IE Interfaces
Vol. 12, No. 4, pp. 540-550, December 1999.

단순진화 알고리듬을 위한 애플리케이션 프레임워크 개발†

이수연[1] · 정호연[2] · 서광언[1] · 김여근[1]

[1]전남대학교 산업공학과/ [2]전주대학교 산업공학과

Development of an Application Framework for Simple Evolutionary Algorithms

Soo-Yeon Lee[1] · Ho-Yeon Chung[2] · Kwang-Un Seo[1] · Yeo-Keun Kim[1]

In evolutionary algorithm, there exist various models for the evolution of the population with respect to schemes and strategies for reproduction. In the application of the algorithm to a specific problem, one model suitable to the problem is to be properly chosen and a program expert or a software is needed to help implement and test a designed algorithm. In this study, the software for simple evolutionary algorithms(SEA) with one population is developed. The software is designed as an application framework type, so that it may be friendly, allow users to add some program, and operate under the environment of Windows. For this, hierarchical classes for components of SEA are first designed by means of an object-oriented approach and then a library for SEA is built by them. With the library, developed is an application framework that can generate a frame code for an application program. The software proposed here can be used as a generalized tool for solving problems in a wide range of domains.

1. 서 론

진화알고리듬은 일종의 인공지능기법으로 유전알고리듬(genetic algorithm), 진화전략(evolution strategy), 진화프로그래밍(evolutionary programming) 등이 이에 속한다. 진화알고리듬은 하나의 種(species)이 고정된 환경에 적응하여 진화하는 모형과 여러 종들이 동시에 상호작용하고 상호적응하면서 공진화(coevolution)하는 모형으로 크게 분류할 수 있다. 본 연구에서는 전자를 단순진화 알고리듬이라 부르기로 하고, 이에 관해서 다루기로 한다.

특정 문제에 대해 진화알고리듬을 적용하기 위해서는 모집단의 운영방법이나 진화방식에 대한 선별방법, 유전연산자 등 다양한 요소별 방법과 전략 가운데 적용문제에 적당한 방법을 선택하거나 개발하여, 이를 구현해야 한다. 그러나 이러한 과정은 상당히 복잡하여 진화알고리듬과 프로그래밍에 익숙하지 않은 의사결정자가 진화알고리듬을 의사결정의 기본도구로서 사용하기에는 많은 어려움이 따른다.

따라서 진화알고리듬을 보다 쉽게 활용하기 위해서는 누구나 쉽게 사용할 수 있는 진화알고리듬을 위한 소프트웨어의 개발이 필요하다. 이제까지 개발된 진화알고리듬에 관한 소프트웨어로 EVO(TransferTech GmbH), Generator(New Light Industries,1995), FlexTools(Flexible Intelligence Group, 1996), Genetic Algorithm Toolbox for Matlab(Automatic Control & Systems engineering), Amber Alife Wasps(Roman V, 1998), BUGS(Smith, J. R., 1992), GAC(William, M. S., 1994), GAL(William, M. S., 1992), libga100(Art, C., 1993), Matthew's GAlib(Matthew, W., 1995), GECO(George, P. W. and Williams, Jr.,(1993), GENEsYs(Zbigniew, M., 1992), GPC^{++}(Adam, F. and Thomas, W.), Paragenesis(Mike, V. L., 1992) 등이 개발되어 왔다. 이들 소프트웨어는 크게 세 가지 형태로 분류된다. 첫째는 어느 특정분야만을 위해 개발된 소프트웨어(Mike, V. L., 1992; Roman, V., 1998; Smith, J. R., 1992; TransferTech GmbH; William, M. S, 1994 William, M. S, 1992; Zbigniew, M., 1992)이고, 둘째는 진화알고리듬의 각 요소를 위한 라이브러리(Art, C., 1993; Matthew, W., 1995; Wall, M.,(1996)

† 본 연구는 한국과학재단 특정기초연구(과제번호: 98-0200-09-01-3) 지원으로 수행되었음.

이다. 그리고 셋째는 진화알고리듬 요소들 중에서 사용자가 원하는 방법을 적용하여 여러 분야에 적용할 수 있도록 해주는 소프트웨어(Automatic Control & Systems engineering; Flexible Intelligence Group, 1996; New Light Industries,1995)이다.

개발된 소프트웨어 중에서 진화알고리듬 요소들을 사용자가 원하는 방법을 사용하고, 적용 분야에 따른 특수한 부분을 추가할 수 있도록 하는 소프트웨어는 몇 가지가 개발되어 왔으나, 이들은 모두 전체적인 응용프로그램의 소스코드를 생성해 주는 것이 아닌 Microsoft사의 EXCEL이나, MATLAB과 같은 플랫폼(platform)하에서 사용자의 적용문제에 특수한 부분의 추가만을 할 수 있도록 제공(Automatic Control & Systems engineering; Flexible Intelligence Group, 1996; New Light Industries,1995)한다. 따라서 진화알고리듬의 각 모든 요소에 대해 사용자가 자신의 의도를 반영시킬 수 있는 유연성을 제공하지 못한다는 단점을 갖는다. 한편 응용 프로그램 개발에 이용할 수 있는 라이브러리는 알고리듬 구성요소에 대한 솔루션은 제공하고 있으나 응용프로그램의 전체적인 구조는 제공하지 못한다는 단점을 갖는다. 따라서 기본적인 진화알고리듬에 대한 강건하고 간결한 라이브러리의 지원과 함께 응용프로그램의 전체적인 골격코드를 생성해 주는 소프트웨어의 개발이 필요하다.

여러 분야에 대한 진화알고리듬 응용프로그램은 그 정적구조와 흐름이 서로 유사하다고 할 수 있다. 따라서 진화알고리듬의 구성요소를 파라메타화 하여 이를 구조화 시킴으로써 사용자가 원하는 방법을 사용한 응용프로그램을 생산해 주는 자동화된 소프트웨어 생산시스템을 개발할 수 있다. 본 연구에서는 단순진화 알고리듬을 위한 소프트웨어를 애플리케이션 프레임워크(application framework)의 형태로 개발하고자 한다. 여기서 애플리케이션 프레임워크란 사용자가 개발하고자 하는 응용프로그램의 흐름과 전체적인 골격코드를 생성해 주고, 사용자는 자신의 응용프로그램에 고유한 코드만을 추가함으로써 하나의 완전한 응용프로그램을 생성하는 접근방법을 말한다(Tailgent Inc, 1993).

본 연구에서 개발하는 단순진화 알고리듬을 위한 애플리케이션 프레임워크는 크게 세 부분으로 구성된다. 첫째는 진화알고리듬의 구성요소를 위한 라이브러리이고, 둘째는 사용자의 의도를 수용하기 위한 인터페이스, 그리고 셋째는 사용자가 선택한 요소별 방법을 적용한 응용프로그램의 골격코드를 생성하는 메인프레임이다. 본 연구에서는 특히 진화알고리듬에 대한 라이브러리 개발에 있어서 코드의 재사용성과 확장성을 위해 객체지향기법을 적용한다. 이를 위하여 단순진화 알고리듬의 구성 요소들에 대한 객체지향적 접근을 통해 계층적 클래스를 설계하고, 이를 클래스 라이브러리로 구현한다. 그리고 구축된 라이브러리를 이용, 이들을 구조화 시킴으로써 사용자가 작성하고자 하는 응용 프로그램의 골격코드를 생성해 주는 프레임워크를 개발한다.

본 연구의 구성은 다음과 같다. 제2장에서는 단순진화 알고리듬 관련 요소를 기술하고, 제3장에서는 진화알고리듬에 대한 객체지향적 접근법을 통해 클래스 라이브러리의 설계 및 구현방법을 제시한다. 제4장에서는 개발된 클래스 라이브러리를 이용한 애플리케이션 프레임워크의 설계 및 개발 방법론을 설명하고, 제5장에서 적용문제를 통해 개발한 애플리케이션 프레임워크의 유용성을 보인다. 제6장은 결론으로 구성되어 있다.

2. 진화알고리듬

2.1 진화알고리듬

1960년대에 여러 컴퓨터 과학자들은 진화가 공학 문제를 위한 최적화 도구로 사용될 수 있다는 개념으로 독립적으로 진화시스템에 관해 연구되었다. Renchenberg(Michalewicz, 1994)는 파라메터 최적화 문제를 해결하기 위하여 자연 진화의 원리를 모방한 알고리듬을 개발하였다. 이를 진화전략이라 부른다. Fogel, Owens, Walse(Fogel, L. J., Owens, A. J. and Walsh, M. J., 1966)는 환경의 변화를 예측할 수 있는 인공지능 기법으로 진화프로그래밍을 제안하였다. Holland(Holland, J. H., 1995)는 유전알고리듬을 개발하였다. 그는 자연의 적응현상을 모방하여 컴퓨터를 이용한 인공적응 시스템을 개발하는데 목적을 두었다. 이러한 자연의 적응과정을 모방한 유전알고리듬은 다양한 유형의 문제를 해결하는데 훌륭한 도구로 사용되고 있다. 이들 기법들간에는 해의 표현, 유전연산, 재생산방법 등에서 약간의 차이가 있으나 기본적으로 자연선택과 유전법칙을 모방한 일종의 확률적 탐색기법이다. 각 기법들은 초기에는 구성요소에서 상당한 차이가 있었으나 현재는 크게 구분하지 않고 각 기법이 갖는 단점들을 상호 보완하면서 통합된 하나의 알고리듬으로 사용하는 경향이 있다.

진화알고리듬의 가장 큰 특징은 뉴톤법과 같은 고전적 최적화기법이나 타부서치(tabu search), 시뮬레이티드 어닐링(simulated annealing)과 같은 이웃해 탐색기법이 하나의 해를 운용하는 데 반하여, 진화알고리듬은 복수개의 잠재해들로 이루어진 해의 집단(population)을 운용한다는 것이다. 이러한 해집단에 자연선별(natural selection)과 유전법칙의 메카니즘을 적용하여 세대를 진행시키면서 해공간을 탐색해 간다.

2.2 진화알고리듬의 구성요소

진화알고리듬은 문제의 잠재해를 표현한 개체들로 이루어진 모집단을 가지고 시작한다. 모집단은 매 세대마다 일정수의 개체를 유지하고 매 세대에서 각 개체의 적응도(fitness)를 평가하여 이에 따라 다음 세대에 생존할 개체들을 확률적 또는 확정적으로 선별한다. 선별된 개체들 중 일부의 개체들이 임의로 짝을 지어 교배하여 자손을 생성한다. 이때 교차(crossover)

```
begin
  t ← 0
  P(t)의 초기화(초기모집단 생성)
  P(t)의 적응도 평가
 while(종료조건이 만족되지 않으면) do
  begin
    t ← t+1
    P(t-1)로 부터 P(t)를 선별
    P(t)의 유전연산(교차와 돌연변이)
    P(t)의 적응도 평가
  end
end
```

그림 1. 진화알고리듬의 구조.

에 의해 부모의 유전자가 자손에게 상속되고 돌연변이가 일어날 수 있다. 자손은 부모로부터 좋은 유전형질을 상속 받는다고 가정함으로써 다음 세대의 잠재해들은 평균적으로 전 세대보다 더 좋아진다고 본다. 이러한 진화과정은 종료조건을 만족할 때까지 반복한다. 이 과정을 정리하면 <그림 1>과 같다(Michalewicz, 1994). 여기서 $P(t)$는 세대 t에서의 모집단을 나타낸다.

진화알고리듬의 모든 요소별 전략이나 방법을 모두 포함할 수 있는 소프트웨어를 개발하기 위하여 본 연구에서는 1) 알고리듬, 2) 개체 표현과 유전연산자, 3) 선별방법, 4) 기타 요소로 기존 이론을 파악하였다. 앞서 언급하였듯이 본 연구에서는 다양한 문제에 진화알고리듬을 적용하고자 할 때 응용 프로그램의 코드를 생성해 주는 소프트웨어를 개발하고자 한다. 따라서 개발하는 소프트웨어는 가능한 기존의 진화알고리듬에 대한 모든 방법이나 전략을 지원할 수 있도록 해야 한다. 본 연구에서는 각 요소의 대표적인 기법 또는 전략들을 고려한다. 그리고 개발된 소프트웨어는 다양한 기법이나 전략들을 쉽게 추가할 수 있는 구조를 갖도록 한다.

2.2.1. 알고리듬

이제까지 진화알고리듬에서 모집단이 진화하는 과정은 여러 형태로 모형화 되었다. 본 연구에서는 크게 다음과 같이 분류되었다.

- 표준 유전알고리듬(standard genetic algorithm)
- 안정상태 유전알고리듬(steady-state genetic algorithm)
- 수정 유전알고리듬(modified genetic algorithm)
- 진화전략(evolution strategy)
- 진화프로그래밍(evolutionary programming)

이들 각 알고리듬을 구현하는 데 특징적으로 필요한 전략이나 방법들이 있다. 이들 가운데 대체전략(replacement strategy)으로 적응도가 낮은 개체를 확정적 또는 확률적으로 선택하여 대체하는 방법, 생성된 자손과 인자형이 유사한 개체를 선택하여 대체하는 방법, 임의로 선택하여 대체하는 방법 등을 포함하고 있다.

2.2.2. 개체 표현과 유전연산자

진화알고리듬을 적용하는데 첫 단계는 다루는 문제의 잠재해의 표현, 즉 개체의 표현이다. 표현방법은 문제에 따라 다를 수 있으며, 다양한 방법이 제시되어 왔다. 또한 표현방법과 적용문제에 따라 여러 유전 연산자가 개발되었다. 본 연구에서는 인자들의 구성방법에 따라 분류하고, 특히 1차원 배열표현 가운데 대표적인 표현방법을 아래와 같이 분류하고, 각 방법에 따른 대표적인 유전연산자를 라이브러리 함수에 포함시킨다.

- 1차원 배열표현
  - ▷ 이진표현 - 일점(one point)교차, 이점(two point)교차, 균등(uniform)교차 등
  - ▷ 부동소수점 표현 - 이산재결합(discrete recombination)교차, 선형조합(linear combination)교차, 산술(arithmetical)교차, 기하학적(geometrical)교차, 균등(uniform)돌연변이, 불균등(non uniform)돌연변이, 가우스(gaussian)돌연변이 등
  - ▷ 순서표현 - 순서(order)교차, 순서기반(order based)교차, 위치기반(position based)교차, 균등순서기반(uniform order based)교차, 부분사상(partially march)교차, 순환(cycle)교차, 인접인자재결합(edge recombination)교차와 교환(reciprocal exchange), 삽입(inserrion), 역순(inversion), 전위(displacement) 등의 돌연변이 연산자
  - ▷ 그룹표현 - 일점교차, 인접인자(edge based)교차, 구조(structural)교차 등
- 2차원 배열표현 - 이점교차, 합집합(union)연산자, 교집합(intersection)연산자, 발견적 역순 연산자 등
- 이진나무 표현 - 부분나무 교환(swap sub-tree node), 임의변환(swap node), 순열(permutation) 연산자, 편집(editing) 연산자 등

2.2.3. 선별

선별(selection)은 적자생존의 자연법칙에 기초하여, 즉 환경에 대한 적응도에 의해 현 세대의 모집단으로부터 다음세대에 생존할 개체를 선택하는 과정이다. 선별방법으로는 확률바퀴(roulette wheel) 선별, 순위(ranking)선별, 토너먼트(tournament) 선별, 잔여 확률(stochastic remainder) 샘플링, 확률 균등(stochastic uniform) 샘플링, 확정적(deterministic) 샘플링 등을 포함한다.

2.2.4. 기타 요소

진화알고리듬에 대해 앞서 제시한 요소들 외에도 여러 요소가 있으며, 본 연구에서는 다음과 같은 요소에 대해 연구된 내용을 분류하여 라이브러리 함수에 포함시킨다.

- 적응도 함수 스케일링 방법 선형(linear) 스케일링, 지수법칙(power law) 스케일링, 시그마 절단(sigma truncation), 몫짓기

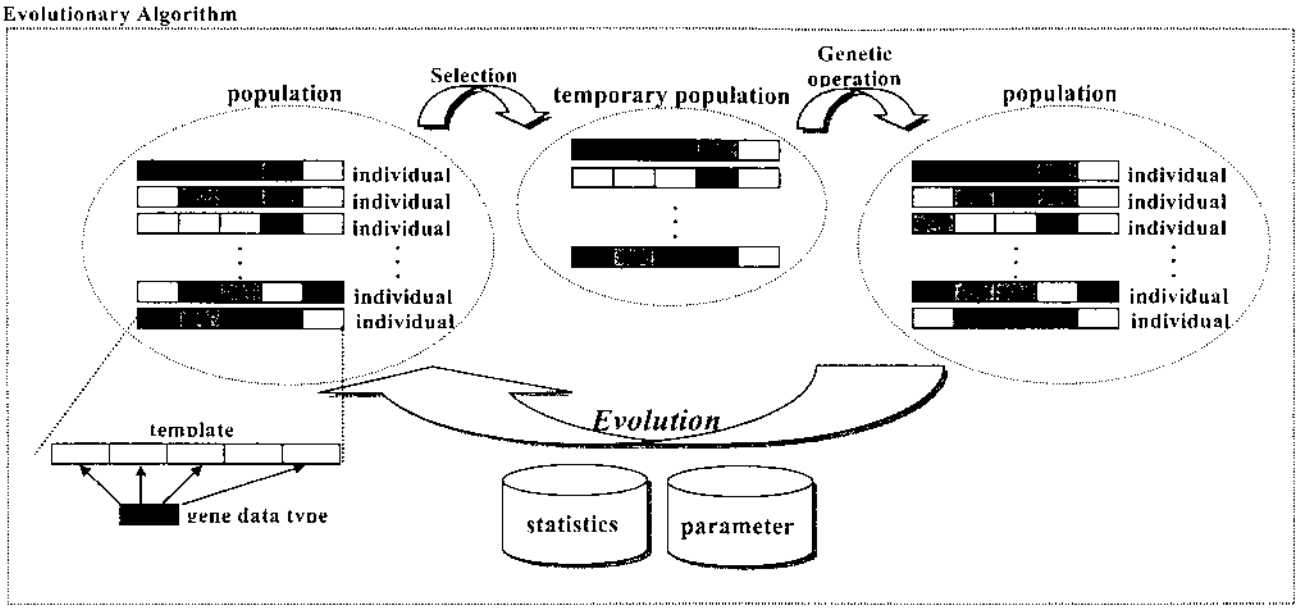


그림 2. 자료저장과 객체추출.

(sharing) 등

• 종료조건 - 진행된 세대 또는 생성된 개체의 수, 해의 개선이 이루어지지 않고 진행된 세대수 또는 생성된 개체수, 계산 소요시간 등

3. 진화알고리듬을 위한 클래스 라이브러리

3.1 객체지향

객체지향이란 소프트웨어를 자료구조와 행위가 결합된 객체들의 구조화된 집합으로 보는 시각을 말한다. 여기서 객체는 시스템을 구성하는 실체로서 각 객체는 한 구성원의 특성과 상태변화를 표현하는 기본단위라고 볼 수 있다. 이때 같은 자료구조와 행위를 갖는 객체들은 동일한 클래스로 분류된다. 상대적으로 각 객체는 소속 클래스의 인스턴스(instance)가 된다. 따라서 객체 지향 설계에 있어서 가장 핵심이 되는 부분은 시스템을 구성하는 객체의 추출과 객체의 속성을 정확히 묘사하는 클래스의 설계, 그리고 클래스간의 관계정의로 요약할 수 있다(Smith, J. R., 1992).

본 연구에서는 진화알고리듬을 위한 라이브러리를 객체지향적 접근을 통해 확장성이 높은 클래스 라이브러리로 개발한다. 객체지향 방법론은 이제까지 여러 방법론이 제시되었으나, 최근에는 객체지향 기술의 권위자인 Grady Booch, James Rumbaugh, Ivar Jacobson 등이 자신들의 주장(booch방법론, OMT, OOSE)과 기타 다른 전문가들의 주장을 통합하여 만든 UML (unified modeling language)이 널리 쓰이고 있다. 이때 UML은 객체지향 개발 방법론 자체가 아닌 구성요소의 하나로 객체지향적 설계를 하기 위한 도구이다. 따라서 본 연구에서는 UML을 사용하여 분석(analysis), 설계(Design), 구현(implementation)의 단계로 클래스 라이브러리를 개발하였다.

표 1. 진화알고리듬 요소들의 객체표현과 구현방법

구성요소	객체여부	구현방법
알고리듬	0	클래스
인자	×	클래스
개체	0	클래스
모집단	0	클래스
선별	0	클래스
스케일링	0	클래스
유전연산자	×	라이브러리 함수
통계량	0	클래스
유전파라메터	0	클래스

먼저, 진화알고리듬에 대해서 객체들을 도출하고, 이들간의 관계를 정의하고, 객체들의 정적구조와 객체간의 관계를 나타내는 객체 다이어그램(object diagram)을 작성하였다. 그리고 정의한 객체에 대한 구체적인 클래스를 설계하고, 클래스간의 관계를 정의하였다. 이때 UML의 비주얼 모델링 도구인 Rational Rose 98을 사용하여 안정적인 설계를 도모하였다. 마지막으로 객체지향 프로그래밍 언어인 C++를 사용하여 이들을 클래스 라이브러리로 구현하였다.

3.2 객체지향적 분석

객체는 구축하려는 시스템을 구성하는 개념적 단위로, 자기 자신의 고유한 성질을 표현하기 위한 데이터구조(멤버데이터)와 데이터를 처리하는 기능(멤버함수)을 함께 갖는 정보단위이다. 따라서 객체를 정의하는 것은 결국 해당 시스템의 자료

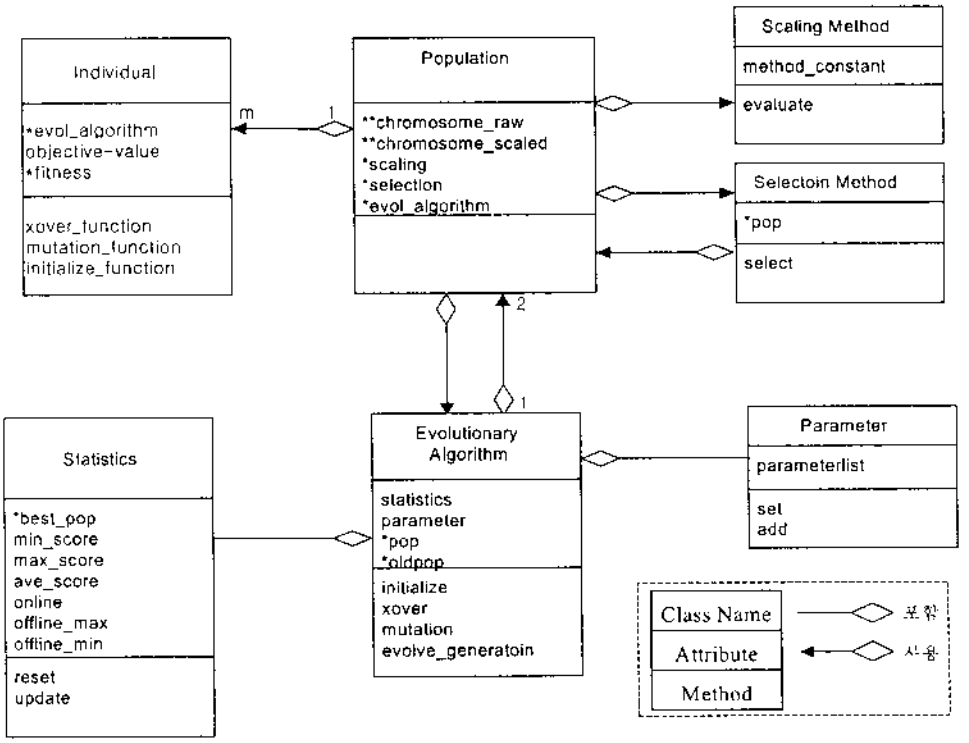


그림 3. 객체 다이어그램.

구조를 결정하는 것이 된다. 본 절에서는 진화알고리듬을 위한 라이브러리를 구축하기 위해 필요한 객체를 도출하고, 이들 객체들간의 관계를 정의한다.

<그림 2>는 진화알고리듬을 구성하는 객체를 추출하기 위해서 프로그래밍 관점에서 자료의 저장과 자료의 변환과정을 나타낸 것이다. 진화알고리듬의 가장 기본적인 객체는 개체 객체와 개체들의 진화를 위한 알고리듬 객체이다. 여기서 개체를 이루는 각 인자들은 독립적인 주체가 아니기 때문에 하나의 객체로서는 불완전하다고 볼 수 있다. 모집단은 개체 객체들의 집합으로, 개체들의 분포형태(예로, 1차원 배열구조 또는 2차원 격자상에 분포 등과 같은)와 각 세대에서 적응도가 가장 좋은 개체의 보관 등을 위해 객체로 추출하였다. 한편 선별과정은 여러 방법이 개발되어 왔으나 기본적으로 각 개체의 적응도에 기초하여 진행된다. 객체지향 개념에서는 유사한 클래스간에 공통된 속성과 방법 등을 모아 하나의 클래스로 만들고, 유사한 클래스들은 이 클래스로부터 공통된 속성과 멤버함수에 대한 정의를 상속받는다. 선별과정은 모집단 객체에서 멤버함수로 정의될 수 있으나, 여러 스케일링 방법과의 조합을 위해 독립적인 객체로 추출하였다. 이때 선별에 관련된 객체는 개체 객체와 모집단 객체에서 사용관계로 둠으로써, 다음 세대를 구성하기 위한 개체의 선별과 짝짓기(mating)를 위한 선별에 사용될 수 있도록 하였다. 개체들은 교차와 돌연변이의 유전연산을 통하여 새로운 개체를 생산한다. 이와 같은 유전연산은 객체들간의 메시지 형태로 하나의 객체로서 인식하기 보다는 객체의 멤버함수로서 간주하였다. 그러나 이제까지 개발된 모든 유전연산자를 멤버함수로 정의한다는 것은 확장성 측면에서 많은 문제점을 갖게 된다. 따라서 유전연산자는 템플릿(template) 함수 형태로 모집단 객체의 멤버함수로 정의함으로써 사용자가 자신의 유전연산자를 쉽게 구현하도록 하였다. 그리고 기존에 개발된 대표적인 유전연산자를 라이브러리 함수로 구현하였다. 진화알고리듬에 사용되는 파라메타와 통계량으로 많은 종류가 있다. 이들은 모두 알고리듬 객체에 포함될 수 있으나, 독립적인 객체로 추출하였다. 결국 진화알고리듬을 이루는 여러 요소들에 대해 본 연구에서 추출한 객체와 구현방법을 정리하면 <표 1>과 같다.

객체지향 분석단계에서는 추출한 객체들간의 관계를 정의함으로써 시스템의 정적구조를 나타낸다. UML에서는 객체들 사이에 일반(association), 상속(generalization), 집합(aggregation), 종속(dependency)의 4가지 객체간 관계가 존재한다(Terry, Q., 1998). 일반관계는 사용한다(use)의 의미로 해석될 수 있는데, 대개 설계 초기에 클래스 간의 관계는 일반으로 표현되며, 개발이 진행됨에 따라 상속, 집합, 종속으로 구체화 된다. 상속관계는 말 그대로 상속관계를 나타낸다. 즉, 상속받은 개체는 상위 개체의 모든 멤버 데이터와 멤버함수를 상속받게 된다. 집합관계는 객체간의 포함관계를 의미하는데, 예를 들어 자동차 객체의 경우, 엔진이라는 객체를 포함한다. 종속관계는 일반관계 중에서 상속, 집합이 아닌 관계에 해당한다. 구현관점에서 볼 때, 멤버함수 안에서 상대 클래스의 객체를 생성하여 사용하는 경우, 멤버함수의 인자로 상대 클래스의 객체가 넘어오는 경우 등이 이에 해당한다(Terry, Q., 1998).

표 2. 상위클래스의 주요 멤버 데이터와 멤버 함수

클래스이름	멤버데이터/멤버함수		해설
Evolutionary_ Algorithm	Attributes:	Parameter	현재의 교차율, 돌연변이율 등의 파라메터
		elitist flag	Elitism 사용여부
		Minmax	최대화 최소화 여부
	Operations:	evolve_generation()	한세대의 진화를 수행
		maximize()/minimize()	최대화/최소화 문제임을 표시
Chromosome	Attributes:	objective_value	목적함수의 값
		*fitness	스케일링 된 적응도
	Operations:	XoverFunction()	교차 함수
		MutaionFuntion	돌연변이 함수
Population	Attributes:	raw_min,max,ave,sum	원 목적함수 값의 최소, 최대, 평균, 합
		scaled_min,max,ave,sum	적응도 값의 최소, 최대, 평균, 합
	Operations:	evaluate()	목적함수의 평가
		scaling()	스케일링
		selection()	선별클래스의 선별함수 호출
ScalingMethod	Attributes:	scale_flag	스케일링 flag
		Multiplier	스케일링 상수
	Operations:	evaluate()	적응도 평가
		copy()	멤버 변수의 복사
SelectionMethod	Attributes:	select_flag	선별 flag
	Operations:	select()	선별을 실행
		copy()	개체의 복사
		assign()	메모리 할당
Statistics	Attributes:	online	on-line performance
		offline_max,min	Max/Min offline performance
	Operations:	reset()	통계치의 초기화
		update()	진화 후 통계치의 갱신
Parameter	Attributes:	parameterlist	파라메터 리스트
	Operations:	add()	파라메터의 추가
		set()	파라메터의 설정

본 연구에서는 통계량 객체와 파라메터 객체를 알고리듬 객체에, 개체 객체를 모집단 객체에 속하는 집합관계로 설정하고, 모집단 객체를 스케일링 객체와 선별 객체의 종속관계로 설정하는 등 객체들간 관계를 설정하였다. <그림 3>은 본 연구에서 설정한 객체와 객체들간의 관계를 나타내는 객체 다이어그램이다.

3.3 클래스 라이브러리의 설계 및 구현

클래스 라이브러리는 높은 생산성과 유지보수의 편리함, 신뢰성 보장 등의 요건을 만족시켜야 한다. 본 연구에서 개발하는 클래스 라이브러리는 분석단계에서 추출한 객체에 대해 클래스로 구체화하였으며, 이때 진화알고리듬의 모든 요소에 대한 추후 연구되는 방법을 쉽게 추가할 수 있도록 하기 위한 클래스의 정의와 이들간의 관계 설정에 초점을 두었다.

먼저, 개체와 진화과정, 선별에 관련된 클래스의 경우 각각 다양한 방법과 전략을 갖기 때문에 공통적인 속성과 기능을 추출하여 상위클래스로 구축하고, 각 방법에 대해서는 하위클

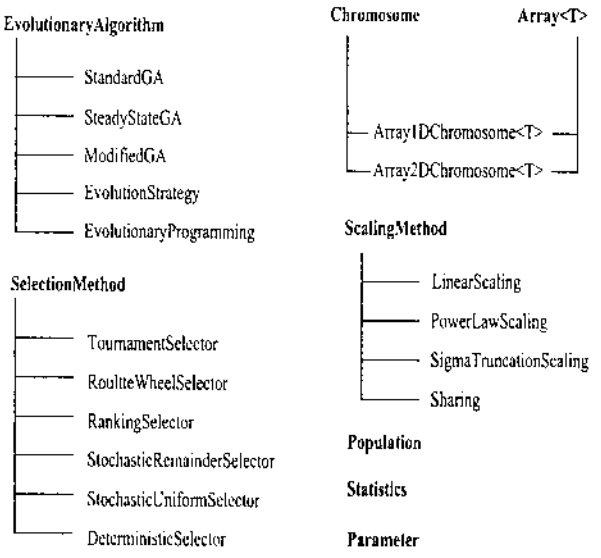


그림 4. 클래스의 계층적 구조.

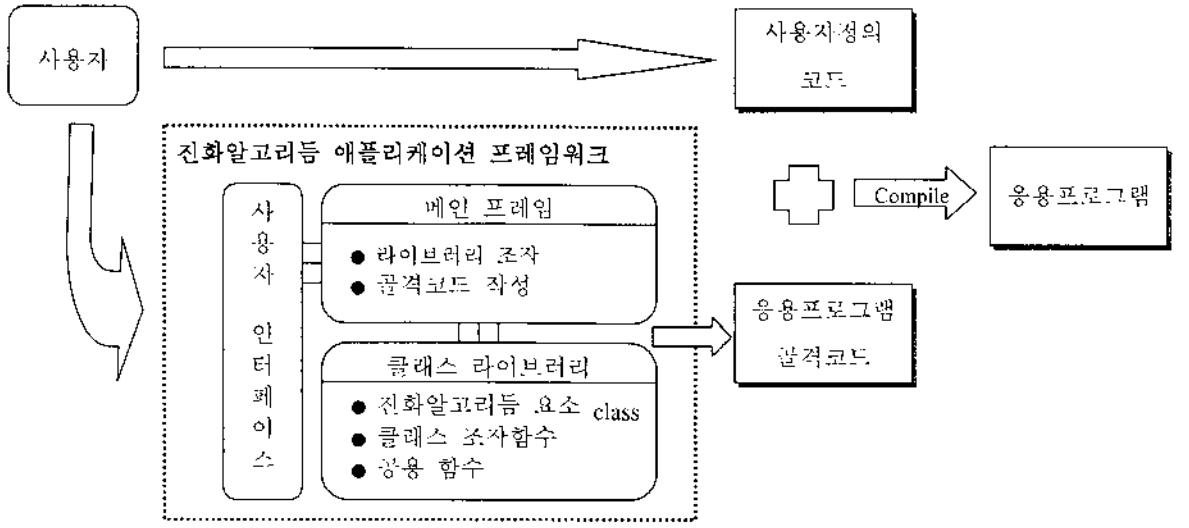


그림 5. 개발한 소프트웨어의 구조 및 적용.

래스를 통해 구체적으로 구현하였다. 하위클래스는 상위클래스의 멤버함수를 계승받기 때문에 사용자가 자신이 개발한 방법을 라이브러리에 추가할 경우 관련 상위클래스로부터 상속받아 자신의 특성 멤버함수만을 구현하면 된다. 따라서 이러한 상속관계는 라이브러리의 확장에 있어서 많은 유연성을 제공한다. 그 밖의 모집단, 파라메타, 통계량에 관한 클래스는 독립적인 클래스로 구현하였다. 본 연구에서 설계한 클래스의 계층적 구조는 <그림 4>와 같다. 또한 <표 2>는 이러한 클래스 가운데 상위클래스에서 정의된 멤버데이터와 멤버함수 중에서 대표적인 내용만을 정리한 것이다.

한편, 클래스 라이브러리에 초기 모집단을 임의로 생성하는 방법과 유전연산자, 난수 발생, 그리고 알고리듬의 수행시간 측정 등을 위한 공용함수를 라이브러리에 추가하였다. 여기서, 1차원 배열표현에 대해서 대표적인 표현방법에 따라 초기 모집단을 임의로 생성하는 방법을 공용함수에 포함하였다. 사용자는 라이브러리에 포함되어 있지 않은 진화알고리듬의 요소를 사용하려 할 때 라이브러리의 클래스 상속을 통하여 자신의 클래스를 구현할 수 있다. 또한 사용자는 개발된 라이브러리에 사용자가 필요한 코드를 추가함으로써 라이브러리 함수에 대한 추가가 가능하다.

4. 애플리케이션 프레임워크

4.1 소프트웨어의 구조

본 연구에서 개발하는 소프트웨어는 다양한 문제에 대한 진화알고리듬의 응용프로그램을 구축하는데 사용되는 일반적인 용도의 소프트웨어이다. 따라서 응용프로그램을 정의하는

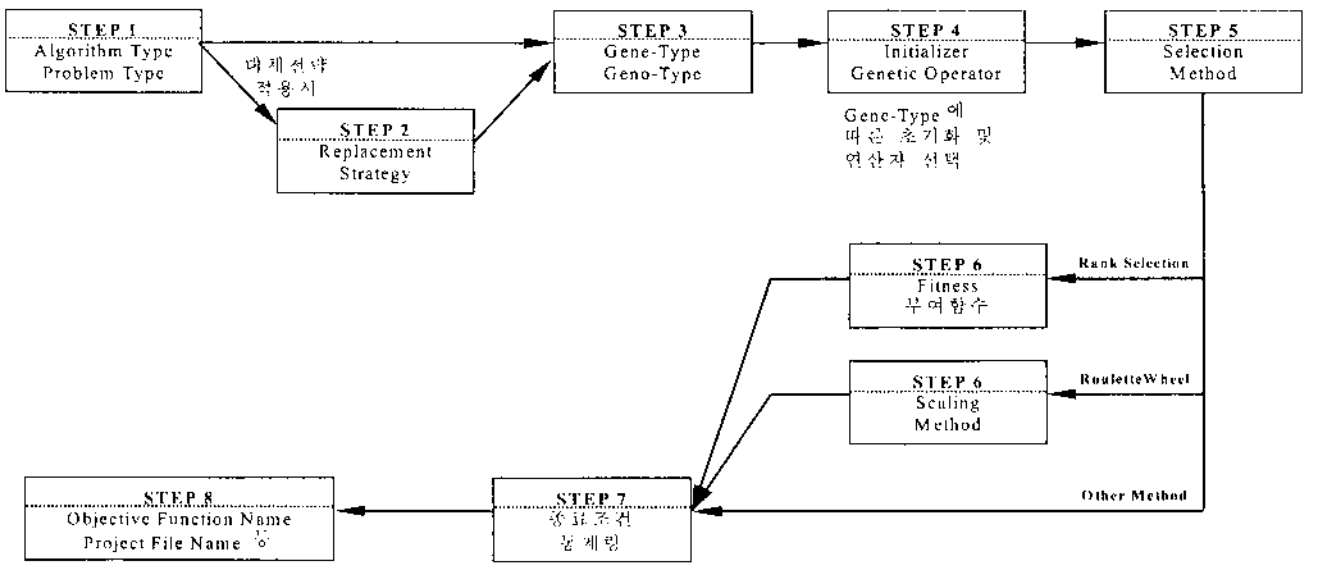


그림 6. 진화알고리듬 요소 선택 흐름도.

데에 필요한 공통의 기능을 효율적으로 구조화하여 클래스 라이브러리로 구현하고, 이를 바탕으로 여러 종류의 라이브러리 간의 접속기능과 접속된 라이브러리의 사용을 지원할 수 있도록 소프트웨어를 설계하고자 한다. 본 연구에서 개발하고자 하는 애플리케이션 프레임워크의 개발원칙과 특징은 다음과 같다.

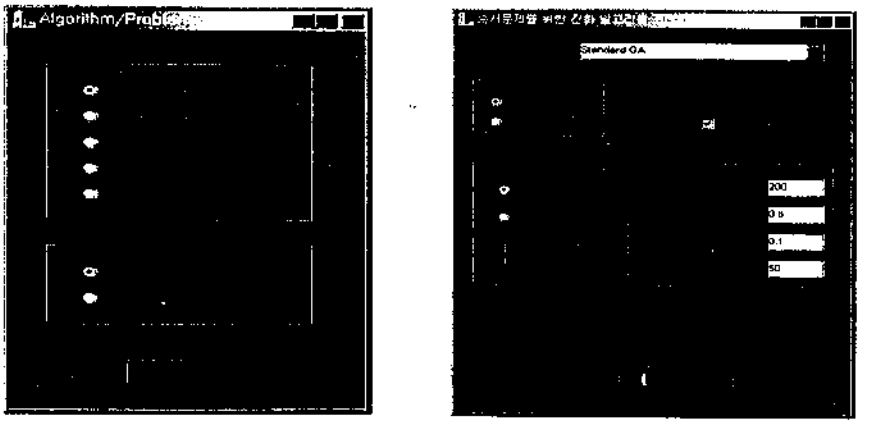
1) 최근 가장 보편적으로 사용되고 있는 OS인 Windows 95 환경에서 운영할 수 있도록 한다.
2) 애플리케이션 프레임워크의 효율적인 구축을 위해 객체지향 프로그래밍 언어인 C++을 사용한다. 따라서 사용자가 작성하는 부분은 C와 C++의 사용이 모두 가능하다.
3) 진화프로그램 요소의 다양한 선택을 할 수 있도록 사용자와 대화형 방식을 사용한다.
4) 진화프로그램의 모든 요소에 대해 제공되는 라이브러리를 사용하지 않고 사용자가 직접 작성할 수 있도록 한다. 따라서 사용자가 새로 개발한 알고리듬이나 진화알고리듬 요소를 사용자의 응용프로그램에 쉽게 추가할 수 있다.
5) 진화알고리듬의 수행결과를 텍스트 형태로 파일과 화면으로 제공할 뿐만 아니라 탐색과정과 통계량을 그래프로도 확인할 수 있도록 한다.

앞서 언급하였듯이 본 연구에서 개발한 애플리케이션 프레임워크는 사용자 인터페이스와 클래스 라이브러리, 그리고 메인프레임으로 구성된다. 사용자 인터페이스는 사용자가 원하는 방법을 적용한 응용프로그램을 생산하기 위해 사용자의 의도를 수용하는 환경을 제공하고, 사용자가 선택한 요소별 방법을 스크립트(script) 파일 형태로 저장하는 부분이다. 메인 프레임은 사용자 인터페이스에서 생성된 스크립트 파일과 3장에서 개발한 클래스 라이브러리를 이용하여 사용자 응용프로그램의 전체적인 골격코드를 생성해 주는 기능을 갖는다. 본 연구에서 개발한 소프트웨어의 구조와 적용과정은 <그림 5>와 같다. 먼저 사용자는 인터페이스(interface)를 통해 적용하고자 하는 진화알고리듬 구성요소를 선택한다. 이때 사용자가 정의하고자 하는 요소에 대해서는 인터페이스에서 제공하는 <사용자 정의> 항목을 선택한다. 모든 요소에 대해 사용자의 선택이 이루어 지면, 메인프레임은 클래스 라이브러리를 이용하여 사용자가 작성하고자 하는 진화알고리듬의 골격코드를 작성한다. 마지막으로 사용자는 제공된 골격코드에 자신이 정의한 코드를 추가함으로써 최종적인 응용프로그램이 완성된다.

4.2 사용자 인터페이스

사용자 인터페이스는 사용자의 요구를 빠짐없이 수용해야 하고, 사용자가 쉽게 사용할 수 있는 기능을 제공해야 한다. 진화알고리듬은 여러 구성요소로 이루어져 있으며, 각 요소별로 전략이나 방법이 매우 다양하고 복잡하다. 따라서 본 연구에서는 사용자로 하여금 진화알고리듬의 기본 절차에 따라 자연스럽게 여러 요소에 따른 방법이나 전략을 단계적으로 선택할 수 있도록 하였다. 그리고 사용자가 선택한 요소별 방법을 스크립트(script) 파일로 저장하였다. 스크립트 파일은 진화알고리듬의 구성요소별로 사용자가 선택한 방법을 저장하고 나열한 텍스트 파일로, 응용프로그램의 골격코드를 작성하는 메인프레임에 사용자의 의도를 전달하는 역할을 담당한다.

본 연구에서는 다음 <그림 6>과 같은 요소선택 흐름도에 따라 사용자가 요소의 선택에 따른 관련 전략을 선택할 수 있도록 한다. 단계1에서는 알고리듬의 형태와 적용문제가 최소화 또는 최대화 문제인가를 선택하도록 한다. 만약 단계1에서 안정상태 유전알고리듬 또는 진화전략을 선택한 경우, 단계 2에서 대체전략을 선택하도록 하고, 그 밖의 경우에는 단계 3으로 간다. 단계 3에서는 개체의 표현 방법을 선택할 수 있도록 하며, 이때 인자의 길이를 입력하도록 한다. 선택한 표현방법에 따라 단계4에서 초기화 방법과 유전연산자를 선택하며, 단계 5에서는 선별방법을 선택한다. 이때 선택된 선별방법에 따



(a)wizard 형태의 인터페이스 (b)옵션창(option window) 형태의 인터페이스

그림 7. 사용자 인터페이스 화면.

라 단계 6에서 부가적인 선택이 이루어지도록 한다. 단계 7에서는 종료조건과 필요한 통계량을 선택하고, 마지막으로 목적함수 파일명과 프로젝트 파일명을 입력하도록 한다.

이러한 절차를 통해 사용자가 진화알고리듬 요소를 선택할 수 있도록 사용자 편의성과 입력오류를 가능한 최소로 할 수 있는 마법사(wizard) 형태의 인터페이스를 구현하였다. 이때 구현된 마법사 형태의 인터페이스는 요소선택 흐름도에 따라 단계 8로 구성된다. 반면 마법사 형태는 특정 선택정보를 수정하려 할 경우 전 단계를 계속 거슬러 올라가야 하기 때문에 진화알고리듬에 익숙한 사용자에게는 단점이 될 수 있다. 따라서 본 연구에서는 마법사형태의 인터페이스와 함께 옵션창(option window) 형태의 대화상자(dialog box)를 제공한다. 옵션창은 윈도우즈 환경에서 보편적으로 사용되는 형태로 진화알고리듬에 익숙한 사용자가 다양한 방법이나 전략을 빠르게 선택할 수 있다. 다음 <그림 7 (a)>는 마법사 형태의 인터페이스를, <그림 7 (b)>는 옵션창 형태의 인터페이스를 나타낸 것이다.

4.3 메인 프레임

메인프레임은 기본적이면서 완벽한 진화알고리듬 소프트웨어로서의 기능을 제공해야 하고, 라이브러리 내의 자원과 사용자 정의 부분을 효과적으로 통합하고 사용자 애플리케이션 골격코드를 적절히 작성할 수 있는 기능을 제공해야 한다. 본 연구에서 개발한 메인프레임은 코드 템플릿(code template)과 코드 생성기(code generator)로 구성된다. 코드 템플릿은 시스템 내에 미리 작성된 템플릿 형태의 파일로, 진화알고리듬의 구성요소별 방법에 대한 템플릿과, 최종 응용프로그램의 인터페이스를 위한 템플릿, 그리고 사용자가 정의한 방법들에 대한 주석을 붙여주는 역할을 위한 템플릿 등으로 구성된다. 코드 생성기는 사용자 인터페이스에서 생성한 스크립트 파일과

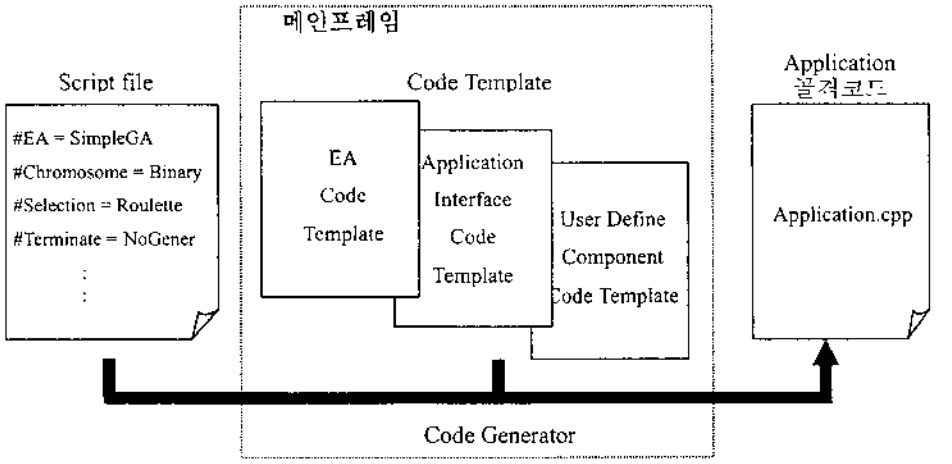


그림 8. 메인프레임의 구동원리.

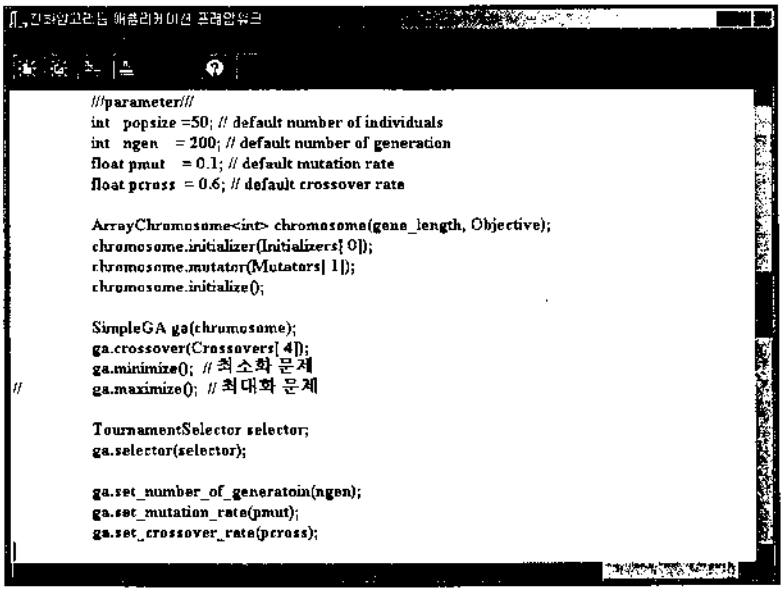


그림 9. 메임프레임에 의해 생성된 코드.

코드 템플릿을 이용하여, 사용자의 최종 응용프로그램의 골격 코드를 생성하는 역할을 수행한다. 개발된 메인프레임의 구동 원리는 <그림 8>과 같다.

먼저, 코드 생성기는 사용자 인터페이스에서 생성한 스크립트 파일을 해석하여 사용자의 의도를 파악한다. 그리고 미리 작성된 코드 템플릿 가운데 응용프로그램에 적용할 진화알고리듬 구성요소별 방법에 맞는 코드부분을 추출하여 최종 응용프로그램의 골격코드를 생성한다. 이때 코드 템플릿에 미리 작성된 코드는 개발된 클래스 라이브러리를 사용한 것으로, 특정 컴파일러에 종속적이지 않도록 코드를 작성하였으며, 매우 간결하면서 강건한 코드를 갖도록 하는데 중점을 두었다. 특히 최종 응용프로그램의 인터페이스는 윈도우즈의 API함수를 이용하여 구현함으로써 사용자에게 많은 유연성을 제공하고자 하였다. 이때 진화알고리듬의 특성상 운용환경이나 결과가 유사하기 때문에 개발의 편의를 위하여 파라메터 입력창을 제외한 나머지 인터페이스는 모두 동일하게 구현하였다. 그러나 사용자 응용프로그램의 인터페이스는 추후 더욱 보완할 예정이다.

<그림 9>는 메인프레임에 의해 자동으로 생성된 애플리케이션 골격코드를 보여준다. 사용자는 메인프레임에 의해 생성된 코드를 기반으로 평가함수와 같은 자신의 응용프로그램에 특수한 부분에 대한 코드만을 추가 작성함으로써 최종 응용프로그램이 완성된다.

5. 적용사례

본 장에서는 대표적인 순서 문제인 외판원문제에 대해 본 연구에서 개발한 애플리케이션 프레임워크를 이용한 적용 예를 보이고자 한다. 외판원문제에 애플리케이션 프레임워크 적용을 위해 <그림 6>에서 제시한 진화알고리듬 요소 선택 흐름도에 따라 요소를 선택한다. 예제문제는 20개의 도시 문제로, 사용된 진화알고리듬 전략들은 <표 3>과 같다.

마법사 형태의 인터페이스를 사용할 경우, 요소의 선택과정은 다음과 같다. 단계 1에서는 진화알고리듬 형태와 문제형태를 선택하는데, 진화알고리듬 형태로 표준유전 알고리듬을 선택하고, 문제형태로는 최소화를 선택한다. 이때 elitism전략을 선택할 수 있다. 단계 1에서 표준유전 알고리듬이 선택되었으므로 단계 2의 인터페이스는 나타나지 않는다. 단계 3에서 선택하는 개체형태로 예제에서 경로표현을 사용하므로 1차원 배열을 선택한다. 인자형태는 각 도시들이 개체의 인자로 표현되므로 정수형을 선택한다. 개체길이는 전체 도시의 수가 20이므로 개체길이 입력란에 20을 입력한다. 모집단 크기에서는 50을 선택한다. 단계 3에서 개체형태로서 1차원 배열이 선택되었으므로 단계 4에서는 1차원 배열형태로서 사용 가능한 유전연산자를 선택하게 된다. 단계 4의 초기화함수는 개체의 인자값이 중복되거나 누락됨이 없는 초기화함수로서 OrderInitializer를 선택한다. 그리고 교차는 부분사상교차, 돌연변이는 교환을 선택하고 각 파라메터 값을 선택한다. 단계 5에서는 선별방법으로 토너먼트선별을 선택한다. 선택된 토너먼트선별은 부가적인 선택이 필요 없기 때문에 단계 7로 간다. 단계 7에서는 종료조건으로서 세대수를 선택하고 필요한 통계량을 선택한다. 예제에서는 세대별 최선해와 평균, 그리고 온라인(online) 성능과 오프라인(offline) 성능을 선택하였다. 마지막 단계 8에서는 목적함수 파일과 프로젝트 파일명을 입력한다.

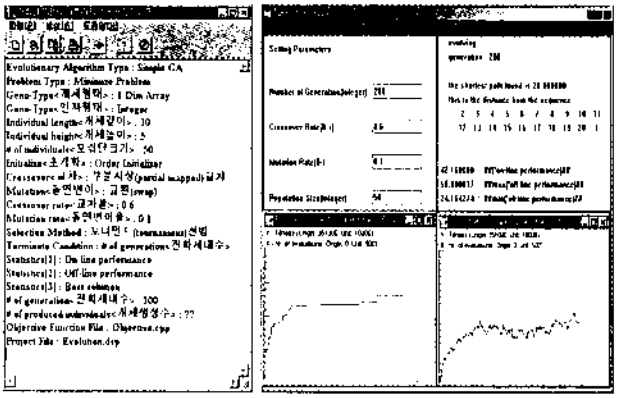
각 단계별 선택이 끝나면 <그림 10> (a)와 같이 외판원문제를 위해 선택된 진화알고리듬 전략이 면에 나타난다. 선택정보 확인 후 메뉴에서 생성을 선택하여 프레임워크에 의해 생성된 외판원문제의 골격코드를 저장한다. 최종적으로 생성된 골격코드에 외판원문제와 관련하여 필요한 부분을 추가한 후 최종 소스코드를 생성하여 실행하면 <그림 10> (b)와 같이 알고리듬 수행 후 각종 통계치 등의 정보를 화면에 출력한다.

표 3. 예제에서 사용된 구성요소

구성요소	사용방법
진화알고리듬 형태	표준유전 알고리즘
목적함수형태	최소화문제
개체표현	1 차원 배열표현(정수)
개체길이	20
선별방법	토너먼터선별
초기화함수	OrderInitializer
교차	부분사상교차(PMX)
돌연변이	교환(swap)

6. 결 과

본 연구에서는 단순진화 알고리듬을 위한 소프트웨어를 애플리케이션 프레임워크 형태로 개발하였다. 이를 위해 진화알고



(a) 최종 선택 내용 (b) 실행 결과 화면

그림 10. 최종 실행결과.

리듬에 대한 객체지향적 모델링을 통해 클래스 라이브러리를 구축하고, 구축된 라이브러리를 이용, 이들을 구조화 시킴으로써 사용자가 작성하고자 하는 응용 프로그램의 골격코드를 생성해 주는 프레임워크를 개발하였다. 이때 사용자 편의성을 중시한 설계로 사용자로 하여금 최소한의 작업으로 진화알고리듬을 통한 문제 해결을 가능케 하고, 새로 개발된 요소나 방법의 추가가 가능한 소프트웨어의 개발에 중점을 두었다.

개발된 애플리케이션 프레임워크를 통하여 생성된 사용자 응용프로그램은 코드가 매우 간결하고 생성된 코드의 높은 가독성(readability)으로 인하여 쉽게 다른 문제로의 응용이 가능하다. 또한 생성된 코드를 통하여 사용자는 강건한 진화알고리듬 애플리케이션의 개발이 가능하다. 또한 라이브러리의 확장성을 통하여 향후 개발될 여러 진화알고리듬의 기법들을 효과적으로 라이브러리에 추가시킴으로써 확장성과 활용도가 높은 진화알고리듬 소프트웨어가 될 것이다. 향후 단순진화알고리듬과 최근 진화계산 분야에서 활발히 연구되고 있는 공진화 알고리듬을 모두 지원할 수 있는 소프트웨어를 개발할 예정이다.

참고문헌

김여근, 윤복식, 이상복 (1997), *메타휴리스틱*, 영진문화사

이주헌 (1995), *실용소프트웨어 공학론*, 법영사

Adam, F. and Thomas. W., GPC++, *http://www.emk.e-technik. th-darmstadt. de/~thomasw/gp.html.*

Art, C. (1993), Libga100, Available by anonymous ftp from *ftp://www.aic.nrl. navy.mil/pub/galist/src/.*

Automatic Control & Systems Engineering, GA Toolbox for Matlab, *http://www. shef.ac.uk/uni/projects/gaipp/gatbx.html.*

Fogel, L. J., Owens, A. J. and Walsh, M. J. (1966), *Artificial Intelligence throughSimulated Evolution*, NY: John Wiley.

George, P. W. and Williams, J. (1993), GECO, Avariable by anonymous ftp from *ftp://www.aic.nrl.navy.mil/pub/galist/src/.*

Holland, J. H. (1975), *Adaptation in Neural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor.

Matthew, W. (1995), Matthew's Galib, *http://lancet.mit.edu/ga/.*

Mike, V. L. (1992), Paragenesis, Available by anonymous ftp from *ftp://www. aic.nrl.navy.mil/pub/galist/src/.*

Michalewicz (1994), *Genetic algorithms + Data structures=Evolution programs*, 3rd Ed., Springer-Verlag, Berlin.

New Light Industries Ltd. (1995), Generator, *http://www.iea.com/~nli/GA- /generator.html.*

Roman, V. (1998), Amber Alife Wasps, *http://www.geocities.com/CapeCanaveral /2971/awasps.html.*

Smith, J. R. (1992), BUGS, Available by anonymous ftp from *ftp://www.aic. nrl.navy.mil/pub/galist/src/.*

Taligent (1993), Inc., *Building Object-Oriented Frameworks.*

Terry, Q. (1998), *Visual Modeling with Rational Rose and UML*, Addi son-Wesley.

TransferTech GmbH, EVO(Evolutionary Optimizer), *http://www.transfertech.de-/www/evse_gen.htm.*

Wall, M. (1996), GAlib: A C++ library of GeneticAlgorithm Components. Available by anonymous ftp from *ftp://lan.et.mit.edu/ga.*

William, M. S. (1994), GAC, Available by anonymous ftp from *ftp://www. aic.nrl.navy.mil/pub/galist/src/.*

William, M. S. (1992), GAL, Available by anonymous ftp from *ftp://www.aic.nrl. navy.mil/pub/galist/src/.*

Zbigniew, M. (1992), GENEsYs, Available by anonymous ftp from ftp://www. aic.nrl.navy.mil/pub/galist/src.

Flexible Intelligence Group, LLC., FlexTools, http://www.flextool.c.

이수연

1992년 전남대 산업공학과 석사

1999년 전남대 산업공학과 박사

현재: (주)도울정보기술 개발팀장, 동신대 컴퓨터학과 겸임교수

관심 분야: Evolutionary Algorithm의 응용 및 확장, ERP, 시스템 분석 및 운용 S/W개발 등

정호연

1988년 서울대 산업공학과 석사

1994년 서울대 산업공학과 박사

현재: 전주대 기계산업공학부 부교수

관심 분야: 네트워크 최적화, GIS응용, Evolutionary Algorithm의 응용

서광언

1998년 전남대 산업공학과 학사

현재: 전남대 산업공학과 석사과정

관심 분야: S/W개발, Evolutionary Algorithm의 응용 및 확장, Web응용 등

김여근

1979년 서울대 산업공학과 석사

1986년 서울대 산업공학과 박사

현재: 전남대 산업공학과 교수

관심 분야: Evolutionary Algorithm, Tabu Search와 Simulated Annealing의 이론과 응용, 조립라인의 설계와 분석, Scheduling, 통신망 설계 및 운용 등