퍼지 및 지능시스템학회 논문지 2005, Vol. 15, No. 6, pp. 730-735

효과적인 배낭 문제 해결을 위해 DNA 코딩 방법을 적용한 DNA 컴퓨팅

DNA Computing Adopting DNA coding Method to solve effective Knapsack Problem

김은경* · 이상용**
Eun-Gyeong Kim · Sang-Yong Lee

* 공주대학교 컴퓨터 공학과
** 공주대학교 정보통신공학부

요 약

배낭 문제는 단순한 것 같지만 조합 최적화 문제로서, 다항 시간(polynomial time)에 풀리지 않는 NP-hard 문제이다. 이 문제를 해결하기 위해 기존에는 GA(Genetic Algorithms)를 이용하여 해결하였다. 하지만 기존의 방법은 DNA의 정확한 특성을 고려하지 않아, 실제 실험과의 결과 차이가 발생하고 있다.
본 논문에서는 배낭 문제의 문제점을 해결하기 위해 DNA 컴퓨팅 기법에 DNA 코딩 방법을 적용한 ACO(Algorithm for Code Optimization)를 제안한다. ACO는 배낭 문제 중 (0,1)-배낭 문제에 적용하였고, 그 결과 기존의 방법보다 실험적 오류를 최소화하였으며, 또한 적합한 해를 빠른 시간내에 찾을 수 있었다.

Abstract

Though Knapsack Problem appears to be simple, it is a NP-hard problem that is not solved in polynomial time as combinational optimization problems. To solve this problem, GA(Genetic Algorithms) was used in the past. However, there were difficulties in real experiments because the conventional method didn't reflect the precise characteristics of DNA.
In this paper we proposed ACO (Algorithm for Code Optimization) that applies DNA coding method to DNA computing to solve problems of Knapsack Problem. ACO was applied to (0,1) Knapsack Problem; as a result, it reduced experimental errors as compared with conventional methods, and found accurate solutions more rapidly.

Key Words : DNA 컴퓨팅, DNA 코딩 방법, 배낭 문제

1. 서 론

배낭 문제는 조합 최적화 문제로서, 다항 시간(polynomial time)에 풀리지 않는 NP-hard 문제이다. 이 문제를 해결하기 위해 보통 GA를 사용하며, 이는 자연의 생물학적 진화 원리에 기반하여 확률적인 검색과 최적화를 수행한다. 또한 계산에 의한 전통적인 최적화 방법과 비교하면, GA는 강인성과 어떠한 문제에도 쉽게 적용될 수 있는 범용성을 갖고 있다.

하지만 이러한 장점에도 불구하고 GA는 탐색에 있어 문제점을 갖고 있다. GA는 전역탐색(exploration)과 지역탐색(exploitation)을 사용하는데, 전역탐색만 수행할 경우 무작위 검색과 같은 결과를 나타낸다. 그리고 지역탐색만을 수행할 경우 국부 최적해를 찾는 결과를 초래하게 된다. 또한 개체 집단의 다양성과 다음 세대의 선택 강도가 각각 전역탐색과 지역탐색에 의해 결정되므로, 선택 강도의 증가와 감소는 개체 집단의 다양성을 감소 또는 증가시킨다[1]. 따라서 검색 초기에는 개체 집단의 다양성을 강조하고, 검색이 진행될수록 점차로 선택 강도를 높여 지역탐색을 강조해야 한다. 하지만 검색 조절의 적합한 시점을 판단하는 것은 쉽지 않다.

본 논문에서는 이러한 문제점들을 해결하기 위해 다음과 같은 방법을 사용하였다. 우선 조합 최적화 문제인 배낭 문제를 해결하기 위해서 DNA 컴퓨팅을 사용하였다. DNA 컴퓨팅은 1994년 Adleman이 NP-complete 문제인 해밀토니안 경로 문제(Hamiltonian Path Problem, HPP)를 생물학 실험 과정만으로 해결함으로써, 조합 최적화 문제들을 효율적으로 해결할 수 있다[2]. 그리고 DNA 컴퓨팅에 GA의 문제점들을 잘 해결할 수 있는 방법으로 DNA 코딩 방법을 적용하였다.

즉, 본 논문에서는 DNA 컴퓨팅 알고리즘에 DNA 코딩 방법을 적용한 ACO를 사용해 배낭 문제 중 하나인 (0,1)-배낭 문제에 적용하여 실험하였다. 그 결과 ACO는 막대한 병렬성과 저장능력을 갖는 DNA 컴퓨팅으로 빠른 탐색과 우수한 해를 효율적으로 찾을 수 있었으며, DNA 컴퓨팅에 적용한 DNA 코딩 방법으로는 GA의 문제점들을 잘 해결할 수 있었다.

접수일자 : 2005년 2월 4일
완료일자 : 2005년 5월 17일
감사의 글 : 이 논문은 2005년도 두뇌한국21사업에 의하여 지원되었음

2. 관련 연구

2.1 배낭 문제

배낭 문제(knapsack problem)는 단순한 것 같지만 수많은 조합형 특성을 탐색할 수 있게 해주는 NP-hard 문제로서, 그 해를 얻기가 대단히 어렵다. 이 문제는 사전에 무게와 이익이 사전에(삭제) 알려진 품목(물체)들의 집합이 주어질 때, 무게의 한계를 초과하지 않으면서 전체 이익이 최대가 되도록 배낭을 채우는 문제를 말한다[3][4]. 즉, n개의 품목과 하나의 배낭이 있을 때, 식(1)과 같이 이득 F(x)를 최대로 하는 품목을 선택하는 문제이다.

$$F(x) = \sum_{i=1}^{n} p_i x_i \qquad \text{식(1)}$$

배낭의 용량은 W이며, 식(2)의 조건을 만족해야 한다.

$$\sum_{i=1}^{n} w_i x_i \leq W \qquad \text{식(2)}$$

이때 $x=(x_1,\cdots,x_n)$이며, x_i는 0또는 1의 값을 갖는다. 또한 p_i는 품목 i의 이득이며, w_i는 품목 i의 무게를 나타낸다.

x_i의 값에 0과 1사이의 소수를 허용하면 분할가능 배낭 문제(fractional knapsack problem)라고 하며, 0과 1만 허용하면 (0,1)-배낭 문제라고 부른다. 분할가능 배낭 문제에서는 품목들을 쪼개서 배낭에 넣을 수 있으나, (0,1)-배낭 문제에서는 품목을 통채로 배낭에 넣든지 아니면 버리든지를 결정해야 한다. 분할가능 배낭 문제는 욕심쟁이법으로 간단히 해결할 수 있다. 그러나 (0,1)-배낭 문제는 NP-hard에 속하는 문제로서 욕심쟁이법과 동적계획법 등으로 다항 시간에 풀리지 않는 어려운 문제이며, 이를 풀기 위해서는 $\Omega(2n)$의 시간이 요구된다[5][6][7].

2.2 DNA 컴퓨팅

DNA 컴퓨팅은 실제 생체 분자인 DNA나 RNA와 같은 살아 있는 세포를 응용한 것으로, 합성 DNA를 정보소자로 사용하여 풀고자 하는 문제에 대한 해법을 DNA 코드로 기술한다. 그리고 주어진 DNA 조각들로부터 화학적으로 합성, 검출함으로써 답을 찾아 내는 기술을 말한다.

DNA는 4개의 염기인 A(Adenine), T(Thymine), C(Cytosine), G(Guanine)가 2중 나선 구조로 구성되어 있다. 이들 염기에 대용량 데이터를 저장할 수 있는 메모리 기능을 가지고 있으며, 정해진 규칙에 의해 상호 보완적인 방식의 Watson-Crick 결합을 하고 있다[8].

그리고 복잡한 염기 조합의 패턴은 하나의 유전 정보를 담고 있으며, 인체내에서 자연 발생하는 효소에 의해 읽혀지고 있다. 효소는 생물학 실험 방법들과 함께 DNA 컴퓨팅의 연산자로 사용되고 있다. 따라서 일반 컴퓨터의 연산자를 사용하지 않고, 여러 가지 실험 과정들을 연산자로 사용한다. 대표적으로 Melting과 Annealing, Ligation, Polymerase Chain Reaction(PCR), Enzyme reaction, Gel Electrophoresis, Antibody Affinity가 있다[9].

[그림 1]는 Adleman의 DNA 컴퓨팅 알고리즘을 나타낸 것이다. Adleman은 주어진 문제에 맞게 DNA 코드로 표현하고, 생성된 DNA 코드를 단 한번의 합성과 분리 과정을 거쳐 조합 최적화 문제를 해결하였다[2].

이러한 DNA 컴퓨팅의 특징을 살펴보면 매우 낮은 에너지로 작동되기 때문에 많은 에너지가 필요없다. 그리고 나노 수준의 막대한 병렬성을 이용하여 NP-complete에 효과적인 접근이 가능하게 되었다. 또한 계산 속도와 정보의 저장 및 처리 효율에서도 우수함을 보이고 있다[10][11].

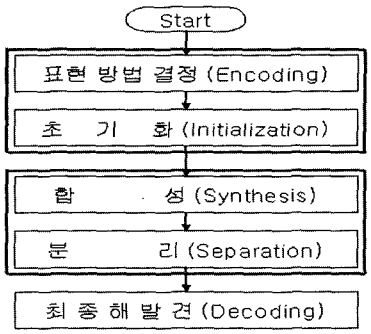

그림 1. Adleman의 DNA 컴퓨팅 알고리즘
Fig. 1. Adleman's DNA computing algorithm

2.2 DNA 코딩 방법

DNA 코딩 방법은 1995년 Yoshikawa가 제시한 변형된 형태의 유전자 알고리즘이다[12]. 일반적인 유전자 알고리즘은 0, 1의 이진수를 사용하지만, DNA 코딩 방법은 A(Adenine), G(Guanine), T(Thymine), C(Cytosine)의 4진수를 사용하여 선택, 재생, 교배, 돌연변이 연산을 한다. 그리고 A, T, G, C 중 3개의 염기(코돈 ; codon)가 하나의 의미 단위인 아미노산을 지정하며, 그 수는 중복을 제외한 20가지가 있다.

그림 2. DNA 염색체의 번역 예
Fig. 2. Example of DNA chromosome translation

그림 2에서 보는 것처럼 염기서열은 시작 코돈인 ATG에서부터 정지 코돈인 TGA(TAA, TAG)까지 아미노산 번역을 한다. 이러한 아미노산 번역은 짧은 DNA 코드에서도 많은 정보를 얻을 수 있다.

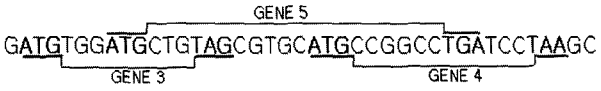

그림 3. 유전자 중복의 예
Fig. 3. Example of gene overlap

DNA 코딩 방법의 특징은 그림 3에서 보는 것처럼 염색체의 중복을 허용함으로써, 새로운 염색체를 만들 수 있다. 그리고 하나의 아미노산을 만드는 코돈이 여러 개이므로 지식 표현이 쉬우며, 교배점이 임의로 주어지기 때문에 염색체의 길이가 가변적이다. 이러한 가변길이 표현방법은 염색체의 길이가 길수록, 고정길이 표현 방법 보다 성능이 훨씬 우수하며 다양한 개체군을 생성한다. 이러한 특징들은 염색체의 기능과 동작을 더욱 생물학적으로 가깝게 모델링할 수 있다.

3. ACO

본 논문은 배낭 문제를 GA로 해결할 때, 발생했던 문제점

들을 해결하기 위해 Adleman의 DNA 컴퓨팅 알고리즘을 보완한 ACO(Algorithm for Code Optimization)를 제안한다. ACO는 기존의 DNA 컴퓨팅에 DNA 코딩 방법을 적용하여 DNA 코드를 생성한다. 그리고 DNA 코딩 방법의 연산자와 생물학 연산자를 이용하여 반응횟수 만큼 합성과 분리 과정을 반복 처리하여 해를 찾는다.

ACO의 특징은 배낭 문제에 대하여 DNA의 특징을 최대한 반영한 코드를 생성할 수 있으며, 비효율적인 탐색으로 인한 시간과 노력이 낭비되는 것을 줄일 수 있다. 또한 반복적인 합성과 분리 과정을 통해 생물학적 실험 오류율을 최소화하여 많은 해를 탐색할 수 있다.

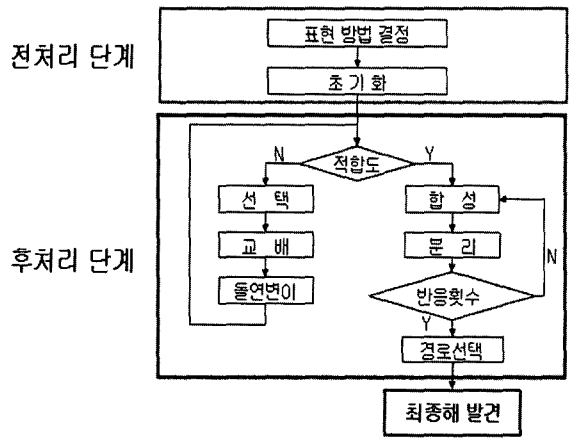

그림 4. ACO의 흐름도
Fig. 4. Flowchart of ACO

그림 4는 ACO의 전체 흐름도를 표현한 것으로 전처리 단계와 후처리 단계로 구분된다. 그리고 그림 5는 ACO 알고리즘이다.

```
1. Encoding
      - DNA 코딩 방법 적용
2. Initialization
3. While(Fitness estimation)
      - Fitness → Selection → CrossOver → Mutation
4. while(BioOperation)
  가. Tube Synthesis
    1) Synthesis
      - 각 n개-튜브에 template와 상보적 template 삽입
    2) Union
      - n개-튜브 -> 1개의 튜브
    3) Annealing
  나. Separation
    1) Cut
      - 제한 효소들로 DNA 서열 절단
    2) Labeling
      - DNA 서열 -P(Dephosphorylating)로 표지화
    3) CuttingOut
       i ) Target DNA 추출하여 각각에 tube에 삽입
       ii ) (TargetT1(wi) == max)
            TargetT1(max)+P(phosphorylating)로 표지화
       iii) i )과 ii )를 Union -> TargetU
    4) Ligation
      - {TargetU(wi)<W , TargetU(pi)<P}
        -> TargetR을 추출
5. Decoding(최종해 발견)
```

그림 5. ACO 알고리즘
Fig. 5. ACO algorithm

먼저 ACO의 전처리 단계를 살펴보면, 표현 방법 결정 과정과 초기화 과정으로 나뉜다. 표현 방법 결정 과정에서는 주어진 DNA 코드를 DNA 코딩 방법으로 품목 i의 무게 w_i와 이득 p_i를 생성한다. w_i와 p_i는 바로 DNA 코드로 표현할 수 없으므로 그림 6에 따라 생성한다.

```
Initialize {
 W_Node = Weight_DNA,
 P_Node = Profit_DNA,
 DNA_seq = (DNA sequence),
 Start_Codon = "ATG"
      }
Begin
 DNA_Len = length(DNA_seq)
 temp_DNA_seq = DNA_seq

 For(;DNA_Len;) {
 temp_index = InString(temp_DNA_seq,Start_Codon)
 temp_DNA_seq = Middle(DNA_seq,temp_index+3)
 temp_next_index = InString(temp_DNA_seq,Start_Codon)
 i= i+1

 if(i % 2 == 0) {
  Weight_DNA(i) = Start_Codon+Middle(
    temp_DNA_seq, temp_index+3,temp_next_index)
  hydrogen_Com(Weight_DNA)
  temp_DNA_seq = Middle(temp_DNA_seq,
    temp_next_index)
 }else {
  Profit_DNA(i-1) = Start_Codon+Middle(
    temp_DNA_seq, temp_index+3,temp_next_index)
  Label_pro(Profit_DNA)
  temp_DNA_seq = Middle(temp_DNA_seq,
    temp_next_index)
  }
}
End
```

그림 6. 품목 i의 표현 프로시져
Fig. 6. Procedure of item(i) expression

먼저 시작 코돈(ATG)의 위치를 파악하고, i번째 시작 코돈에서 (i+1)번째 시작 코돈 전까지의 DNA 코드를 wi로, (i+1)번째 시작 코돈에서 (i+2)번째 시작 코돈 전까지의 DNA 코드를 pi로 표현한다. 그 다음 순서는 wi의 실제 무게를 표현하기 위해 A/T쌍과 G/C쌍의 수소 결합 수를 각각 낮은 무게와 높은 무게에 포함시켜서 wi를 생성한다. 그리고 pi는 방사능 라벨 즉, P(Phosphorylating)를 부착한다. 그림 7은 품목 i의 표현 예이다.

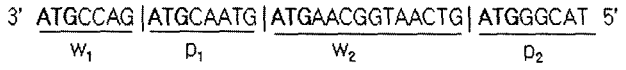

그림 7. 품목 i의 표현 예
Fig. 7. Example of item(i) expression

이렇게 표현된 품목 i인 w_ip_i와 품목 (i+1)인 $w_{i+1}p_{i+1}$의 연결은 상보결합인 $\overline{p_iw_{i+1}}$ 로 표현 한다. 그림 8은 품목 i와 품목 (i+1)에 대한 연결을 나타낸다.

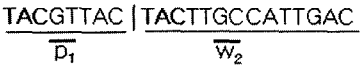

그림 8. 품목 i와 품목 i+1의 연결 예
Fig. 8. Example of join item(i) and item(i+1)

위와 같은 방법으로 표현 방법 결정 과정이 끝나면, 다음 과정인 초기화 과정에서 결정한 표현 방법을 반영한 DNA 코드들을 생성한다.

ACO의 후처리 단계를 살펴보면 적합도를 만족하지 않을 경우, DNA 코딩 방법의 연산자를 이용하여 적합도를 재평가한다. 만족하는 경우 반응횟수만큼 합성과 분리 과정을 거쳐 우수한 경로를 선택하여 최종해로 한다.

표 1. 아미노산 코드
Table 1. Amino acid code

Phe	16	Pro	3	His	15	Glu	13
Leu	7	Thr	5	Gln	11	Cys	6
Ile	8	Ala	1	Asn	9	Trp	19
Met	14	Tyr	18	Lys	12	Arg	17
Ser	2	Val	4	Asp	10	Gly	0

무게와 이득을 포함한 품목 i의 적합도 평가는 [표 1]의 아미노산 코드를 적용하여 비례 선택법(roulette wheel)을 역함수로 평가한다. 그리고 잘못된 결합이나 결합 위치 이동과 같은 생물학 실험에서 발생할 수 있는 오류의 조건을 미리 제거한다. 교배는 2점 교배를 하고 국소 해에 빠질 위험성을 벗어나기 위해 랜덤하게 교배점을 선택한다. 돌연변이는 임의의 염기쌍을 선택해 한 염기쌍을 변화시키는 방법을 사용하여 세대수 만큼 반복한다.

이렇게 하여 높은 적합도를 갖는 우수한 서열을 선택하고, 생성된 서열을 Tube에 넣어 반응횟수 만큼 반복하는 Tube Synthesis과 Separation을 수행한다. Tube Synthesis 과정을 살펴보면 Synthesis, Union, Annealing으로 구성된다. 먼저 Synthesis는 생성된 DNA 가닥의 문자열을 Tube에 넣는 [Char]->Tube의 구문을 갖으며, 생성된 DNA 가닥은 template와 상보적 template로 구분진다. 그리고 n개-Tube안에 각각 삽입한다. 그 다음 Union은 n개-Tube를 1개의 Tube에 혼합하는 과정을 말하며, Tube-> Tube->Tube의 구문을 갖는다. 마지막으로 Annealing은 단일 가닥의 DNA 서열을 정해진 수만큼 이중 가닥으로 결합하는 과정으로 Tube->Int->Tube의 구문을 갖다.

다음 과정으로 결합된 DNA 가닥들을 분리하는 Separation을 수행한다. 이 과정은 Cut, Labeling, CuttingOut, Ligation으로 구성된다. 먼저 Cut는 제한 효소를 이용하여 DNA 가닥을 특정 위치에서 잘라내는 것으로 Tube->[Char]->Tube의 구문을 갖는다. 다음으로 잘라진 DNA 가닥에서 P(Phosphorylating)를 제거하는 Labeling으로 Tube->[Char]->Int->Tube의 구문을 갖는다. Labeling은 +/-P(Phosphorylating/ Dephosphorylating), +/-B (Biotinylating/Debiotinylating), +/-C(set Cy5-label/remove Cy5-label)의 세부 연산들을 포함하고 있다. Labeling이 끝난 후 Tube->Int->Tube의 구문을 갖는 CuttingOut을 수행한다. CuttingOut은 세가지 세부 연산으로 구분된다. 먼저 Labeling 된 DNA 가닥들 중에서 필요한 가닥을 추출하여 Tube에 삽입한다. 다음으로 삽입된 각각의 Tube 중 길이가 가장 큰 DNA 가닥을 선택하여 P를 붙이는 Labeling을 해 준다. 마지막으로 앞에서 수행된 Tubes를 Union해서 TargetU를 생성한다. Separation의 마지막 처리 과정인 Ligation은 TargetU에 들어 있는 DNA 가닥 중 길이가 배낭의 용량(W)을 초과하지 않으며, 이득(P)에 대하여 최대가 되는 TargetR을 추출한다.

합성과 분리를 반응횟수 만큼 반복한 후 추출된 DNA 가닥 TargetR은 마지막으로 겔 전기 영동법을 이용하여 결과를 확인하고 Decoding 한다. 제안한 알고리즘의 과정을 거친 DNA 가닥들은 배낭 문제의 해를 만족하는 결과를 얻기에 충분하다.

4. 실험 및 분석

ACO의 성능을 확인하기 위해 [표 2]의 배낭 문제의 상태값을 이용하였고, GA와 ACO로 해결하여 비교 평가하였다. 단, 배낭에 담을 수 있는 용량(W)은 1664으로 조건을 만족해야 한다.

표 2. 배낭 문제의 상태값
Table 2. Value of Knapsack problem

번호(n)	이득(pi)	무게(wi)
1	62	975
2	83	689
3	42	527

시뮬레이션은 P-Ⅳ, 2GHz, RAM 512M의 PC에서 C언어를 사용하여 진행되었다. 실험에서 사용한 파라미터들은 [표 3]과 같이 설정하였다. 교배 연산 비율과 돌연변이 연산 비율은 최소 영역인 시작 코돈의 값을 지닌 DNA 코드의 교배를 고려하였다. 그리고 총 반응횟수는 반복횟수와 반응횟수를 곱한 값이며, ACO와 GA 모두 동일하게 1000으로 설정하여 실험에 대한 오차를 줄였다.

표 3. 파라미터들
Table 3. Parameter's

변 수		ACO	GA
집단 크기		1000	1000
세대수		100	100
교배 연산 비율		0.3	0.3
돌연변이 연산 비율		0.2	0.2
총 반응횟수	반복횟수	10	1
	반응횟수	100	1000
생물학적 실험 오류율		0.01	0.01

전처리 단계인 초기화 과정과 표현방법 결정 과정에서의 평가는 집단의 크기가 각각 200, 400, 600, 800, 1000에서 적

합도에 만족하는 초기 개체수를 비교하였다. [표 4]는 각 집단의 크기에서 적합도(F)에 만족하는 개체수로 ACO가 GA보다 높은 적합도에서 많은 수의 초기 개체수를 형성하면서 초기 개체군을 형성하였다.

표 4. 집단의 크기에 대한 초기 개체수
Table 4. Number of early individal about set size

집단의 크기	ACO의 초기 개체수		GA의 초기 개체수	
	0.5≦F<0.8	0.8≦F<1	0.5≦F<0.8	0.8≦F<1
200	7	1	8	0
400	8	1	10	0
600	8	3	9	2
800	6	4	13	3
1000	7	6	14	5

후처리 단계에서 적합도 평가는 초기 개체수가 동일하게 형성한 집단의 크기가 200인 경우와 600인 경우를 가지고 적합도를 비교하였다. 그림 9에서와 같이 ACO의 평균 적합도는 GA의 평균 적합도보다 고른 결과를 얻을 수 있었다. 그리고 초기 개체에서 좋은 부모 개체를 생성하여 배낭 문제의 용량에 충분히 만족시킬 수 있었다. 또한 각 세대별 평균 적합도에서도 ACO는 다음 세대에 영향을 주어 좋은 적합도를 얻을 수 있었다. 이는 ACO의 전처리 단계에서 생성된 가변 길이의 적은 DNA 코드로 배낭 문제의 최대 이득을 표현하는 것이 가능함을 확인하였다.

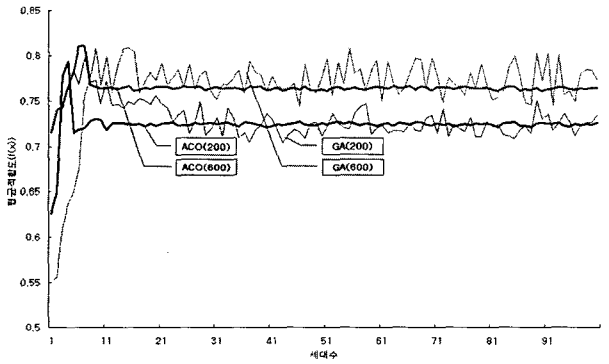

그림 9. 적합도 평가
Fig. 9. Fitness estimation

5. 결 론

본 연구에서는 배낭 문제를 기존의 방법인 GA를 이용한 DNA 코드 표현 방법의 문제점을 분석하고, 이를 해결하기 위해 DNA 컴퓨팅에 DNA 코딩 방법을 적용한 ACO를 제안하였다. ACO는 집단의 크기에 상관없이 GA보다 고른 적합도를 형성하였으며, 높은 적합도를 갖는 결과를 얻었다.

이는 ACO가 생산한 DNA 코드로 (0,1)-배낭 문제의 조건인 용량에 대하여 정확히 만족하고 있으며, 이득에 있어서도 최대로 발생시킬 수 있다는 것이다. 따라서 ACO는 GA의 문제점인 전역탐색이나 지역탐색의 한계점을 극복하고, 빠른 시간내에 조건에 만족하는 우수한 해(품목)을 찾을 수 있었다.

향후 연구 과제로는 더 많은 품목을 갖는 배낭 문제를 이용하여 NCBI에 등록된 실제 DNA를 이용한 DNA 서열 디자인에 대한 연구가 필요할 것이다. 그리고 다른 인공지능 문제인 조합 최적화 문제에 적용했을 때 동일한 결과를 발생시킬 수 있는지의 분석과 실험이 필요하다.

참 고 문 헌

[1] Z. Michalewicz, Genetic Algorithms + Data Structures=Evolution Programs, Springer- Verlag, 3rd, revised and extended edition, 1999.
[2] L. M., Adleman, "Molecular computation of solutions to combinatorial problems", Science, vol. 266 pp. 1021-1024, 1994.
[3] 진강규, "유전알고리즘과 그 응용", 교우사, pp. 282-289, 2000.
[4] S. Khuri, T. Bäck, J. Heitkötter, "The Zero/one Multiple Knapsack Problem and Genetic Algorithms", Proceedings '94 ACM Symp. on Applied Computing, Phoenix, AZ, 1994.
[5] http://pl.changwon.ac.kr/algorithm/99/algorithm_sche.html
[6] http://www.biggood.com
[7] Gilles Brassard, Paul Bratley, "Fundamentals of algorithmics", 그린, pp. 239~584, 1998.
[8] P. Wasiewicz, T. Janczak, J. J. Mulawka, A. Pluciennicza, : "The Inference vai DNA Computing", In [CEC99], pp. 988-993, 1999.
[9] L. Kari. DNA computing : the arrival of biological mathematics. The mathematical Intelligencer, vol. 19, no. 2, 1997.
[10] G. H. Gonnet, C. Korostensky, S. A. Benner, "Evaluation Measures of Multiple Sequence Alignments", Journal of Computational Biology, vol. 7, no. 1-2, pp. 261-276, 2000.
[11] R. Deaton, S. A. Karl, "Introduction to DNA Computing", 1999 Genetic and Evolutionary Computation Conference Tutorial Program, pp. 75-93, Orlando, Florida, July 14, 1999.
[12] T. Yoshikawa, T. Furuhashi, Y. Uchidawa, "Acquisition of Fuzzy Rules of Constructing Intelligent Systems using Genetic Algorithm based on DNA Coding Method" Proceedings of International Joint Conference of CFSA/IFIS/SOFT'95 on Fuzzy Theory and Applications.

저 자 소 개

김은경(Eun-Gyeong Kim)

2001년 : 공주대학교 화학과 졸업(학사)
2003년 : 공주대학교 대학원 컴퓨터공학과 (공학석사)
2003년~현재 : 공주대학교 대학원 컴퓨터공학과 박사 과정

관심 분야: 바이오인포메틱스, 인공생명, DNA 컴퓨팅, 유전자알고리즘 등
E-mail : rotnrwk@kongju.ac.kr

이상용(Sang-Yong Lee)

1984년 : 중앙대학교 전자계산학과 (공학사)
1988년 : 일본동경대학대학원 총합이공학 연구과(공학석사)
1988년~1989년 : 일본 NEC 중앙연구소 연구원
1993년 : 중앙대학교 일반대학원 전자계산학과 (공학박사)
1993년~현재 : 공주대학교 정보통신공학부 교수
1996년~1997년 : University of Central Florida 방문교수

관심 분야: 인공지능, 에이전트, 컴퓨터게임, 바이오인포매틱스
E-mail : sylee@kongju.ac.kr