 결과 비교

		직접 코딩	문법 코딩
퍼지 규칙		47 (39.50)	27 (34.56)
퍼지 소속 함수		14 (14.25)	12 (13.60)
NMSE	학습 데이타	0.0358 (0.0585)	0.0433 (0.0522)
	시험 데이타	0.2646 (0.2016)	0.1506 (0.1911)

특성을 이용하여 불필요한 탐색 공간에서의 비효율적인 탐색을 배제함으로써 직접 코딩에 기반한 퍼지 시스템보다 빠른 수렴 속도를 보여준다.

표 4는 시간열 예측 문제에 대한 본 논문에서 제안된 방법과 직접 코딩에 기반한 퍼지 시스템의 결과를 보여주며, 100번의 실험 중 가장 좋은 성능을 보이는 퍼지 시스템의 결과이다(표 4에서 괄호 안의 수치는 100번의 수행에 대한 평균값이다). 표 4의 결과는 문법 코딩에 기반한 퍼지 시스템이 직접 코딩에 기반한 퍼지 시스템보다 적은 수의 퍼지 소속 함수와 퍼지 규칙을 생성하였음에도 불구하고 학습 및 시험 데이타에 대해서 훨씬 작은 NMSE를 생성함을 보여준다.

그림 19는 본 논문에서 제안한 방법에 의해서 생성된 예측 결과를 보여준다. 그림 19(a)에서 실선은 시간열 예측 문제의 실제 곡선을 나타내며, 점선은 본 논문에서 제안한 퍼지 시스템을 이용한 예측 모델의 곡선을 나타낸다. 그림 19(b)는 실제값과 예측값의 차이를 나타내는 잔류오차(residual error)를 보여준다. 관련 연구와 비교를 위해서, [6]에서 제안된 직접 코딩에 기반한 퍼지 시스템에 대해서 동일한 데이타를 사용하여 예측을 수행하였다. 그림 20(a)는 실제값과 직접 코딩에 기반한 퍼지 시스템에 의한 예측 결과를 보여준다. 그림 20(b)는 잔류오차를 보여준다.

그림 19는 제안된 방법에 의해서 생성된 예측값과 실제값의 차이가 직접 코딩에 기반한 퍼지 시스템에 의한 예측값과 실제값의 차이보다 적음을 보여준다. 직접 코딩에 기반한 퍼지 시스템은 불필요한 입력 변수와 퍼지 소속 함수를 유전자 알고리즘의 진화 과정을 통하여 도태시켜 최적화된 퍼지 규칙을 추출할 수 있는 구조를 가지고 있음에도 불구하고, 입력 공간의 증가에 따른 해 개체의 길이의 증가 문제를 그대로 안고 있다. 따라서, 유전자 알고리즘의 탐색 공간을 입력 변수에 대한 모든 퍼지 소속 함수의 전체 개수의 크기의 증가에 효율적으로 감소시킬 수 있는 표현 방법 없이 연결성 행렬 그 자체를 해개체로 코딩함으로써 최적의 퍼지 시스템을 얻기 위해서 더 많은 탐색 시간과 계산 복잡도를 요구한다. 반면, 문법 코딩에 기반한 퍼지 시스템은 퍼지 시스템의 구조를 생성시키는 문법 규칙을 해개체로 코딩하므로 입출력 관계를 나타내는 퍼지 규칙의 구조가 단순하게 표현된다. 문법 코딩에 기반한 퍼지 시스템에서는 퍼지 규칙의 구조를 단순하게 표현할 수 있는 문법 규칙을 탐색하므로 유전자 알고리즘의 탐색 공간은 직접 코딩에 기반한 퍼지 시스템보다 복잡도가 줄어든다. 따라서, 동일한 세대 반복 후에 직접 코딩에 기반한 퍼지 시스템보다는 문법 코딩에 기반한 퍼지 시스템이 더 좋은 예측 결과를 갖는 퍼지 시스템을 생성한다.

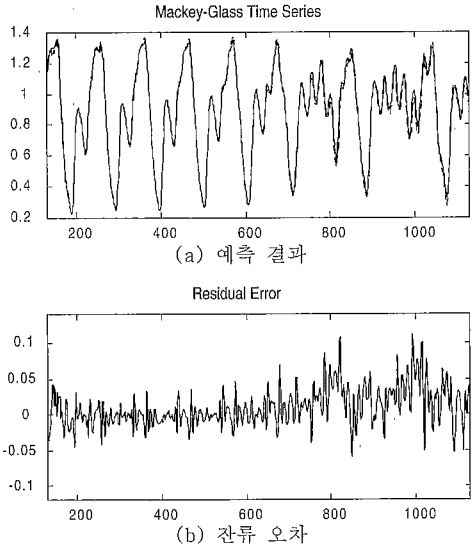


그림 19 문법 코딩에 기반한 퍼지 시스템의 예측 결과

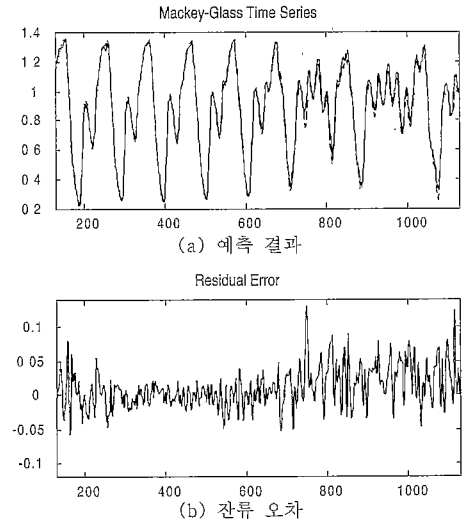


그림 20 직접 코딩에 기반한 퍼지 시스템의 예측 결과

8. 결론 및 향후 연구 과제

본 논문에서는 퍼지 규칙의 구조를 모든 입력 공간을 무조건적으로 포함하는 퍼지 규칙의 규칙 테이블 표현 방법 대신에 퍼지 소속 함수와 퍼지 규칙간의 상호 연관성을 표현하면서 입력 공간에 보다 강건하게 반응하는 선택적 퍼지 규칙에 기반한 그래프 구조의 표현 방법을 제시하였다. 또한, 그래프 구조의 표현 방법이 입력 공간의 증가를 억제할 수 있는 해결의 실마리를 갖고 있음에도 불구하고 유전자 알고리즘의 탐색 공간은 여전히 입력 공간의 증가에 좌우되는 문제점을 해결하기 위해서 해개체의 표현 방법으로서 문법 코딩 기법을 이용하여 탐색 공간의 증가에 적절히 대처할 수 있는 문법 코딩 기법을 이용한 퍼지 시스템의 설계 방법을 제안하였다. 제안된 방법을 벤치마크 문제로 알려진 아이리스 데이타 문제와 시간열 예측 문제에 적용한 실험에서는 제안한 방법이 적은 수의 퍼지 규칙을 이용함에도 불구하고 직접 코딩 방법을 이용한 퍼지 시스템보다 우수한 성능을 보였다.

본 논문에서 제안한 퍼지 시스템의 설계 방법은 다음과 같은 특징을 갖는다: 첫째, 직접 코딩 방법에 의한 해개체의 표현 방법에서는 입력 공간의 증가에 따라 해개체 길이의 폭발적 증가 문제를 가지고 있다. 본 논문에서 제안한 문법 코딩 기법을 이용한 퍼지 시스템의 설계 방법은 퍼지 시스템의 구조를 생성시키는 문법 규칙을 해개체로 코딩함으로써 해개체 길이의 폭발적 증가 문제를 해결하였다. 둘째, 문법 규칙의 확장에 의해서 생성되는 그래프 구조는 적은 수의 문법 규칙만으로도 복잡한 구조를 모듈화된 구조로 단순하게 표현 가능하다. 세째, 직접 코딩 방법이 연결성 행렬 그 자체를 나타내는 표현형의 탐색인 반면, 문법 코딩 방법은 그래프 구조를 생성시키는 문법 규칙을 탐색하는 유전형의 탐색이다. 따라서, 복잡한 구조를 보이는 표현형의 탐색보다는 단순한 형태를 갖는 유전형의 탐색이 진화 과정의 이론적 분석에 효율적일 것이다. 넷째, 본 논문에서 제안한 퍼지 시스템의 설계 방법은 구조식별 단계와 매개변수의 조정 단계가 동시에 수행됨으로써 퍼지 시스템의 부분적 최적화가 아닌 전역적 최적화를 꾀한 퍼지 시스템의 자동적인 설계 방법이다.

본 논문에서 제안된 문법 코딩 기법에 기반한 퍼지 시스템에서 유전자 알고리즘은 퍼지 소속 함수와 퍼지 규칙의 구조를 생성시키는 문법 규칙을 나타내는 유전형을 탐색한다. 복잡한 표현형의 탐색보다는 단순한 표현 형태를 갖는 유전형이 진화 과정의 이론적 분석을 위해서 효율적일 것이다. 앞으로, 유전형의 진화 과정에 대한 이론적 분석 및 분석 결과를 유전자 알고리즘의 진화 과정에 반영하는 연구가 수행되어야 할 것이다.

참 고 문 헌

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

[2] L. X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice-Hall, 1994.

[3] J. Liska and S. S. Melsheimer, "Complete design of fuzzy logic systems using genetic algorithms," *Proc. of the 3rd IEEE Int'l Conf. on Fuzzy Systems*, pp. 1377-1382, 1994.

[4] K. Shimojima, T. Fukuda, and Y. Hasegawa, "Self-tuning fuzzy modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithm," *Fuzzy Sets and Systems*, Vol. 71, pp. 295-309, 1995.

[5] 길준민, 정창호, 강성훈, 박주영, 박대희, "수목구조 지능시스템을 이용한 고차원 공간 위에서의 비선형근사", *한국 퍼지 및 지능시스템 학회 논문지*, 제 6권, 제 3호, pp. 25-36, 1996.

[6] 길준민, 박대희, 박주영, "유전자 알고리즘을 이용한 고차원 입력공간을 갖는 퍼지 시스템의 설계", *한국 정보 과학회 논문지*, 제 24권, 제 4호, pp. 403-412, 1997.

[7] J. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 23, No. 3, pp. 665-685, 1993.

[8] J. R. Jang and C. T. Sun, "Predicting chaotic time series with fuzzy if-then rules," *2nd IEEE Int'l Conf. on Fuzzy Systems*, pp. 1079-1084, 1993.

[9] C. L. Karr, "Design of an adaptive fuzzy logic controller using a genetic algorithm," *Proc. of 4th Int'l Conf. on Genetic Algorithms*, pp. 450-457, 1991.

[10] H. Nomura, I. Hayashi, and N. Wakami, "A learning method of fuzzy inference rules by descent method," *Proc. of the IEEE Int'l Conf. on Fuzzy Systems*, pp. 203-210, 1992.

[11] H. Nomura, I. Hayashi, and N. Wakami, "A self-tuning method of fuzzy reasoning by genetic algorithm," in *Fuzzy Control Systems*, A. Kandel and G. Langholz, Editors. Baca Raton, FL.: CRC Press, pp. 337-354, 1994.

[12] D. Park, A. Kandel, and G. Langholz, "Genetic-based new fuzzy reasoning models with application to fuzzy control," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 24, No. 1, pp. 39-47,

1994.
[13] D. Park, A. Kandel, and G. Langholz, "Hybrid neural-fuzzy reasoning model with application to fuzzy control," in *Fuzzy Control Systems*, A. Kandel and G. Langholz, Editors. Boca Raton, FL.: CRC Press, pp. 355-379, 1994.
[14] P. Thrift, "Fuzzy logic synthesis with genetic algorithms," *Proc. of 4th Int'l Conf. on Genetic Algorithms*, pp. 509-513, 1991.
[15] L. X. Wang and J. M. Mendel, "Backpropagation fuzzy systems as nonlinear dynamic system identifiers," *Proc. of IEEE Int'l Conf. on Fuzzy Systems*, pp. 1409-1418, 1992.
[16] T. C. Chin and X. M. Qi, "Genetic algorithms for learning the rules base of fuzzy logic controller," *Fuzzy Sets and Systems*, Vol. 97, pp. 1-7, 1998.
[17] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," *IEEE Trans. on Fuzzy Systems*, Vol. 3, No. 2, pp. 129-139, 1995.
[18] M. A. Lee and H. Takagi, "Integrating design stages of fuzzy systems using genetic algorithms," *Proc. of IEEE Int'l Conf. on Fuzzy Systems*, pp. 612-617, 1993.
[19] K. Tanaka, M. Sano, and H. Watanabe, "Modeling and control of carbon monoxide concentration using a neuro-fuzzy technique," *IEEE Trans. on Fuzzy Systems*, Vol. 3, No. 3, pp. 271-279, 1995.
[20] A. Kuehlmann and L. Ginneken, "Grammar-based optimization of synthesis Scenarios," *Proc. of IEEE Int'l Conf. on Computer Design: VLSI in Computers and Processors*, pp. 20-25, 1994.
[21] P. Wyard, "Context free grammar induction using genetic algorithms," *IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives*, pp. P/11/1-5, 1993.
[22] F. Gruau, "Cellular encoding as a graph grammar," *IEE Colloquium on Grammatical Inference: Theory, Applications and Alternatives*, pp. 17/1-10, 1993.
[23] H. Kitano, "Neurogenetic learning: an integrated method of designing and training neural networks using genetic algorithms," *Physica D*, Vol. 75, pp. 225-238, 1994.
[24] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, 1994.
[25] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*, Springer-Verlag, 1996.

길 준 민

1994년 고려대학교 자연과학대학 전산학과 졸업(학사). 1996년 고려대학교 대학원 전산학과 졸업(이학석사). 2000년 고려대학교 대학원 컴퓨터학과 졸업(이학박사). 관심분야는 퍼지 시스템, 유전자 알고리즘, 셀룰러 오토마타, 이동 컴퓨팅 등임.

고 명 숙

1989년 이화여자대학교 물리학과 졸업(학사). 1993년 고려대학교 대학원 컴퓨터학과 졸업(이학석사). 1998년 고려대학교 대학원 컴퓨터학과 졸업(이학박사). 1999년 ~ 현재 고려대학교 정보통신기술공동연구소 연구조교수. 관심분야는 퍼지 및 신경망 이론, 유전자 알고리즘, 셀룰러 오토마타, 인공 생명 등임.

황 종 선

1978년 Univ. of Georgia, Statistics and Computer Science 박사. 1978년 South Carolina Lander 주립대학교 조교수. 1981년 한국표준연구소 전자계산실 실장. 1995년 한국정보과학회 회장. 1982년 ~ 현재 고려대학교 컴퓨터학과 교수. 1996년 ~ 현재 고려대학교 컴퓨터과학기술대학원 원장. 관심분야는 알고리즘, 분산시스템, 데이타베이스 등임.