論文
2-3-12

퍼지 적응 제어를 이용한 프로그램 볼륨 복잡도 측정 (Measurement of program volume complexity using fuzzy self-organizing control)

김 재 웅*
(Jae-Woong Kim)

요 약

소프트웨어 척도는 소프트웨어를 특징화하기 위한 효율적인 방법을 제공한다. 척도들은 전통적으로 식의 정의를 통해 구성되었지만 이러한 접근은 소프트웨어를 특징화해주는 모든 매개변수사이의 모든 관계를 완전하게 이해할 수 있을 때만 적용가능하다는 한계가 있다. 이 논문은 비선형 함수를 매우 정교하게 근사화시키고, 인지 심리 이론을 결합할 수 있는 퍼지 논리 시스템을 사용하였다. 우선, Java 프로그램으로부터 수집된 12개 척도들의 인자들로부터 다중 회귀식을 추출하였다. 또한 그 중에서 프로그램 볼륨 인자에 인지 심리 이론을 적용하고 퍼지 학습을 수행하여 프로그램 볼륨 복잡도를 측정하였다. 이 연구는 소프트웨어 척도의 분석과 설계의 더 나은 조사를 위한 토대가 될 수 있을 것이다.

ABSTRACT

Software metrics provide effective methods for characterizing software. Metrics have traditionally been composed through the definition of an equation, but this approach restricted within a full understanding of every interrelationships among the parameters. This paper use fuzzy logic system that is capable of uniformly approximating any nonlinear function and applying cognitive psychology theory. First of all, we extract multiple regression equation from the factors of 12 software complexity metrics collected from Java programs. We apply cognitive psychology theory in program volume factor, and then measure program volume complexity to execute fuzzy learning. This approach is sound, thus serving as the groundwork for further exploration into the analysis and design of software metrics.

1. 서론

현재의 컴퓨터 환경에서는 하드웨어의 급속한 발전에 비해 소프트웨어 개발 기술이 상대적으로 낙후되고 소프트웨어 크기와 복잡도가 기하급수적으로 증가하기 때문에 소프트웨어의 개발과 유지보수를 더욱 힘들게 하고 있다. 소프트웨어는 하드웨어와는 달리 변형이 가능하며, 논리적 혹은 자체적인 결함에 의해서뿐만 아니라 더 좋은 품질을 유지하기 위해서 유지보수에 많은 시간과 노력을 필요로 하기 때문에 소프트웨어의 유지보수는 시간이 흐를수록 소프트웨어의 개발에 드는 비용의 상당한 부분을 차지하고 있다. 1970년대에 전체 개발비 중 35~40%를 차지하던 것이, 1990년대에 와서는 70~80%까지 이르고 있어 유지 보수 비용을 관리하는 것이 소프트웨어 공학자들에게 가장 큰 관심사가 되고 있다.

* 정회원 : 전북대학교 전산통계학과 박사과정

논문접수 : 2001. 3. 21.
심사완료 : 2001. 4. 11.

따라서 소프트웨어의 이해 용이성, 변경 용이성, 시험 용이성을 정량적으로 평가하여 개발된 소프트웨어를 유지보수하는데 얼마만큼의 노력이 필요한가를 예측하는데 도움을 주는 소프트웨어 복잡도 척도에 대한 연구가 지난 30여년간 수행되어 왔다[1]. 최근까지 전통적인 소프트웨어 복잡도 척도들을 확장하거나, 객체지향 특성을 정량화한 척도를 포함하여 많은 객체지향 복잡도 척도들이 제안되었다. 많은 척도들이 식의 정의를 통해 구성되었지만 이러한 접근은 소프트웨어를 특징화해주는 모든 매개변수사이의 모든 관계를 완전하게 이해할 수 있을 때만 적용가능하다는 한계가 있었다. 또한 대부분의 척도들이 단지 복잡도에 영향을 주는 요인의 수를 세는 방법으로 척도를 제안하여, 그 요인 수에 따라 증가하는 인지심리 복잡도를 고려하지 않았다는 문제점이 있었다.

따라서 프로그램 이해와 관련되어 인지 심리 이론을 소프트웨어 공학에 접목시킨 연구들이 많이 행해져왔다. Miller[2]는 실험을 통해 실체를 처리할 수 있는 인간의 단기 기억(short-term memory) 용량이 7±2 덩이(chunk)임을 알아내었다. 또한 내포 구조를 처리하기 위해서는 단기 기억 한계가 3±1 덩이라는 것이 제시되었는데 이는 다른 덩이의 이해가 선행되어야 하므로 더 많은 노력이 필요하기 때문이다. Franck[3]는 프로그램 제어 논리를 처리할 때에는 3±1 덩이에 한계된다는 것을 경험적으로 제시하였다. Gugerty[4]와 Littman[5]은 인간의 단기 기억이 프로그램 이해에 중요한 역할을 하고 있다고 설명하고 있다. 인지심리 이론에 근거를 둔 지침들에서는 프로그램 구성성분들은 실세계를 표현한 덩이이므로, 소프트웨어 개발이나 복잡도 측정에 있어서 인간의 단기 기억의 한계 7±2와 3±1을 고려하여야 하며, 특히 복잡도 측정에서는 소프트웨어 규모, 복잡도 그리고 난이도 등에 영향을 주는 요인 수 증가에 따라 가중되는 인지심리 복잡도를 반영해야 한다고 주장하고 있다.

퍼지 시스템은 compact 공간상에서 어떠한 비선형 함수도 매우 작은 오차 내에서 근사할 수 있다는 것이 증명되어 있고, 퍼지화 과정에서 인지 심리 복잡도를 반영할 수 있다. 본 논문에서는 퍼지 적응 제어 방법을 이용하여 Java 프로그램의 볼륨 복잡도를 측정할 수 있는 모델을 개발하였다. 우선 널리 사용되고 있는 12개의 척도들에 대해 통계적 분석을 수행하여 프로그램의 특징인 인자들을 추출하였다. 사용된 척도들은 Briand[6]가 제안한 속성을 만족하는 척도들과 Java 언어의 특징인 내부 클래스(inner class)를 반영한 척도와 크기 척도이다. 추출한 인자들에 대해 단계적 변수투입법(stepwise method)으로 회귀 분석을 수행하여 다중공선성이 있는 척도들을 제거한 후, 인자들에서 가장 큰 설명력을 가지는 프로그램 볼륨에 대해 인지 심리 이론을 반영한 퍼지 학습을 수행하여 프로그램의 볼륨 복잡도를 측정하는 모델을 개발하였다.

본 논문의 구성은 다음과 같다. 2장에서는 인자분석과 다중 회귀 분석, 퍼지 이론과 인지 심리 이론에 대해 설명하고, 3장에서는 측정할 소프트웨어 척도에 대한 정의와 자료 분석 환경과 방법론 및 통계분석 결과를 살펴본다. 4장에서는 본 연구에서 적용한 프로그램 볼륨 측정 퍼지 시스템을 나타내고, 5장에서 결과 평가를 수행하고, 마지막으로 6장에서 결론을 기술한다.

2. 관련연구

2.1 통계적 분석

많은 소프트웨어 척도들이 유사한 구조를 측정하고 있고, 일부 척도들은 매우 높은 상관관계를 가지고 있다. 그러므로 척도들간의 관련성을 규명하는 것이 가장 의미있는 척도를 선택하는데 있어서 중요하다.

인자 분석은 변수들간의 상관관계를 고려해서 이들 측정치사이에 공유하는 구조를 파악해 내는 기법으로 연구자들이 주어진 많은 정보에 대해 분석이 용이하도록 적은 수의 인자(factor)로 제시해 주는 분석 방법이다. 최초의 인자 분석 행렬은 변수와 인자간의 관계가 명확하게 나타나지 않으므로 변수가 인자에서 무시해도 좋은가 중요한 영향이 있는가를 명확히 보여주기 위하여 회전된 구성요소를 고려한다. 회전을 수행하는데는 배리맥스(varimax) 회전 방법이 가장 빈번하게 사용된다. 인자는 고유값이 1보다 큰 것으로 추출하고, 보통 첫 번째 인자가 전체 변화량의 가장 큰 비율을 설명한다. 인자 분석을 수행하기 위한 표본의 수는 50개 이상이어야 하며 일반적으로 측정 변수의 4~5배가 필요하다[7]. 인자 분석에서

얻은 인자 값을 다중 회귀 분석에서 사용할 수 있다.

다중 회귀 분석(multiple regression analysis)은 변수들의 상호관계를 분석하고 독립변수의 변화로부터 종속변수의 변화를 예측하기 위해 사용된다. 다중회귀 분석을 수행할 경우 포함된 독립변수들이 높은 상관관계를 가질 경우에는 한 변수의 설명력이 다른 변수에 의해 흡수되는 다중공선성(multicollinearity)이 발생하기 때문에 추정된 계수가 통계적으로 유의하지 않게 나타날 가능성이 높다. 다중공선성의 발생을 방지하기 위해서는 미리 변수들간의 상관관계를 검토하여 상관관계가 높은 두 변수중 하나를 제거하거나, 단계적 변수투입법(stepwise method)을 사용하여 상관관계가 높은 변수들 중 가장 설명력이 있는 독립변수만을 분석에 포함시켜야 한다[7].

2.2 퍼지 이론

퍼지논리시스템은 지식베이스(knowledge base), 퍼지화(fuzzifier), 비퍼지화(defuzzifier) 그리고 퍼지 추론기관(fuzzy inference engine)의 네 부분으로 구성된다[8]. 여기서 지식베이스 부분의 형태에 따라서 크게 MISO(Multi-Input Single Output) 구조와 MIMO (Multi-Input Multi-Output) 구조로 나눌 수 있으며 각각 If-then 형태의 규칙으로 구성된다. 네 부분의 구성요소들에 대한 구성도는 [그림 1]과 같다.

본 논문의 지식베이스(knowledge base)는 다음과 같은 MISO의 형태를 갖는 If-then규칙으로 이루어져 있다.

$$R = \bigcup_{l=1}^{M} R_l \tag{1}$$

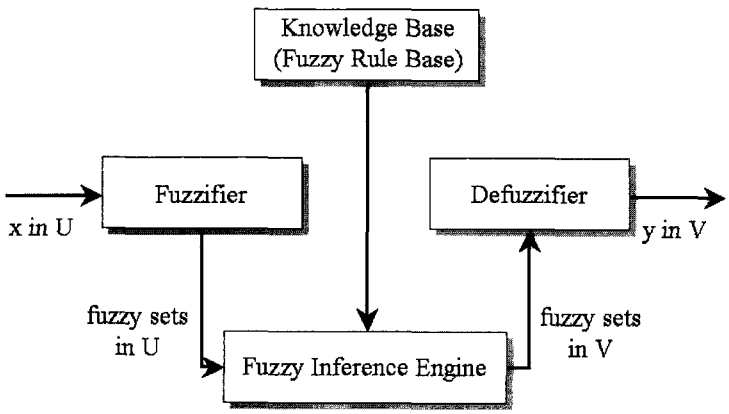
여기서 $\cup$ 은 논리합을 나타내며 각각의 퍼지규칙 R_l 는 다음과 같다.

$$R_l : If\ x_1\ is\ A_1^l,\ x_2\ is\ A_2^l, \cdots,\ x_n\ is\ A_n^l,\ then\ y\ is\ B^l \tag{2}$$

여기서

$$U = U_1 \times U_2 \times \cdots \times U_n,\ U_i \subset R,\ \text{for } i = 1, \cdots, n,$$
$$V \subset R \tag{3}$$

이며 입력벡터와 출력은 각각 $x = (x_1, x_2, \cdots, x_n)^T \in U$, $y \in V$이다. 또한 A_i^l은 입력공간 U_i 상에 정의된 퍼지집합이고, B^l은 출력공간 V 상에 정의된 퍼지집합이다. 각각의 소속정도는 $\mu_{A_i^l}(x_i)$과 $\mu_{B^l}(y)$로 나타낸다. M은 퍼지규칙 전체의 수를 나타낸다. 예를 들어 각 입력변수 상에 r개의 퍼지집합을 정의한다면 $M = r^n$이다.



[그림 1] 퍼지 논리 시스템의 기본 구성도

[Fig.1] Basic configuration of fuzzy logic system

퍼지 추론기관(fuzzy inference engine)은 퍼지규칙 베이스의 퍼지 If-then 규칙과 추론의 조합규칙을 이용하여 입력공간 U에서의 퍼지집합이 출력공간 V의 퍼지집합으로 변환(mapping)을 수행한다.

퍼지화(fuzzification)는 어떤 상태값 x'에 대해서 퍼지집합 $A' \in U$으로의 변환을 수행한다. 일반적으로 이러한 사상에는 단일값과 퍼지값의 두 가지 방법이 있으나 본 논문에서는 다음과 같은 단일값 퍼지화를 사용하였다.

$$\mu_{A'}(x') = \begin{cases} 1 & \text{for } x' = x \\ 0 & \text{for } otherwise \end{cases}, \quad \text{for} x' \in U. \tag{4}$$

비퍼지화(defuzzification)는 출력공간 V 내의 퍼지집합으로부터 출력공간 V 내의 단일값 y로의 변환을 수행한다. 일반적으로 이러한 비퍼지화의 방법에는 최대값방법, 최대평균법, 무게중심법, 개선된 무게중심법등이 있으나 본 논문에서는 다음과 같은 무게중심법을 사용하였다.

$$y = \frac{\sum_{l=1}^{M} \bar{y}^l \cdot \mu_{B'}(\bar{y}^l)}{\sum_{l=1}^{M} \mu_{B'}(\bar{y}^l)}. \tag{5}$$

여기서 $\bar{y}^l$은 퍼지집합 B'의 중심값이다. 즉, $\bar{y}^l$는 $\mu_{B'}(y)$가 가지는 소속정도중에서 가장 큰 값을 가지는 점에 대응되는 V내의 실수값을 나타낸다.

연산으로 대수곱(algebraic product)을 사용하고 퍼지 조건명제로 퍼지 conjuction방법을 사용하며 또한 단일값 퍼지화, 무게중심법 비퍼지화를 이용한 퍼지 논리시스템은 다음과 같이 표현될 수 있다.

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l [\Pi_{i=1}^{n} \mu_{A_i^l}(x_i)]}{\sum_{l=1}^{M} [\Pi_{i=1}^{n} \mu_{A_i^l}(x_i)]} \tag{6}$$

여기서 $\bar{y}^l$은 μ_B^l이 최대가 되는 값으로 $\mu_B^l(\bar{y}^l) = 1.0$을 가정하였다.

이와 같은 퍼지 논리 시스템을 학습 알고리즘을 이용하여 개발하기 위해서는 소속 함수 $\mu_{A_i^l}(\cdot)$를 정의해야 하는데 본 논문에서는 퍼지논리시스템을 이용한 퍼지 적응제어시에 적응기능을 부여하기 위하여 다음과 같은 가우시안(Gaussian) 함수를 선택하였다.

$$\mu_{A_i^l}(x_i) = a_i^l \exp\left[-\left(\frac{x_i - \bar{x}_i^l}{\delta_i^l}\right)^2\right] \tag{7}$$

여기서 $\bar{x}_i^l$은 x_i에 대한 퍼지집합 A_i^l의 소속정도가 1.0인 부분에 대응되는 U_i내의 한 점으로 R_l 규칙에서 U_i에 설정한 퍼지집합 A_i^l의 위치를 나타내는 값이다.

(6)에서 (7)과 같은 가우시안 형태의 소속함수를 사용한다면 퍼지논리시스템은 다음과 같이 표현될 수 있다.

$$f(x) = \frac{\sum_{l=1}^{M} \bar{y}^l \left[\Pi_{i=1}^{n} a_i^l \exp\left(-\left(\frac{x_i - \bar{x}_i^l}{\delta_i^l}\right)^2\right)\right]}{\sum_{l=1}^{M} \left[\Pi_{i=1}^{n} a_i^l \exp\left(-\left(\frac{x_i - \bar{x}_i^l}{\delta_i^l}\right)^2\right)\right]} \tag{8}$$

2.3 인지 심리 이론

유지보수 작업 중 약 50% 이상의 노력이 프로그램을 이해하는데 드는 것으로 알려져 있기 때문에 프로그래머가 어떻게 프로그램을 이해하는가 하는 인지과정(mental process)을 아는 것은 매우 중요하다.

G. Greeno[9]와 R. Mayer[10]등이 주장한 바에 의하면 프로그램 의미의 인지 과정은 한마디로 덩이화(chunking) 과정이라 할 수 있다. 숙련된 프로그래머와 미숙련자를 비교한 경험적 연구에 의하면 덩이는 프로그램을 이해하는 과정에 있어서 매우 중요한 역할을 한다. 프로그래머는 프로그램을 한 줄씩 읽어 이해하는 것이 아니라 하나의 통일된 기능을 위한 프로그램의 의미 있는 덩이를 발견하고 그 의미를 나름대로 파악한 후에는 덩이를 이루는 모든 실행문을 잊어버리고 기능만을 나타내는 간단한 어휘로 기억하게 된다는 것이다. 작은 프로그램의 덩이들이 계속 모아지면 또 다른 차원의 통일된 의미의 덩이를 이루게 된다. 이렇게 덩이화 작업이 계속되면서 프로그래머는 전체 프로그램을 이해할 수 있게 되는 것이다. 프로그램 이해의 과정이 이렇게 진행되는 이유

는 G. Miller[2]가 단기 기억에서의 기억 폭(memory span)은 7±2개의 덩이라고 제안한 바와 같이 인간의 단기 기억(short-term memory)에는 한계가 있어서 생소한 프로그램을 접하였을 때 기억할 수 있는 양은 매우 적기 때문이다.

프로그램의 구성 성분에는 프로그램 라인 수나 메소드 수, 변수 수 같은 단순 평면구조를 가지는 것과 내포 블록처럼 하나의 덩이를 이해하는데 다른 덩이의 이해가 선행되어야 하는 상속이나 메소드 호출같은 내포 구조를 가지는 것이 있다. 전자의 덩이를 평면 덩이라 하고, 후자의 덩이를 내포 덩이라고 한다. 평면 덩이를 인지하는데는 Miller의 7±2가 사용되지만 내포 덩이를 인지하기 위해서는 더 많은 노력이 필요하기 때문에 3±1이 적용된다. 또한 프로그램의 제어 논리를 이해할 때도 3±1이 적용된다.

소프트웨어의 복잡도 척도를 제안하는데 있어서 심리적 복잡도에 영향을 주는 요소들을 간과할 수는 없으므로 객체들의 구성 및 객체들간의 관계에 따라 인지심리적 복잡도를 고려한 가중치가 적용되어야 한다.

3. 실험

3.1 데이터 수집

최근 몇 년 동안 소프트웨어 척도에 대해 요구되는 속성을 제공하는 논문이 여러 편 발표되었다[14, 15, 16]. 이러한 연구들은 기존에 제안된 척도와 새롭게 제안한 소프트웨어 척도를 검증하기 위해 사용되었다. 그러나 주관적인 기준에 의해 정의된 소프트웨어 복잡도 척도들이 많기 때문에 아직도 그 척도들에 대한 타당성 논의가 계속되고 있다. 품질평가에 필요한 메트릭스를 적용하기 위해서는 일관성있는 개념 정의가 중요하다. 본 연구에서는 코드와 관련있는 크기, 길이, 복잡도, 응집도, 결합도 등의 개념에 대해 Briand[6]가 제안한 속성을 만족하는 척도를 측정 대상으로 하였다. 측정한 척도가 <표 1>에 서술되어 있다.

퍼지 학습을 위해 척도 값을 추출한 대상 프로그램은 썬 마이크로시스템즈사의 JDK 1.2의 awt 패키지에서 220개의 프로그램(AWT 그룹)을 사용하였으며, 프로그램의 크기는 500 SLOC 이하였다. 척도 값의 측정은 Banda의 Java Source Metrics[20]와 Andrew와 Rajesh의 JMetric[21] 등의 자동화 도구를

<표 1> 측정 소프트웨어 척도

<Table 1> Software metrics to measure

척 도	설 명	척도 분류
WMC	클래스내 메소드의 사이클로매틱 복잡도(Cyclomatic Complexity)값의 합[17, 18]	크기(Size)
NOC	자식 클래스의 수[18]	〃
NOM	클래스에 정의된 메소드(Method)의 수	〃
NOPM	클래스에 정의된 공용 메소드(Public Method)의 수	〃
LOC	실행문과 비실행문 모두를 포함하고 주석을 제외한 라인 수	〃
SLOC	실행문만을 고려한 라인 수	〃
DIT	클래스 상속 깊이[18]	길이(Length)
NIM	호출된 메소드(Invoked Method)의 수	결합도(Coupling)
NOCB	메소드에서 참조하는 협력자(Collaborator)의 수	〃
Coh	클래스의 응집도 $Coh = \frac{\text{전체 사용 변수 수}}{\text{메소드 수} \times \text{변수 수}}$ [19]	응집도(Cohesion)
NIC	내부 클래스(Inner Class)의 수	Java 특성
DOIC	내부 클래스(Inner Class)의 상속 깊이	〃

주로 사용하였고, 일부 척도값을 산출하는 프로그램은 C++로 구현하여 사용하였다.

3.2 인자 분석 결과

데이타 집합의 특징을 조사하기 위해 12개의 척도값을 추출한 결과 표준 편차가 평균값보다 큰 척도들이 있었지만 인자 분석에 적합한 표본인가를 검증해주는 KMO 통계량을 구한 결과 0.809로 인자분석에 적합한 표본으로 판단되었다[7]. <표 2>는 회전한 인자행렬을 보여준다.

<표 2> 회전된 인자 행렬

<Table 2> Rotated factor Matrix

척도 \ 인자	인자 1	인자 2	인자 3	인자 4	인자 5
WMC	0.958	0.025	0.023	0.055	0.030
NOC	0.021	-0.027	0.039	-0.042	0.996
NOM	0.873	-0.026	0.291	0.014	0.058
NOPM	0.825	-0.065	0.293	0.023	0.045
LOC	0.941	0.201	-0.030	0.006	0.030
SLOC	0.932	0.164	-0.054	0.017	0.023
DIT	-0.021	-0.013	0.024	0.982	-0.043
NIM	0.888	0.221	-0.010	-0.047	-0.049
NOCB	0.816	0.062	0.162	-0.247	-0.060
Coh	-0.189	-0.145	-0.893	-0.017	-0.041
NIC	0.313	0.854	-0.107	-0.139	0.007
DOIC	-0.047	0.841	0.341	0.116	-0.045
Eigenvalue	5.704	1.585	1.130	1.067	1.012
% of variance	47.532	13.212	9.413	8.895	8.430

수집한 12개의 척도 값으로부터 5개의 인자를 추출하였다. 각 인자와 척도의 교차 부분에 나타난 값이 인자와 특정 척도와의 관계를 보여주고 있다. 각 인자에 나타난 값이 0.5보다 클 때, 해당 척도가 인자에 포함되는 것으로 하였다. 인자 1에는 크기 척도로 분류되었던 WMC, NOM, NOPM, LOC, SLOC 척도와 결합도 척도 NOCB, NIM가 적재되는 것으로 나타났다. 이들 척도는 프로그램 볼륨과 관계가 있다. 인자 2에는 내부 클래스 사용과 관계있는 NIC 척도와 DOIC 척도가 적재되었다. 나머지 3개의 인자는 각각 하나의 척도로 구성되었다. 인자 3에는 COH 척도가 적재되었는데, 이것은 응집도와 관련이 있다. 인자 4에는 상속과 관련된 DIT 척도가 적재되었고, 인자 5에는 자식 클래스의 수와 관련된 NOC 척도가 적재되었다. 이들은 데이터 집합에서 복잡도 측정과 관련하여 각각의 근원적인 구조를 나타낸다고 볼 수 있다. 인자 1은 전체 변화량의 47.5%를 설명하고, 인자 2는 13.2%를 설명하고 있다. 인자 3, 4와 5도 각각 9.4%, 8.9%와 8.4%의 변화량을 설명하고 있다. 인자들은 다중 회귀 분석과 퍼지 시스템을 개발하는데 사용된다.

3.3 다중 회귀 분석 결과

기존의 연구에서는 인자의 중요성이나 척도들의 상관관계에 관계없이 모든 척도들을 대상으로 회귀 분석을 수행하여 인간의 인지 능력을 고려한 소프트웨어의 품질이나 유지보수 노력을 측정하기가 어려웠다. 본 논문에서는 향후 복잡도 척도를 제안할 때 인간의 인지 능력과 인자의 상관관계를 고려한 가중치를 적용하기 위해 인자별로 회귀 분석을 수행하였다. 인자 분석에서 같은 인자에 속한 척도들은 척도들간의 높은 상관관계로 인해 한 척도의 설명력이 다른 척도에 의해 흡수될 수가 있다. 그럴 경우 추정된 계수가 통계적으로 유의하지 않게 나타날 가능성이 있기 때문에 가장 설명력이 있는 척도들만을 분석에 포함하기 위해 인자 값을 종속 변수로 하고 각 인자에 포함된 척도들을 독립 변수로 하여 다중 회귀 분석(multiple regression analysis)을 수행하였다. 분산확대지수의 값이 커질 경우에도 회귀계수의 표준오차가 커지기 때문에 해당 변수를 제거하거나 단계적 변수투입법(stepwise method)을 사용하여 상관관계가 높은 변수들 중 가장 설명력이 있는 독립변수만을 분석에 포함시켜야 한다. 보통 최종 척도 집합의 기준은 분산확대지수 값이 50보다 큰 척도가 존재하지 않는 것이다.

<표 3>은 인자 1에 대해 단계적 변수투입법을 사용하여 회귀분석을 수행한 결과를 보여준다. 두 척도가 제외되었음에도 0.997의 높은 조정 R2(Adjusted R2) 값을 가지는 것으로 나타났다. 이 결과는 인지에 포함된 척도들 중 다중공선성을 가지는 척도를 제외하고도 인자를 충분하게 설명할 수 있다는 것을 나타낸다.

<표 3> 단계적 변수투입법을 사용한 회귀 분석 결과
<Table 3> Result of regression analysis using stepwise method analysis

모 델	비표준화계수	표준화계수	분산확대지수
상수	4.505		
WMC	1.074	0.145	15.335
NOPM	1.861	0.078	2.596
NOCB	1.207	0.027	2.639
NIM	1.180	0.193	4.098
SLOC	2.173	0.615	13.257

4. 퍼지 모델

프로그램 볼륨 복잡도를 측정하기 위해 인자분석을 통해 프로그램 볼륨 인자를 추출하였고, 회귀분석을 통하여 다중 공선성이 있는 척도들을 제거하였다. 나머지 척도들을 가지고 복잡도를 측정하기 위해 앞에서 정의한 퍼지 논리 시스템을 보다 간단한 선형의 식으로 표현하고, 또한 매개변수의 표현도 간단하게 하는 퍼지 기준함수(fuzzy basis function)를 다음과 같이 정의하였다.

$$\xi_l(x) = \frac{\Pi_{i=1}^{n}\mu_{A_i^l}(x_i)}{\sum_{l=1}^{M}\Pi_{i=1}^{n}\mu_{A_i^l}(x_i)} \qquad (10)$$

여기서 $\mu_{A_i^l}(x_i) = a_i^l \exp[-(\frac{x_i - \overline{x}_i^l}{\delta_i^l})^2]$이다.

$\overline{y}^l$, a_i^l, $\overline{x}_i^l$, δ_i^l 중 나머지는 고정으로 하고, $\overline{y}^l$을 적응가능한 매개변수로 놓으면 (8)과 같은 퍼지논리시스템은 다음과 같은 선형의 식으로 표현 할 수 있다.

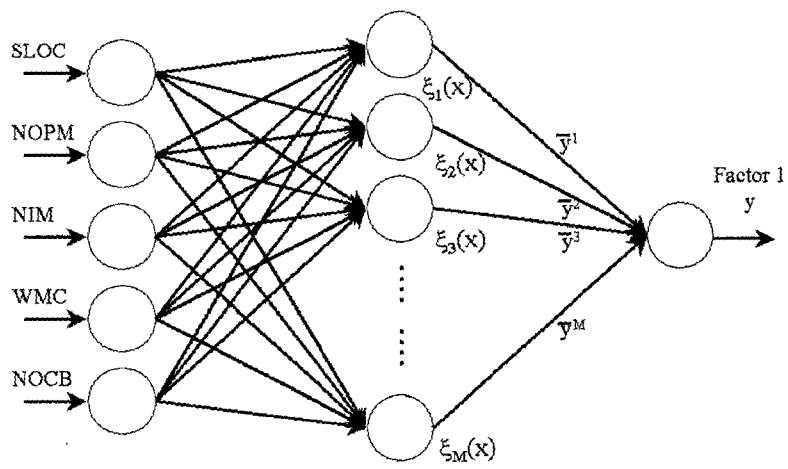
$$f(x) = \sum_{l=1}^{M}\theta_l\xi_l(x) = \theta^T\xi(x) \qquad (11)$$

여기서 $\theta=(\overline{y}^1, \overline{y}^2, \cdots, \overline{y}^M)^T$인 매개변수 벡터이고 $\xi(x)=(\xi_1(x), \xi_2(x), \cdots, \xi_M(x))^T$는 regressive 벡터이다.

이러한 MISO 퍼지논리시스템을 신경회로망구조로 표현하면 [그림 2]와 같이 나타낼 수 있다. 여기서 $\xi_1(x), \xi_2(x), \cdots, \xi_M(x)$는 신경망회로의 은닉층(hidden layer)내의 노드에 대응되며 $\overline{y}^1, \overline{y}^2, \cdots, \overline{y}^M$는 은닉층에서 출력노드로의 가중치(weight)이다.

본 논문에서는 (7)의 가우시안 함수에서 $\overline{x}_i^l$와 δ_i^l 값을 주기 위해 각 척도마다 인지 심리 이론을 적용하였다.

한 클래스에 속한 메소드의 수를 측정하는 NOPM 척도와 협력자의 수를 측정하는 NOCB 척도는 평면 덩이에 해당하므로 인간 단기 기억의 한계로 주장하고 있는 7±2의 덩이 개념을 적용할 수가 있다. 따



[그림 2] 인공신경망 구조로 나타낸 MISO 퍼지 논리 시스템 구조
[Fig. 2] The structure of a MISO fuzzy logic system as the type of neural network

라서 가우시안 함수를 사용한 퍼지화 과정에서 7을 기준으로 하여 퍼지 집합을 정의하였다. 한 클래스의 실행문 수를 측정하는 SLOC 척도의 경우는 한 클래스당 가지는 메소드의 수가 평면 덩이에 해당되므로 7±2 덩이를 적용할 수 있고 또한 메소드당 줄 수 역시 평면 덩이이므로 7±2 덩이를 적용하면 25에서 81의 값을 가지는데 중앙값인 50을 기준으로 하여 퍼지 집합을 정의하였다.

한 클래스에서 호출한 메소드의 수를 측정하는 NIM 척도는 평면 덩이와 내포 덩이 모두에 관계가 있다. 한 클래스당 가지는 메소드의 수는 평면 덩이에 해당하고 호출된 메소드의 수는 내포 덩이에 해당하므로 7±2 덩이 개념과 3±1 덩이 개념을 적용하면 10에서 36의 값을 가질 수 있다. 따라서 NIM 척도는 중앙값인 21을 기준으로 퍼지집합을 정의하였다.

클래스 내 메소드의 사이클로매틱 복잡도 합을 계산하는 WMC 척도는 평면 덩이와 제어 논리에 관계가 있다. 제어 논리 또한 3±1로 한계된다고 하였기 때문에 NIM 척도와 마찬가지로 21을 기준으로 퍼지 집합을 정의하였다.

각각의 척도에서 이 값들을 고려하여 $\overline{x_i}'$와 δ_i' 값으로 주고, 프로그램에서 측정한 척도 값들을 분할하여 퍼지화하였다.

5. 평가

전체 시스템의 복잡도를 계산하기 위해서는 추출된 5개의 직교적인 인자에 대해 학습을 수행하여중

<표 4> 회귀식과 퍼지 모델 결과 값

<Table 4> Result value of regression equation and fuzzy system

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
1	312	47	32	196	196	1256.317993	2545.794189
2	252	37	13	164	102	994.856995	2068.736328
3	235	46	17	131	122	866.554016	2050.911621
4	349	15	18	125	137	1160.911011	1928.970459
5	230	33	18	75	95	767.999023	1608.666870
6	182	40	11	36	118	702.629028	1149.566162
7	121	25	23	106	61	510.677002	1074.481445
8	134	13	19	106	66	537.901978	957.671936
9	112	18	15	66	60	449.826996	714.131592
10	115	23	16	48	59	431.527008	711.057678
11	166	20	11	27	41	472.514008	640.224854
12	73	27	16	84	47	377.036987	631.487732
13	99	12	16	76	48	380.272003	568.917419
14	88	18	11	55	48	338.475006	482.630737
15	70	14	14	62	46	323.191986	413.564850
16	91	20	7	27	38	329.681000	349.380402
17	61	25	5	45	41	287.704987	327.242371
18	57	18	8	14	32	231.445999	199.320511
19	43	10	11	27	29	192.485001	160.558716
20	33	6	18	40	16	179.634003	139.787155
21	24	12	13	17	17	124.191002	96.194283
22	21	13	7	19	16	117.156998	77.989021
23	17	13	7	10	14	98.796997	59.386250
24	24	13	4	6	14	104.867996	58.634781
25	17	11	7	10	14	97.147003	55.033142
26	23	11	3	5	13	99.892998	48.781731
27	8	5	9	6	7	52.186001	27.759098
28	17	3	5	2	7	61.629002	25.877260
29	5	4	3	2	4	30.660999	11.183234
30	9	1	3	0	1	35.887001	10.072245

<표 7> 회귀식과 퍼지 시스템의 결과 2

<Table 7> Result 2 of regression equation and fuzzy model

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
25	17	11	7	10	14	97.147003	55.033142
26	23	11	3	5	13	99.892998	48.781731

치를 얻어내어야 한다. 그러나 그 인자 중 프로그램 볼륨 인자에는 5개의 척도가 포함되어 있으므로 이 척도들로부터 인자의 값을 추출하여야 전체 복잡도를 얻기 위한 인자들의 학습과정에 반영할 수 있다. 본 논문에서는 프로그램 볼륨 인자 값을 얻기 위해 정의된 퍼지 모델에 대해서 220개의 학습 데이터를 가지고 100번의 학습을 수행하였다. 오차는 다음 식 (12)에 의해 계산되었는데 수행 결과 0.004의 오차를 보였다.

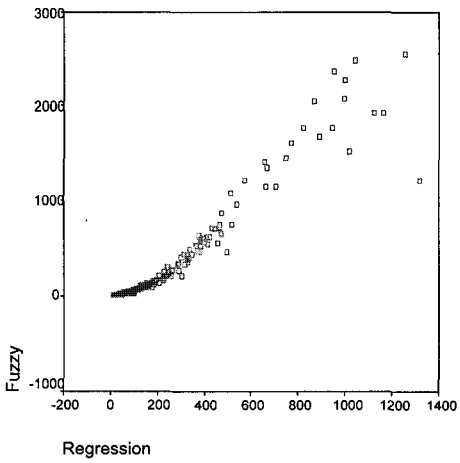
$$E = \frac{1}{n}\sqrt{\left(\sum_{i=1}^{n}(y-\hat{y})^2\right)} \qquad (12)$$

<표 5> 실행 결과 기술 통계

<Table 5> Description statistics of result

	Mean	Max	Min
Regression	230.0564	1317.311035	7.789000
Fuzzy	310.3212	2545.794189	1.200639

이 결과를 본 논문에서 제시된 퍼지 모델의 유용성을 평가하기 위해 397개의 프로그램에 적용하여 다중 회귀 분석에서 나온 결과와 비교한 내용을 <표 4>와 <표 5>에서 보여주고 있다. 회귀식이 단순히 선형적인 결과를 가져오는데 반하여, 퍼지 모델에서는 인지 심리 이론의 반영으로 보다 넓은 분포를 가지는 것을 알 수 있다. 하지만 [그림 3]에서 보는 것처럼 두 모델의 결과 값에 대한 pearson의 상관계수(correlation coefficient)는 0.959로 상당히 밀접한 관계가 있는 것으로 나타났다. 또한 <표 4>에서 보는 것처럼 두 시스템의 결과가 서로 상반되게 나오는 경우도 많았다. 이것은 각 척도 값의 크기만 반영되어 결정되는 회귀식의 결과와는 달리 퍼지 모델에서는 각 척도마다 다르게 가지는 인지 심리 이론의 덩이(chunk) 개념이 반영되어 다르게 나타난 결과이다. 각각의 프로그램에 대해 퍼지 시스템의 결과를 기준으로 회귀식의 결과와 퍼지시스템의 결과가 다른 경우를 살펴보았다.



[그림 3] 회귀식과 퍼지 모델의 상관관계

[Fig. 3] Correlation between regression equation and fuzzy model

<표 6> 회귀식과 퍼지 모델의 결과 1

<Table 6> Result 1 of regression equation and fuzzy model

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
10	115	23	16	48	59	431.527008	711.057678
11	166	20	11	27	41	472.514008	640.224854

① SLOC 척도 값을 제외한 나머지 척도 값이 모두 큰 경우

SLOC 척도는 인지 심리 이론을 적용했을 때 50 줄이 단기 기억의 한계이므로, 프로그램 11의 SLOC 척도 값이 프로그램 10의 SLOC 척도 값보다 한 단계 위의 퍼지 집합에 속해 높은 값을 가지게 되지만 프로그램 10의 다른 척도 값들이 모두 프로그램 11의 척도 값들보다 한 단계 이상의 퍼지 집합에 속하기 때문에 회귀식의 결과와는 달리 퍼지 시스템에서는 프로그램 10이 훨씬 높은 복잡도 값을 가지게 된다.

② SLOC 척도 값은 작고 NOPM 척도 값은 같지만 NOCB, WMC, NIM 척도의 값이 큰 경우

③ SLOC 척도 값은 작고 WMC 척도 값은 같지만 NOPM, NOCB, NIM 척도 값이 큰 경우

SLOC 척도 값이 단기 기억의 한계로 설정하였던 50보다 작을 경우에는 결과 값에 큰 영향을 주지 못한다. 또한 프로그램 25와 26, 27과 28의 SLOC 척도를 제외한 다른 척도 값들이 같은 퍼지 집합에 속하지만 프로그램 25와 27의 척도 값들이 높은 값을 가지고 있으므로 프로그램 25와 27이 높은 복잡도 값을 가지게 된다.

④ SLOC 척도 값과 NOPM 척도 값이 작지만 NOCB, NIM, WMC 척도 값이 큰 경우

<표 8> 회귀식과 퍼지 시스템의 결과 3

<Table 8> Result 3 of regression equation and fuzzy model

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
27	8	5	9	6	7	52.186001	27.759098
28	17	3	5	2	7	61.629002	25.877260

<표 9> 회귀식과 퍼지 시스템의 결과 4

<Table 9> Result 4 of regression equation and fuzzy model

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
15	70	14	14	62	46	323.191986	413.564850
16	91	20	7	27	38	329.681000	349.380402

<표 10> 회귀식과 퍼지 시스템의 결과 5

<Table 10> Result 5 of regression equation and fuzzy model

Prog. No.	Metrics					Result	
	SLOC	NOPM	NOCB	NIM	WMC	Regression	Fuzzy
3	235	46	17	131	122	866.554016	2050.911621
4	349	15	18	125	137	1160.911011	1928.970459

프로그램 15는 회귀식에서 가장 큰 비중을 차지하는 SLOC 척도와 NOPM 척도가 프로그램 16보다 작지만 프로그램 15의 다른 척도 값들이 모두 프로그램 16의 척도 값들보다 한 단계 이상의 퍼지 집합에 속하기 때문에 회귀식의 결과와는 달리 퍼지 시스템에서는 프로그램 15가 훨씬 높은 복잡도 값을 가지게 된다.

⑤ SLOC, NOCB, WMC 척도 값이 작지만 NOPM, NIM 척도 값이 큰 경우

회귀식에서는 3개의 척도 값이 크기 때문에 프로그램 4의 복잡도가 크게 나타났지만, 퍼지 시스템에서 NOPM 척도는 단기 기억의 한계수로 7을 사용하여 퍼지 집합을 분할하였기

때문에 두 프로그램사이에는 네 단계 이상의 차이가 있다. 이 차이가 프로그램 4의 SLOC, NOCB, WMC 척도 값들이 복잡도에 미치는 영향보다 크기 때문에 프로그램 3의 복잡도가 크게 나타났다.

이상의 결과들이 인지 심리 복잡도를 잘 반영하였는지를 알아보기 위하여 컴퓨터과학과 박사과정에 재학중인 대학원생들을 통해 비교 분석한 결과가 <표 11>에 나타나있다. 인지 심리 이론을 적용하고 퍼지 학습을 수행하여 복잡도를 측정한 결과가 회귀식에 의해 얻어진 결과보다 사용자에게 훨씬 높은 만족도를 얻는 것을 볼 수 있다. 분석자가 비교하였을 때 어느 것이 복잡하다고 평가하기 곤란한 경우의 데이터들은 제외하였다.

<표 11> 두 식에 대한 결과 만족도 평가

<Table 11> Evaluation of degree of satisfaction about two equation

	Mean	Max	Min
Regression	59.5%	68.6%	54.1%
Fuzzy	86.2%	94.1%	77.7%

퍼지 학습에 사용된 학습 데이터 중에 사이즈가 큰 프로그램들이 100 SLOC 이하의 프로그램보다 적었던 관계로 학습이 미흡하여 <표 12>에 보면 사이즈가 큰 프로그램에 대해 사용자의 복잡도 측정 만족도가 약간 떨어지는 것을 알 수 있다.

<표 12> 퍼지 시스템에 대한 크기별 만족도 분석결과

<Table 12> Result of degree of satisfaction per size about fuzzy system

	Mean	Max	Min
> 100 SLOC	82.5%	92.5%	75%
< 100 SLOC	87%	94.4%	78.3%

6. 결론

본 논문에서는 퍼지 적응 제어 방법을 이용하여 Java 프로그램의 볼륨 복잡도를 측정할 수 있는 모델을 개발하였다. 우선 널리 사용되고 있는 12개의 척도들에 대해 통계적 분석을 수행하여 프로그램의 특징인 인자들을 추출하였다. 서로 직교적인 인자들을 추출함으로써 퍼지 규칙의 수를 감소할 수 있고, 훈련에 사용되는 데이터가 되도록 직교적이어야 한다는 조건을 만족할 수 있다. 추출한 인자들에 대해 단계적 변수투입법(stepwise method)으로 회귀 분석을 수행하여 다중공선성이 있는 척도들을 제거한 후, 인자들에서 가장 큰 설명력을 가지는 프로그램 볼륨에 대해 인지 심리 이론을 반영한 퍼지 학습을 수행하여 프로그램의 볼륨 복잡도를 측정하는 모델을 개발하였다. 퍼지 방법을 이용한 모델은 다중 회귀 분석에서 밝혀낼 수 없었던 입력 데이터의 비선형성을 인식할 수 있었고, 인지 심리 이론을 적용할 수 있어서 회귀 분석에서보다 훨씬 나은 결과를 얻을 수 있었다.

향후 연구과제로는 크기가 큰 프로그램에 대한 추가 학습을 수행하여 보다 정확한 결과를 추출할 수 있도록 노력하고, 프로그램의 볼륨 복잡도 뿐만 아니라 전체 프로그램의 복잡도를 측정할 수 있는 퍼지 모델을 개발하는 것이 필요하다. 또한 퍼지 모델의 바탕이 되는 척도들에 대한 측정부터 전체 복잡도 값의 측정까지를 시각적으로 자동화해줄 수 있는 도구의 개발이 필요하다.

※ 참고문헌

[1] H. Zuse, *Software Complexity: Measures and Methods*, Walter de Gruyter & Co., 1991.

[2] G. A. Miller, "The Magical Number Seven Plus or Minus Two : Some Limits on Out Capacity to Process Information," *Psychological Reviews*, Vol. 63, pp. 81-87, 1956.

[3] Franck XIA, "Module Coupling: A Design Metric," *APSEC '96*, Dec. 1996.

[4] Gugerty L. and olson G., "Comprehension differences in debugging by skilled and novice programmers," *Proceedings of the First Workshop on Empirical Studies of Programmers*, 1986.

[5] Littman D. C., Pinto J., Letovsky S. and Soloway E., "Mental Models and Software Maintenance," in *Proceedings of the Second Workshop on Empirical Studies of Programmers*, 1987.

6] Briand L., Morasca S., and Basili V., "Property Based Software Engineering Measurement," *IEEE Trans. on Software Engineering*, Vol. 22, No. 1, pp. 68-86, Jan. 1996.

[7] Samuel B. Green, Neil Salkind, Teresa Akey, and Neil S. Salkind, *Using SPSS for Windows: Analyzing and Understanding Data*, Upper Saddle River, NJ: Prentice Hall, 1997.

[8] L. X. Wang, *Adaptive fuzzy systems and control.* Prentice-Hall International, Inc., 1994.

[9] James G. Greeno, "The Structure of Memory and the Process of Solving Problems," *in Contemporary Issues in Cognitive Psychology*, pp. 103-133. 1973.

[10] R. Mayer and B. Shneiderman, Syntactic/Semantic Interactions in Programmer Behavior : Model and Experimental Results, *International Journal of Computer and Information Sciences*, Vol. 8, pp. 213-238, 1979.

[11] Letovsky, S., "Cognitive processes in program comprehension," *Proceedings of the First Workshop on Empirical Studies of Programmers*, 1986.

[12] Letovsky, S., and Soloway, E., "Strategies for Documenting Delocalized Plans," *IEEE Software*, pp. 41-49, May 1986.

[13] L. Weissman, "Psychological Complexity of Computer Program : An Experimental Methodology," *ACM SIGPLAN Notices*, Vol. 9, pp. 25-36, 1974.

[14] Lakshmanian K. B., Jayaprakash S., and Sinha P. K., "Properties of Control-Flow Complexity Measures," *IEEE Trans. on Software Engineering*, Vol. 17, No. 12, pp. 1289-1295, Dec. 1991.

[15] Tian J. and Zelkowitz M. V., "A Formal Program Complexity Model and Its Application," *Journal of Systems and Software*, Vol. 17, pp. 253-266, 1992.

[16] Weyuker E. J., "Evaluating Software Complexity Measures," *IEEE Trans. on Software Engineering*, Vol. 14, No. 9, pp. 1357-1365, Sept. 1988.

[17] McCabe, T. J., "A Complexity Measure," *IEEE Trans. on Software Engineering*, Vol. 2, No. 4, pp. 308-320, April 1976.

[18] Chidamber S. R. and Kemerer C. F., "A Metrics Suite for Object-Oriented Design," IEEE Trans. on Software Engineering, Vol. 20, No. 6, pp. 476-493, Jun. 1994.

[19] Briand L., Daly J., and Wust J., "A Unified Framework for Cohesion Measurement in Object- Oriented Systems," *Empirical Software Engineering Journal*, 3(1), pp. 65-117, 1998.

[20] Brian W. Bush, BANDA *Java Packages*, http://www.sladen.com/Java.

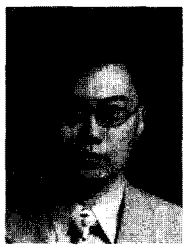
[21] Andrew Cain, Rajesh Vasa, *Java Metric Analyser*, http://www.csse.swin.edu.au/cotar/jmetric/ index.html.

[22] 유대근, 권영식, 통계분석을 위한 SPSSWIN 8.0, 기한재, 1999.

[23] 김재웅, 유철중, 장옥배, "Java 프로그램에 대한 복잡도 척도들의 실험적 검증", *정보과학회논문지 : 소프트웨어 및 응용*, 제 27 권, 제 12 호, pp. 1141-1154, 2000.

[24] 이광형, 오길록, *퍼지 이론 및 응용 II권:응용*, *홍릉과학출판사*, 1991.

김재웅



1993년 2월 전북대학교
전자계산학과 졸업(이학사)
1995년 2월 전북대학교
전자통계학과 졸업(이학석사)
1995년 3월～현재
전북대학교 전산통계학과
박사과정
1995년 3월～1996년 3월
군산대학교 계산통계학과 조교
2000년 10월～현재
(주)케이테크 선임연구원
관심분야 : 소프트웨어공학,
소프트웨어 척도,
컴포넌트 복잡도, 인지과학