ε-다중목적함수 진화 알고리즘을 이용한 DNA 서열 디자인

(DNA Sequence Design using ε-Multiobjective Evolutionary Algorithm)

신 수 용 † 이 인 희 † 장 병 탁 ††
(Soo-Yong Shin) (In-Hee Lee) (Byoung-Tak Zhang)

요 약 최근 들어 DNA 컴퓨팅이 활발하게 연구되면서, DNA 컴퓨팅에서 가장 기본적이고도 중요한 DNA 서열 디자인 문제가 부각되고 있다. 기존의 연구에서 DNA 서열 디자인 문제를 다중목적 최적화 문제로 정의하고, elitist non-dominated sorting genetic algorithm(NSGA-II)를 이용하여 성공적으로 DNA 서열을 디자인하였다. 그런데, NSGA-II는 계산속도가 느리다는 단점이 있어서, 이를 극복하기 위해 본 논문에서는 ε-다중목적함수 진화알고리즘(ε-Multiobjective evolutionary algorithm, ε-MOEA)을 DNA 서열 디자인에 이용하였다. 우선, 두 알고리즘의 성능을 보다 자세히 비교하기 위해서 DTLZ2 벤치마크 문제에 대해서 적용한 결과, 목적함수의 개수가 작은 경우에는 큰 차이가 없으나, 목적함수의 개수가 많을 경우에는 ε-MOEA가 NSGA-II에 대해서 최적해를 찾는 정도(convergence)와 다양한 해를 찾는 정도 (diversity)에 있어서 각각 70%, 73% 향상된 성능을 보여주었고, 또한 최적해를 찾는 속도도 비약적으로 개선되었다. 이러한 결과를 바탕으로 기존의 DNA 서열 디자인 방법론으로 디자인된 DNA 서열들과 7-순환외판원 문제 해결에 필요한 DNA 서열을 NSGA-II와 ε-MOEA로 재디자인하였다. 대부분의 경우 ε-MOEA가 우수한 결과를 보였고, 특히 7-순환외판원 문제에 대해서 NSGA-II와 비교하여 convergence와 diversity의 측면에서 유사한 결과를 2배 이상 빨리 발견하였고, 동일한 계산 시간을 이용해서는 22% 정도 보다 다양하게 해를 발견하였으며, 92% 우수한 최적해를 발견하는 것을 확인하였다.

키워드 : DNA 서열 디자인, ε-다중목적 진화연산, NSGA-II, 다중목적 진화연산

Abstract Recently, since DNA computing has been widely studied for various applications, DNA sequence design which is the most basic and important step for DNA computing has been highlighted. In previous works, DNA sequence design has been formulated as a multi-objective optimization task, and solved by elitist non-dominated sorting genetic algorithm (NSGA-II). However, NSGA-II needed lots of computational time. Therefore, we use an ε-multiobjective evolutionary algorithm (ε-MOEA) to overcome the drawbacks of NSGA-II in this paper. To compare the performance of two algorithms in detail, we apply both algorithms to the DTLZ2 benchmark function. ε-MOEA outperformed NSGA-II in both convergence and diversity, 70% and 73% respectively. Especially, ε-MOEA finds optimal solutions using small computational time. Based on these results, we redesign the DNA sequences generated by the previous DNA sequence design tools and the DNA sequences for the 7-travelling salesman problem (TSP). The experimental results show that ε-MOEA outperforms the most cases. Especially, for 7-TSP, ε-MOEA achieves the comparative results two times faster while finding 22% improved diversity and 92% improved convergence in final solutions using the same time.

Key words : DNA sequence design, ε-MOEA, NSGA-II, MOEA

· 본 연구는 교육인적자원부 BK21-IT, 산업자원부 차세대 신기술 개발 사업의 분자 진화 컴퓨팅(MEC) 과제 및 과학기술부 국가지정연구실(NRL) 사업에 의하여 일부 지원되었다. 또한 이 연구를 위해 장비를 지원하고 공간을 제공한 서울대학교 컴퓨터연구소에도 감사 드린다.

† 학생회원 : 서울대학교 컴퓨터공학부
syshin@bi.snu.ac.kr
ihlee@bi.snu.ac.kr
†† 종신회원 : 서울대학교 컴퓨터공학부
btzhang@bi.snu.ac.kr

논문접수 : 2004년 9월 16일
심사완료 : 2005년 10월 14일

1. 서 론

최근 DNA 컴퓨팅이 새로운 컴퓨팅 모델로 부각을 받으면서, 다양한 DNA 컴퓨팅 연구 결과들이 나오고 있다 [1,2]. DNA 컴퓨팅은 DNA 분자와 같은 생체 분자를 정보 저장 및 연산의 매체로 사용하고, 생화학 실험 방법들을 연산자로 사용하여 기존의 컴퓨터로 해결이 불가능

한 NP 문제를 해결하거나, 생체 분자를 사용한다는 특징을 이용하여 의료 진단 등의 응용에 적용되고 있는 새로운 계산 모델이다[3]. 기존의 컴퓨터가 0과 1의 이진수에 기반을 둔 것에 반해서 DNA 컴퓨팅은 A(Adenine), T(Thymine), G(Guanine), C(Cytosine)의 네 개의 염기로 정보를 표현한다. 약 1 그램의 DNA는 10^{21}개의 DNA 염기를 가지며 따라서 10억 terabits의 정보저장 능력을 지닌다. 또한 1 mole의 DNA 수용액에는 아보가드로수만큼의 즉 6×10^{23}개의 분자를 가지고 있으며 이들은 용액 상에서의 화학 반응에 의해 초병렬적 정보처리가 가능하다. DNA 컴퓨팅은 이러한 많은 수의 초미세구조의 연산 소자가 초고집적도로 모여서 정보를 저장하고 초병렬적으로 처리함으로써 기존의 실리콘 기술로서 불가능한 정보처리 능력을 발휘할 수 있는 잠재력을 지니고 있는 것으로 평가되고 있다.

이런 장점들을 바탕으로 다양한 분야에 DNA 컴퓨팅이 응용되면서 연구가 활발히 진행되기 시작하였고, 이로 인해 DNA 컴퓨팅의 가장 기본적인 단계인 DNA 서열 디자인 문제가 부각되기 시작하고 있다. DNA 컴퓨팅의 연산 과정인 DNA 반응은 생화학적 반응이기 때문에 항상 오류의 가능성을 내재하고 있다. 따라서 DNA 서열 디자인을 통해서 정보를 효율적으로 표현하면서, 연산 과정에서 발생할 수 있는 오류의 가능성들을 최소화한 DNA 서열을 만드는 것이 중요한 이슈이다[4]. 기존의 연구를 통해 DNA 서열 디자인 문제는 다중목적 최적화 문제(multi-objective optimization problem, MOP)에 속하는 것으로 밝혀졌고, 특히 다양한 목적함수를 가지고 있으며, 문제에 따라 사용되어야 할 목적함수가 다르기 때문에 다수의 목적함수를 유연하게 처리할 수 있는 MOEA가 적용되기 적합한 문제라는 것을 발견할 수 있었다. 따라서 DNA 서열 설계 문제를 MOP로 수식화하였고, elitist non-dominated sorting genetic algorithm(NSGA-II)을 이용해서 성공적으로 최적화된 DNA 서열을 디자인할 수 있었다[4]. 그러나, NSGA-II는 우수한 성능에도 불구하고 계산 시간이 많이 걸린다는 단점이 있어서, 본 논문에서는 기존의 연구 결과를 바탕으로 보다 최적화되고 빠른 시간에 DNA 서열을 생성하기 위해서 ε-multiobjective evolutionary algorithm(ε-MOEA)를 도입하였다[5,6]. 우선, ε-MOEA의 우수성을 검증하기 위해서 대표적인 벤치마크 문제인 DTLZ2에 대해, 기존에 사용한 NSGA-II[7]와 비교분석해 보았고, 기존의 DNA 서열 디자인 알고리즘으로 설계된 DNA 서열들을 재설계하여 비교하여 보았고, 7-순환외판원 문제에 대해서 DNA 서열을 디자인하여 성능을 비교하였다.

본 논문의 구성은 다음과 같다. 2장에서는 DNA 컴퓨팅을 위한 DNA 서열 디자인에 대해 살펴보고, 3장에서 MOEA의 기본적인 내용과 ε-MOEA에 대해서 설명을 한 후, ε-MOEA를 DNA 서열 디자인 문제에 특화시킨 내용에 대해 소개한다. 4장에서는 실험 결과를 보여주며, 마지막으로 5장에서 본 논문의 결론을 내리고자 한다.

2. DNA 컴퓨팅을 위한 DNA 서열 디자인

DNA 컴퓨팅에서 서열 디자인의 역할은 크게 2가지로 구분할 수 있다. 첫 번째는 주어진 문제의 정보를 DNA 서열로 표현하는 것이고, 두 번째는 오류의 가능성이 가장 적은 서열을 생성하는 것이다. 이 두가지의 역할이 그림 1에 설명되어 있다. 그림 1(a)는 그래프 문제에서 3개의 정점을 서로 다른 DNA 서열로 나타낸 예제를 보여주고 있는데, 이 경우 주어진 정보는 정점이라고 할 수 있고, 이 정점을 서로 다른 DNA 서열로 표시하는 것이 서열 디자인의 역할 중 하나이다. 두 번째 역할은 그림 1(b)와 (c)에 설명되어 있다. 그림 1(b)에 설명된 것처럼 각 서열들이 서로 Watson-Crick 상보결합을 형성하지 않도록 하여야 하는데, 만약 그림 1(c)처럼 잘 설계되지 않은 DNA 서열을 사용할 경우 서로 다른 정보를 표현하여야할 DNA 서열들이 상보 결합을 형성하여 의도하지 않은 결과를 생성해 내는 문제점이 있다. 따라서 DNA 컴퓨팅에서는 이러한 연산 과정상의 오류를 최소화하여 주어진 문제의 해답을 효율적으로 발견할 수 있도록 DNA 서열을 설계하는 것이 중요한 문제이다[4,8]. 일반적으로 첫 번째 역할인 주어진 정보

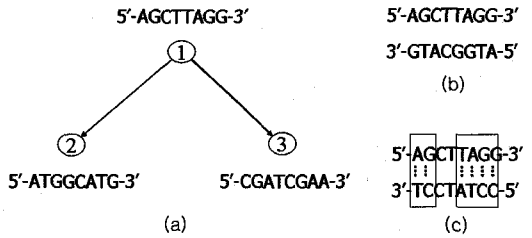

그림 1 (a) 그래프 문제에서 정점을 DNA 서열로 표시한 예. DNA 서열이 가지고 있는 방향성을 표시하기 위해 5'과 3'을 나타내었다. (b) 정점 1과 정점 2의 DNA 서열의 Watson-Crick 상보결합 여부를 보여주는 예. 상보 결합이 형성되지 않는 것을 확인할 수 있다. (c) 임의의 다른 DNA 서열 5'-CCTATCCT-3'과 정점 1의 DNA 서열을 비교한 예. 네모 칸으로 구분된 영역에서 상보결합이 형성되는 것을 확인할 수 있고, 이 경우 정확한 연산이 수행되지 않는다.

를 표현하는 것은 오류의 가능성이 배재된 DNA 서열들을 다수 생성한 후, 각 서열에 정보를 1:1 매핑하는 것으로 가능하기 때문에, 두 번째 역할인 오류의 가능성을 최소화시키는 DNA 서열을 만드는 것이 가장 어렵고도 중요한 문제라고 할 수 있다. 따라서, 본 논문에서는 오류의 가능성을 최소화시키는 DNA 서열을 만드는 것에 초점을 맞추었다.

오류의 가능성이 가장 적은 서열이라는 것은 DNA 서열들이 서로 독립적이 되도록, 즉 자기 자신 및 다른 DNA 서열들과 상보결합을 형성하지 않으며, 2차 구조도 만들어내지 않도록 디자인된 서열을 의미한다[8]. 이 문제를 해결하기 위해서 많은 연구자들이 다양한 방법으로 연구를 수행했는데[4,8], 우선 가장 간단한 방법으로 exhaustive 탐색이 사용되었고, 다양한 heuristic과 알고리즘이 사용되었다. 사용된 heuristic으로는 미리 정의된 template를 사용한 방법, 그래프 탐색 알고리즘을 이용해 DNA 서열을 디자인한 방법 등이 있고, 그 외에도 simulated annealing, dynamic programming 기법, 진화 연산 등 다양한 방법이 적용되었다(보다 자세한 내용은 [4,8]의 내용과 참고문헌 참조).

2.1 DNA 서열 디자인 기준

서로 독립적인 DNA 서열을 생성하기 위해서 다양한 기준들이 사용될 수 있는데, 본 논문에서는 H-measure, similarity, continuity, hairpin, melting temperature (Tm), GC 함량의 6가지 기준을 사용하였다. 그런데, 마지막 두 가지 기준은 최적화 대상인 목적 함수(objective)가 아니라 제약 조건(constraint)로 해석하는 것이 훨씬 문제를 효율적으로 해결한다는 사실을 발견하여 최종적으로 표 1에 있는 것처럼 4가지 목적함수($F(x)$)와 2가지 제약조건($G(x)$)을 DNA 서열 생성 기준으로 결정하였다[4]. H-measure는 다른 DNA 서열들과의 Watson-Crick 상보성을 고려하여 결합여부를 판단하는 기준이고, similarity는 다른 서열과의 유사도를 측정하는 기준이다. Continuity는 특정 염기가 연속적으로 나타나는 정도를 판단하며, hairpin은 서열이 스스로 휘어 상보결합을 통해 hairpin 구조를 형성하는 지를 계산한다. 마지막으로 Tm은 DNA 가닥의 녹는점(상보 결합된 서열의 50%가 분리되는 온도)이고, GC 함량은 DNA 서열에서 G(Guanine)와 C(Cytosine)가 차지하는 비율이다. 보다 자세한 설명은 [4]에 기술되어 있다. 이러한 기준들을 사용하여 독립적인 DNA 서열을 생성해 내는 것은 NP 문제 중 하나로 증명되어 있을 만큼 어려운 문제이다[8].

DNA 서열 디자인 문제를 소개한 기준들을 이용하여 다중목적 최적화 문제형태로 표현하면 아래와 같다.

입력 DNA 서열 집합 x에 대해서

$$optimize\ F(x) = (f_{H-measure}(x), f_{similarity}(x), f_{continuity}(x), f_{hairpin}(x))$$

$$subject\ to\ Tm_{Low} \le G_{Tm}(x) \le Tm_{High},$$

$$GC_{Low} \le G_{GC}(x) \le GC_{High}$$

여기서, Tm_{Low}와 Tm_{High}는 Tm의 최소, 최대 제약범위이고, GC_{Low}와 GC_{High}는 GC 함량의 최소, 최대 제약범위이다.

3. DNA 서열 디자인을 위한 ε-Multiobjective Evolutionary Algorithm

ε-multiobjective evolutionary algorithm(ε-MOEA)에 대해서 자세히 설명하기 이전에 multi-objective evolutionary algorithm(MOEA)에 대해서 우선 간단히 설명을 하고, ε-MOEA에 대해서 기술하고자 한다. 그 이후 ε-MOEA를 주어진 문제인 DNA 서열 디자인에 적용하기 위해 수정한 내용에 대해서 설명한다.

3.1 다중목적 진화연산

다중목적함수 진화연산(multiobjective evolutionary algorithm, MOEA)은 진화연산의 유연성과 다수의 목적함수를 처리할 수 있는 능력, 복잡한 탐색 공간을 비교적 쉽게 탐색할 수 있는 능력에 기반을 두어, 다수의 목적함수가 서로 trade-off 관계를 가지는 다중목적 최적화 문제(multiobjective optimization problem, MOP)를 풀기 위한 새로운 대안으로 부각되고 있다. MOEA는 기존의 방법과는 달리 목적함수를 통합하지 않고, 각

표 1 DNA 서열 디자인 기준

Objectives	
H-measure : $f_{H-measure}(x)$	두 서열간의 의도하지 않은 hybridization 정도
Similarity: $f_{similarity}(x)$	두 서열간의 유사도
Continuity: $f_{continuity}(x)$	특정 염기가 연속적으로 나타나는 정도
Hairpin: $f_{hairpin}(x)$	2차 구조를 생성할 가능성
Constraints	
Melting temperature: $G_{Tm}(x)$	선택된 서열의 녹는점
GC content: $G_{GC}(x)$	선택된 서열에서의 G와 C의 함량

각의 해들 사이의 dominance 관계를 이용하여 후보해들을 비교하여 진화시키는데, 해들 사이의 dominance 관계를 알아볼 때 해들이 가지는 목적함수 값들을 별도의 변환 없이 바로 이용하기 때문에, 가중치 합을 사용할 때와 같이 목적함수 공간을 왜곡해서 특정한 해를 찾을 수 없게 된다거나 하는 단점이 없고, 실제 적용에 필요한 파라미터 수도 줄일 수 있다는 장점이 있다[9].

두 해 사이의 dominance 관계는 다음과 같이 정의된다[9]. M개의 목적함수 $f_1, f_2, \cdots, f_M$을 최대화하는 MOP에서, 두 해 x와 y 사이에 다음과 같은 식이 만족될 경우, 'x는 y를 dominate한다.'

$$\forall i \in \{1,\cdots,M\},\ f_i(x) \geq f_i(y)$$
$$\exists i \in \{1,\cdots,M\},\ f_i(x) > f_i(y)$$

즉, 두 해 x, y 중에서 x가 y를 dominate하기 위해서는 x가 y보다 모든 목적함수에 대해서 나쁘지 않고, 최소한 1개 이상의 목적함수에 대해서 더 좋은 값을 가져야 한다. 따라서 x가 y를 dominate할 경우, x가 y보다 더 좋은 해라는 의미가 된다(그림 2(a) 참고). 만약 두 해 사이에 dominance 관계가 성립하지 않는다면, 두 해 사이의 우열이 정의되지 않으므로 두 해는 동등하게 취급하게 되고 non-dominated 해들이라고 한다. 또한, 어떤 해가 주어진 population 내의 해들뿐만 아니라 해 공간 내의 가능한 모든 해에 대해서 dominate되지 않을 경우, 이러한 해들을 'Pareto-최적해'라고 정의한다[9].

그런데, MOP에서 목적함수들 사이에는 trade-off 관계가 존재하므로 모든 목적함수를 동시에 최적화시키는

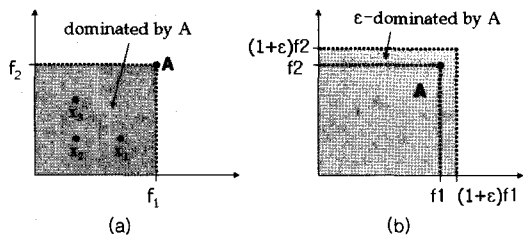

그림 2 f_1과 f_2는 모두 최대화 함수이다. (a) dominance 관계. x_1과 x_2를 볼 때, f_2에 대해서는 같은 값을 가지지만, f_1에 대해서 x_1이 x_2보다 좋기 때문에 x_1은 x_2를 dominate한다. 그러나 x_1과 x_3에서는 서로 dominate되는 관계를 결정할 수 없다. 그리고 회색 영역은 A에 의해 dominate되는 영역을 의미한다. (b) ε-dominance. 회색으로 된 영역은 A에 의해 ε-dominate되는 영역을 의미한다. (a)와 비교해 보면, f_1과 f_2에 대해서 ε만큼 dominate되는 영역이 커진 것을 알 수 있다.

하나의 Pareto-최적해가 존재하는 것은 불가능하다. 그 대신 서로 non-dominated되면서 해 공간의 모든 다른 해들을 dominate하는 Pareto-최적 해집합이 존재한다. 그러므로 MOEA의 실행결과는 대개 이러한 Pareto-최적해의 집합에 가까워지도록 진화시킨 population 내에서의 non-dominated 해집합이 되며, MOEA의 목표는 Pareto-최적 해집합에 최대한 근접하면서도(convergence) 다양한 해를 포함한(diversity) non-dominated 해집합을 찾는 것이다.

3.2 ε-다중목적 진화연산

앞에서 설명한 것처럼 MOEA는 convergence와 diversity를 통해 최대한 다양한 Pareto-최적 해집합을 찾고자 노력을 하지만, convergence와 diversity를 동시에 만족시키는 알고리즘을 만들기가 힘들다는 문제점이 있어 주로 어느 한쪽에 특화된 알고리즘이 개발되어 왔다[9]. 이러한 한계를 극복하기 위해 최근 ε-MOEA가 제안되었는데, ε-MOEA는 고정된 크기의 non-dominated 해들을 보관하고 있으면서 ε-dominance의 개념을 이용해 진화를 시켜나가는 새로운 MOEA 기법으로 steady-state GA에 기반을 두어 convergence와 diversity 양쪽을 모두 만족시켜주는 새로운 알고리즘이다[4,5].

ε-MOEA를 설명하기 전에 가장 중요한 ε-dominance의 개념부터 설명을 하고자 한다. ε-dominance 관계는 비슷한 해들을 묶기 위해 도입된 개념으로 하나의 해가 다른 해를 ε-dominate하기 위해서는 모든 목적함수에 대해 ε이상 크거나 같아야 한다(그림 2(b) 참고). 즉, M개의 목적함수를 최대화시키는 경우, x가 y를 ε-dominate하기 위해서는 다음과 같은 관계식을 만족시켜야 한다.

$$\forall i \in \{1,2,\cdots,M\}, f_i(x) + \epsilon_i \geq f_i(y)$$

ε-MOEA는 ε-dominance 관계를 이용하기 때문에 전체 해 공간을 ε의 크기를 가지는 격자 공간(grid)으로 나누어 탐색하게 되는데, 이러한 탐색 방식으로 인하여 diversity와 convergence의 측면에서 성능을 동시에 향상시킬 수 있게 되었다. 우선, 해를 탐색할 때 격자로 나누어진 탐색 공간에서 하나의 격자 안에서는 하나의 대표해만을 보존하기 때문에 보존된 해들 사이의 최소 거리가 항상 유지되어 diversity가 보장되게 된다. 또한 전체 알고리즘을 통하여 격자 단위의 탐색을 수행함과 동시에 하나의 격자 내부에서도 더 좋은 대표해만을 보존하는 방식의 세밀한 탐색도 이루어지기 때문에 convergence 측면에서도 우수한 성능을 보인다. 또한 population을 2개로 구분하여 archive라는 elite 집단을 별도로 유지하는데, 이를 통하여 탐색 도중 생성된 해들 중에서의 non-dominated 해집합을 유지할 수 있으므로,

단순히 한 세대의 population 내에서의 non-dominated 해집합만을 유지하는 것보다 더 나은 convergence 성능을 보일 수 있다.

그림 3에 ε-MOEA의 흐름이 pseudo 코드 형태로 설명되어 있다. 먼저 일반 population에서 domination 관계를 사용한 토너먼트를 이용하여 하나의 부모를 선택하고, archive에서 임의로 또 하나의 부모를 선택한 다음, 교차 및 돌연변이 연산을 통하여 새로운 해를 생성해 낸다. 그 후, archive 내의 해 중에서 새로운 해에 의해 ε-dominate되는 것이 있거나, archive의 어떤 해도 새로운 해를 ε-dominate하지 못하면 새로운 해에 의해 ε-dominate되는 해들을 모두 archive에서 제거하고, 새로운 해를 넣는다. 또한, population의 해 중에서 새로운 해에 의해 dominate되는 해가 있으면, 이 중에서 하나를 새로운 해로 바꾸고 다음 세대로 넘어가게 된다[5].

3.3 DNA 서열 디자인을 위한 알고리즘 수정

앞에서 설명한 것처럼 ε-MOEA는 기존의 다른

1. 초기 해를 무작위로 생성한 후, 함수값을 계산.
2. domination 관계에 따라 sorting하고 다른 front를 모두 dominate하는 front를 archive로 삼는다.
3. 현재 population과 archive에서 각각 부모를 선택하여 자손 1개를 생성
   3-1. population에서 2개의 개체를 선택
   3-2. 3-1에서 선택한 개체 사이에 domination 관계가 성립하면 dominate하는 개체를 선택, 성립하지 않으면 임의로 하나를 선택
   3-3. archive에서 임의로 하나를 선택
   3-4. 3-2와 3-3에서 선택된 부모로부터 유전 연산자에 따라 새로운 자손 1개를 생성한 후 함수값 계산
4. archive의 각 개체들과 새로운 자손을 비교하여 archive를 갱신.
   4-1. 새로운 자손이 ε-dominate하는 archive 개체가 있으면, 그 개체를 제거하고, 새로운 자손을 archive에 추가
   4-2. 같은 격자 공간에 속하는 archive 개체가 있으면, 둘 중에 dominate하는 쪽만 archive에 남김
   4-3. 위의 두 가지 경우에 해당하지 않고, 새로운 자손을 dominate하는 archive 개체도 없다면, 새로운 자손을 archive에 추가
5. population의 각 개체들과 새로운 자손을 비교하여 population 갱신
   5-1. 새로운 자손이 dominate하는 개체가 있으면, 그 개체를 대신하여 새로운 자손을 population에 추가 후 6으로 진행
   5-2. 새로운 자손을 dominate하는 population이 있으면, 새로운 자손을 버림
   5-3. 위의 두 가지 경우에 해당하지 않으면, population 개체 중에서 임의로 선택된 개체 대신에 새로운 자손을 population에 추가
6. 종료조건 검사 후 만족하지 않으면 3으로 되돌아감.

그림 3 ε-MOEA의 pseudo 코드

MOEA에 비해 우수한 특성을 가지고 있었고, ε-dominance 성질을 이용하여 계산 속도가 빠르다는 장점이 있다[4,5]. 따라서 기존에 사용한 NSGA-II의 느린 계산 시간을 극복하고 부가적으로 convergence와 diversity에서 우수한 해를 찾을 수 있을 것으로 예상되어 ε-MOEA를 DNA 서열 디자인에 적용하였다. ε-MOEA를 DNA 서열 디자인 문제에 이용하기 위해 몇 가지 수정을 하였는데, 우선, DNA 서열을 나타내는 개체의 표현을 위해 계층 구조를 사용하여 individual level과 sequence level의 2단계를 사용하였다. Sequence level은 각 DNA 서열을 나타내고, individual level은 DNA 서열들의 집합을 의미한다. 따라서 교차 연산자와 돌연변이 연산자는 2단계를 거쳐서 진행이 된다. 자세한 과정은 그림 4에 설명되어 있다. 또한, 선택 연산자도 일반적인 방법이 아닌 constrained tournament 선택 연산자를 사용하였다. 2.1절에서 설명한 바와 같이 DNA 서열 디자인 문제를 제약조건이 있는 다중목적 최적화 문제로 정의하였기 때문에, 제약조건을 고려하기 위해서 선택 연산자를 수정하였다. Constrained tournament 선택 연산자의 과정은 다음과 같다.

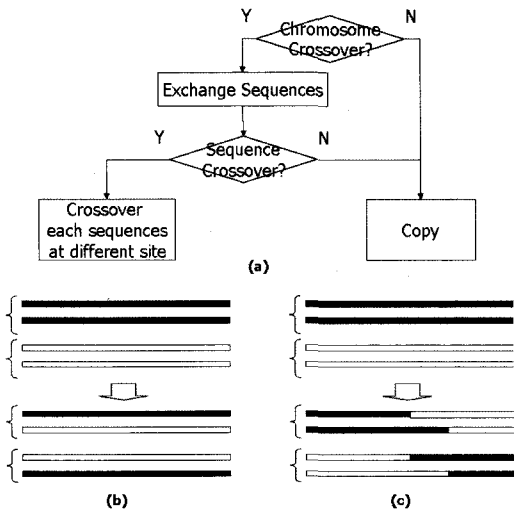

그림 4 2단계 교차 연산자의 예. 돌연변이 연산자도 동일한 과정을 거친다. (a) 교차연산자 적용과정. (b) individual level crossover. (c) sequence level crossover

- Infeasible-feasible : feasible한 개체가 선택
- Infeasible-infeasible : 페널티가 적은 개체가 선택
- Feasible-feasible : 하나의 개체가 다른 개체를 dominate하면 dominate하는 개체가 선택되고, 그렇지 않으면, 두 개의 개체 중에서 하나를 임의로 선택

그리고, 토너먼트의 사이즈는 2로 하였고, 각 제약조건 페널티의 합을 개체의 페널티로 사용하였다.

4. 실험 및 결과

4.1 DTLZ2 함수에 대한 성능 비교

ε-MOEA의 우수성을 검증하기 위해서 기존에 사용하였던 NSGA-II와 비교 분석해 보았다. DNA 서열 디자인 문제에 대해서 직접적으로 비교하기 이전에 두 알고리즘 간의 우열을 보다 명확히 확인하기 위해서 널리 사용되고 있는 benchmark 문제인 DTLZ2[10]에 적용하여 보았다. 지금까지 NSGA-II와 ε-MOEA를 직접적으로 비교 분석하여 그 성능을 검증한 논문이 발표된 적이 없기 때문에 두 알고리즘의 우열비교가 필요하다고 생각되어 실험해 보았다.

DTLZ2 목적함수의 개수를 3, 6, 12개로 변화시켜 문제 복잡도를 증가시키면서 실험하였다. DTLZ2는 목적함수의 개수를 변화시켜가면서 실험을 하기가 쉽고, 다른 MOEA 비교 실험 논문에서도 많이 사용된 test 함수이기 때문에 본 논문에서의 비교분석에 사용하였다. 실험에 사용한 DTLZ2 함수는 다음과 같다.

$$\text{Minimize } f_1(\vec{x}) = (1+g(\vec{x_M}))\cos(x_1\frac{\pi}{2})\cos(x_2\frac{\pi}{2}) \cdots \cos(x_{M-2}\frac{\pi}{2})\cos(x_{M-1}\frac{\pi}{2})$$

$$\text{Minimize } f_2(\vec{x}) = (1+g(\vec{x_M}))\cos(x_1\frac{\pi}{2})\cos(x_2\frac{\pi}{2}) \cdots \cos(x_{M-2}\frac{\pi}{2})\sin(x_{M-1}\frac{\pi}{2})$$

$$\text{Minimize } f_3(\vec{x}) = (1+g(\vec{x_M}))\cos(x_1\frac{\pi}{2})\cos(x_2\frac{\pi}{2}) \cdots \sin(x_{M-2}\frac{\pi}{2})$$

$$\cdots$$

$$\text{Minimize } f_{M-1}(\vec{x}) = (1+g(\vec{x_M}))\cos(x_1\frac{\pi}{2})\sin(x_2\frac{\pi}{2})$$

$$\text{Minimize } f_M(\vec{x}) = (1+g(\vec{x_M}))\sin(x_1\frac{\pi}{2})$$

$$0 \le x_i \le 1, \text{ for } i=1,2,\cdots,n,$$

$\vec{x}=(x_1,x_2,\cdots,x_n)$, $\vec{x_M}=(x_M,x_{M+1},\cdots,x_n)$, for $M \le n$

여기서, $g(\vec{x_M}) = \sum_{x_i \in \vec{x_M}} (x_i - 0.5)^2$

위의 식을 보면 $\sum f_i^2(\vec{x}) = (1+g(\vec{x_M}))^2$, $g(\vec{x_M}) \ge 0$임을 할 수 있다.

교차 연산자와 돌연변이 연산자는 실수 함수 문제에 적합하게 사용하기 위해서 simulated binary crossover [11], polynomial mutation[12]을 사용하였다. Simulated binary crossover는 실수 벡터에 대해서도 이진 문자열(binary string)에 대한 1점 교차 연산과 같은 효과를 얻을 수 있도록 고안된 교차 연산자이다[11]. 또한, polynomial mutation은 이 연산으로 변화되는 정도의 확률이 polynomial 분포를 따르도록 고안된 돌연변이 연산자로서 부모와 가까운 자손일수록 생성될 확률이 높다[12]. 이 두 연산자에 의한 변화 정도는 각각의 변수를 통하여 조절할 수 있다.

ε-MOEA와 NSGA-II 모두 population의 크기는 100이며, *function evaluation*의 횟수를 맞추기 위해서 NSGA-II인 경우 1,000 세대를 사용하였으며, ε-MOEA는 100,000 세대 동안 진화시켰다. NSGA-II에서 convergence와 diversity를 조절하는 controlled elitism의 파라미터로 0에서 1사이의 값을 가지는 reduction rate가 있는데, 이 값이 작을수록 elite를 더 중시하여 선택하게 된다. Reduction rate 값으로 양 극단값인 0, 1은 제외하고 0.1과 0.9를 최소/최대값으로 사용하였고, 그 중간값으로 Deb 논문에서 추천했던 0.6으로 결정하였다[9]. ε-MOEA의 경우도 ε값을 0.1, 0.5, 0.9의 3가지로 변화시켜 가며 다양한 조건에 대해 분석했는데, archive의 크기를 적당히 유지할 수 있도록 0.1과 0.9를 사용하였고, 중간값으로 0.5를 사용하였다. 3장에서 설명한 것과 같이 기존의 MOEA와 ε-MOEA의 가장 큰 차이점은 ε-MOEA는 탐색 공간을 ε크기의 격자로 구분한다는 것이다. 따라서 ε의 크기에 따라 알고리즘의 성능에 차이가 나게 되는데, 이 영향을 보다 확실히 분석하기 위해서 NSGA-II의 reduction rate 변화에 따른 결과값과 비교해 보았다. NSGA-II의 결과는 reduction rate에 따라 front의 크기와 개수가 결정되기 때문에 공정한 비교가 될 것으로 생각된다. 일반적으로 사용되는 파라미터 값들에 따라 교차연산 비율은 0.9이고, 돌연변이연산 비율은 0.01로 결정하였다. 이 연산자들도 각각 하나씩의 파라미터를 포함하고 있는데, 파라미터들의 값도 변화시켜가면서 다양한 경우에 대해서 결과를 비교하였다. 교차연산자는 $\eta_c \in \{1,5,10,20,100\}$의 값들을 사용하였고, 돌연변이연산자는 $\eta_m \in \{0,10,50,100,500\}$의 값들을 이용하였다[11,12].

실험 결과의 convergence와 diversity 정도를 알아보기 위해, 각각 generational distance(GD)[13]와 maximum spread[14]을 사용하여 비교하였다. Generational distance는 알고리즘에서 찾아낸 non-dominated 해집합 Q와 실제 Pareto-최적 해집합 P^* 사이의 평균 거리를 이용하여 Q가 converge한 정도를 측정하는 방법으로, $GD = \frac{(\sum_{i=1}^{|Q|} d_i^p)^{\frac{1}{p}}}{|Q|}$로 정의된다. 여기서 d_i는 Q의 i번

째 해에서 P^*의 각 해까지의 탐색 공간상에서의 유클리드 거리 중 제일 작은 값을 의미한다. 따라서 generational distance는 값이 작을수록 실제 Pareto-최적 해집합에 가깝게 발견하였다는 것을 나타낸다. 그리고 maximum spread는 알고리즘에서 찾아낸 non-dominated 해집합 Q를 둘러쌀 수 있는 가장 작은 hyper box의 대각선 값을 이용하여 Q의 분포 정도를 측정하는 방법으로 $D=\sqrt{\sum_{m=1}^{M}(\max_Q(f_m)-\min_Q(f_m))^2}$로 정의되고, 여기서 $\max_Q(f_m)$와 $\min_Q(f_m)$는 각각 Q에서 m번째 목적함수의 최대값과 최소값을 의미한다. 따라서 maximum spread는 반대로 값이 클수록 다양한 해집합을 발견했다고 할 수 있다.

자세히 수치적으로 비교한 결과가 표 2에 설명되어 있다. 우선, maximum spread 값이 사용된 문제의 목적함수의 개수나 목적함수의 값의 범위에 따라 달라지기 때문에, 각각의 경우에 대한 spread의 이론적인 최대값으로 나누어(spread/ideal 값) 0과 1사이로 표준화하였다. 따라서 표 2에서 spread/ideal 값이 1에 가까울수록 이상적인 값에 가까운 것이라고 할 수 있다. 표 2에서 볼 때 NSGA-II는 목적함수의 개수가 작을 때는 1에 가까운 값을 보이나 목적함수의 개수가 많아지면 diversity가 좋지 않은 것을 알 수 있다. 그러나 ε-MOEA는 목적함수의 개수가 많아지더라도 1에 가까운 값을 유지하여 전체적인 해답의 분포를 일정하게 유지하는 것을 발견할 수 있다. 보다 정확한 비교를 위해서, NSGA-II와 ε-MOEA의 spread/ideal 값을 각각 A, B라고 할 때, $\frac{|A-1|-|B-1|}{|A-1|}$의 값을 계산하여 NSGA-II보다 ε-MOEA이 얼마나 큰 성능향상을 보이는지 보였다. 표 2에 설명된 것처럼 얻어진 최적해의 분포를 보면 ε-MOEA의 결과가 세 경우에 대해 평균적으로 73% 정도 우수한 것을 알 수 있다.

또한 Pareto-최적 해집합에 대한 converge 정도를 GD값으로 비교한 결과도 표 2에 설명되어 있다. 보다 정확한 비교를 위해서 NSGA-II에서의 GD값과 ε-MOEA에서의 GD값의 차이를 NSGA-II에서의 GD값으로 나누어 NSGA-II에서보다 ε-MOEA에서 얼마만큼의 성능 향상이 있는지를 보였다. 역시 문제가 쉬운 경우에는 NSGA-II의 결과가 좋으나, 문제가 조금만 복잡해지더라도 ε-MOEA의 결과가 우수한 것을 재확인할 수 있다. 목적함수가 3개인 경우 NSGA-II가 2배 정도 좋은 성능을 보였으나, 소수점 이하 넷째자리 값 비교이기 때문에 실제로 큰 차이를 보이지는 않았다. 목적함수가 6개, 12개인 경우만 볼 때 ε-MOEA는 각각 95%와 83%의 성능 향상을 보였다, 목적함수 개수의 영향을 보다 명확히 확인하기 위해 목적함수의 개수를 18개로 확장시켜 본 결과, ε-MOEA와 NSGA-II의 성능 차이는 33%로 줄어들었다. Purshouse와 Fleming[15]에 의하면 목적함수의 개수가 증가함에 따라 현재의 population 내에서 dominate되지 않는 해의 비율이 급격하게 증가하게 되며, 또한 dominance resistance라 불리는 교차나 돌연변이 연산에 의해 부모를 dominate하는 해를 찾아내는 일이 점점 어려워지는 현상이 발생하게 된다고 한다. 이러한 문제점들로 인하여 ε-MOEA의 수렴정도가 느려져 NSGA-II와의 성능차이가 줄어든 것으로 생각된다. 그러나 NSGA-II가 목적함수 6개 정도에서 수렴 정도가 느려진 것에 반해서 ε-MOEA는 보다 많은 수의 목적함수에서도 NSGA-II처럼 급격히 수렴속도가 느려지지는 않았으므로 ε-MOEA가 NSGA-II보다 확장성(scalability)이 좋다고 할 수 있겠다.

마지막으로 NSGA-II와 ε-MOEA의 수렴 속도를 확인하기 위해서 두 경우 모두 파라미터 값들 중에서 converge 정도가 제일 좋은 경우를 선택하여 function evaluation 횟수에 따른 generational distance 값을 그래프로 그려보았다. 그림 5와 6이 그 결과를 보여주고 있는데, 목적함수의 개수가 적을 때는 NSGA-II의 수렴속도도 ε-MOEA와 비슷하나(그림 5), 목적함수의 개수가 늘어나자(그림 6) ε-MOEA의 수렴속도가 월등히 빠른 것을 알 수 있다.

4.2 ε-MOEA를 이용한 DNA 서열 디자인

DTLZ2 함수에 대해서 비교해 본 결과, ε-MOEA가 예상한 대로 convergence와 diversity는 물론 수렴 속도까지 빠르다는 것을 확인할 수 있었고, ε-MOEA를 주 목적인 DNA 서열 디자인 문제에 적용해 보았다.

기존의 연구에서 NSGA-II와 기존 여러 DNA 서열 디자인에 사용된 방법론(유전자 알고리즘, simulated annealing 등)을 비교하여 NSGA-II가 기존 여러 방법

표 2 DTLZ2 함수에 대한 NSGA-II와 ε-MOEA의 결과 비교. 각각 5번씩의 실험을 통해 얻어진 결과값들이다.

DTLZ2	NSGA-II			ε-MOEA			Compare	
	GD	Maximum Spread	Spread /ideal	GD	Maximum Spread	Spread /ideal	GD	Spread /ideal
3 OBJ	0.0004	1.7157	0.9906	0.0015	1.7426	1.0061	-2.75	0.3511
6 OBJ	0.0690	4.3550	1.7779	0.0037	2.4865	1.0151	0.9464	0.9806
12 OBJ	0.0902	5.1928	1.4990	0.0155	3.7293	1.07657	0.8282	0.8466

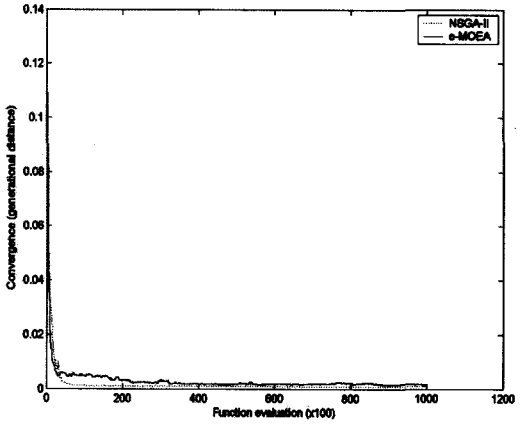

그림 5 목적함수 개수가 3개 인 경우의 function evaluation 횟수 비교

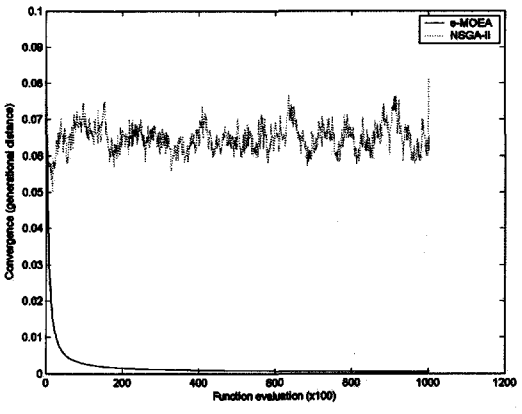

그림 6 목적함수가 12개인 경우

들 보다 모두 우수한 결과를 보여주는 것을 확인하였다 [4]. 본 논문에서는 그러한 연구 결과를 바탕으로 하여 NSGA-II와 ε-MOEA의 결과를 비교하여 보았는데, [4]의 결과 중에서 일반적인 유전자 알고리즘을 사용하여 생성된 DNA 서열 집합[16]과 simulated annealing을 이용하여 생성된 DNA 서열 집합[18], 그리고 순환 외판원 문제를 해결하기 위해 디자인된 DNA 서열[19]을 대상으로 비교하였다. NSGA-II와 ε-MOEA에 사용된 파라미터 값들은 다음과 같다. H-measure와 similarity의 하한값은 6 base와 17%로 하였고, continuity는 2를 사용하였으며, hairpin은 6개의 base가 각각 스템과 루프에 필요하다고 가정하였다. NSGA-II의 reduction rate는 0.65를 사용하였으며, ε-MOEA의 ε 값은 1을 사용하였다. [16]의 서열 집합을 위해서는 개체군 크기는 3000, 최대 세대수는 200으로 하였고, [18]의 경우 5000과 300을 사용하였으며, 교차연산 확률은 0.9, 돌연변이 연산 확률은 0.01로 결정하였다.

[16]에서 Deaton 등이 다중목적 진화연산이 아닌 일반적인 유전자 알고리즘(simple GA)를 사용하여 7개의 길이가 20인 DNA 서열을 생성하였고, [4]에서 NSGA-II가 보다 우수한 DNA 서열을 생성하는 것을 보여주었다. 그 결과를 기반으로 ε-MOEA와 비교한 실험 결과가 표 3과 4에 나타나 있다. 표 3에는 생성된 DNA 서열의 예가 제시되어 있고, 표 4에 자세히 수치적으로 비교한 결과가 나타나 있다. 표 4, 6, 8은 모두 NSGA-II와 ε-MOEA 모두 5번씩 실행을 하여 총 10개의 non-dominated 집합을 모은 후 이 중에서 다시 non-dominated 해집합을 찾아서 이들을 가상의 Pareto-최

표 3 [16]의 DNA 서열 생성 결과. Tm은 nearest neighbor 모델을 사용해 oligomer 10nM, Na^+농도 1M에서 계산하였다.

NSGA-II	Continuity	Hairpin	H-measure	Similarity	Tm	GC%
CTCTTCATCCACCTCTTCTC	0	0	43	58	61.3859	50
CTCTCATCTCTCCGTTCTTC	0	0	37	58	61.4403	50
TATCCTGTGGTGTCCTTCCT	0	0	45	57	64.4631	50
ATTCTGTTCCGTTGCGTGTC	0	0	52	56	65.8284	50
TCTCTTACGTTGGTTGGCTG	0	0	51	53	64.6346	50
GTATTCCAAGCGTCCGTGTT	0	0	55	49	65.3002	50
AAACCTCCACCAACACACCA	9	0	55	43	66.7173	50
ε-MOEA						
AGAAGAAGACGAGGAGAGGA	0	0	36	65	63.8004	50
CGGCACCATAGGAACAAGAA	0	0	48	56	64.7377	50
AAGCGAATCGGAGACAACAC	0	0	49	56	65.2805	50
AGAGGTAGGTAGAGGTTGTG	0	0	47	54	62.2294	50
GGCCGGAACCTAACATAACT	0	0	56	50	64.1893	50
GGAAGCGTGAGAAGAGAAGA	0	0	41	62	63.7698	50
TTATTGATGCGGCGTATGGC	0	0	59	45	65.8611	50

표 4 [16]의 서열에 대한 NSGA-II와 ε-MOEA의 성능 비교

	Convergence (generational distance)		Diversity (maximum spread)	
	NSGA-II	ε-MOEA	NSGA-II	ε-MOEA
평균	100.3865	79.4798	147.6034	119.2923

표 5 [18]의 DNA 서열에 대한 비교 결과. Tm은 표 3과 같이 nearest neighbor 모델을 사용해 oligomer 10nM, Na^+농도 1M에서 계산하였다.

NSGA-II	Continuity	Hairpin	H-measure	Similarity	Tm	GC%
GTGACTTGAGGTAGGTAGGA	0	3	129	115	47.249	50
ATCATACTCCGGAGACTACC	0	3	132	121	47.2304	50
CACGTCCTACTACCTTCAAC	0	0	128	121	47.4589	50
ACACGCGTGCATATAGGCAA	0	3	141	117	52.5401	50
AAGTCTGCACGGATTCCTGA	0	3	132	115	50.9497	50
AGGCCGAAGTTGACGTAAGA	0	0	132	116	51.0482	50
CGACACTTGTAGCACACCTT	0	0	132	123	50.2683	50
TGGCGCTCTACCGTTGAATT	0	0	135	116	52.0565	50
CTAGAAGGATAGGCGATACG	0	0	134	117	46.6253	50
CTTGGTGCGTTCTGTGTACA	0	0	140	116	50.5774	50
TGCCAACGGTCTCAACATGA	0	0	132	121	51.8587	50
TTATCTCCATAGCTCCAGGC	0	0	136	117	48.1017	50
TGAACGAGCATCACCAACTC	0	0	121	121	50.3351	50
CTAGATTAGCGGCCATAACC	0	0	127	119	47.6383	50
ε-MOEA						
CAGGCATCGATTACAGAGTC	0	0	126	115	62.4346	50
ATGCGGCGCTCTGAATATGT	0	0	134	115	67.0951	50
ATCCGAGTCGTTCATACTGC	0	0	136	122	64.2336	50
GCGCAAGTACCACCAACAAT	0	0	131	120	66.183	50
AACAACGATCGCCTTAACGC	0	0	130	123	66.1895	50
GTTAGCGCTTCTTGTGTCGT	0	0	135	114	65.5157	50
GAGGAACTTACCGCATTGTG	0	0	142	125	63.5023	50
AAGGCACATCACAAGGAACC	0	3	118	117	65.2864	50
GCTATGGACATAGTCGAACG	0	3	130	119	62.4815	50
AGCACAACGCTAATAGGAGG	0	0	124	118	64.221	50
GGTTCCACACGAGCATATTG	0	0	142	113	63.5746	50
GTGGAACTAGCGACCAAGAT	0	0	138	124	64.1472	50
CTGAATTGGCAACTGCTTGC	0	0	128	113	65.2675	50
AAGCCACGCGTAACTCCATA	0	0	136	120	66.3238	50

표 6 [18]의 서열에 대한 NSGA-II와 ε-MOEA의 성능 비교

	Convergence		Diversity	
	NSGA-II	ε-MOEA	NSGA-II	ε-MOEA
평균	129.1886	102.9420	439.6902	119.5

적 해집합이라고 생각하고 convergence를 측정한 결과이다. DNA 서열 디자인 문제에서는 해 공간이 방대하여 실제 Pareto-최적 해집합을 모르기 때문에 위와 같은 가정을 하였다. 표 4에서 볼 때 convergence의 측면은 ε-MOEA가 훨씬 좋지만, diversity 측면에서는 ε-MOEA가 조금 좋지 않은 것을 발견할 수 있다. 이는 NSGA-II에서 생성된 DNA 서열이 너무 하나의 목적함수(주로 h-measure)에 특화되어 있어서 ε-MOEA가 최종적으로 발견한 해집합의 diversity가 계산 기준인 maximum spread의 특성 상 NSGA-II이 발견한 해집합보다 나쁜 것처럼 보이는 것을 알 수 있다. 본 논문에 기술하지는 않았지만 DTLZ2의 경우에서도 유사한 경우가 발견되어 분석을 해 본 결과, 이는 maximum spread 기준이 가지고 있는 한계점인 것으로 판단되었고, 실제로는 ε-MOEA가 훨씬 좋은 diversity를 보이는 것을 그래프로 확인할 수 있었다[17]. 역시 이 경우

표 7 7-TSP를 위한 DNA 서열 생성 결과. Tm은 nearest neighbor 모델을 사용해 oligomer 10nM, Na+농도 1M에서 계산

NSGA-II	Continuity	Hairpin	H-measure	Similarity	Tm	GC%
AATAGGAGCAGGAGACAACG	0	0	66	41	63.8672	50
CTCTCATCTCTCCGTTCTTC	0	0	44	52	61.4403	50
TATCCTGTGGTGTCCTTCCT	0	0	58	54	64.4631	50
ATTCTGTTCCGTTGCGTGTC	0	0	55	55	65.8284	50
TCTCTTACGTTGGTTGGCTG	0	0	56	51	64.6346	50
TAGTTCCAAGCGTCCGTGTT	0	0	54	53	66.4596	50
TATCCACACCAACACACCAC	0	0	63	44	64.6161	50
ε-MOEA						
AGCAACAAGAATGCGGCAAG	0	3	57	50	66.7959	50
TACATGACCAAGGACGCCAA	0	0	55	52	66.2632	50
GTGGAAGCTTGTAAGGCGTT	0	0	63	47	65.5567	50
GAGAGAGAACGGAAGAACGA	0	0	51	55	63.4533	50
AATCACTGTTGGATCGGACG	0	0	63	51	64.7547	50
CTCCTTGTCATCATGCTCTG	0	0	66	41	62.6735	50
ACTAGAGTAGGCCGGAGATA	0	0	57	52	63.0744	50

표 8 7-TSP에서 NSGA-II와 ε-MOEA의 성능 비교

	Convergence		Diversity	
	NSGA-II	ε-MOEA	NSGA-II	ε-MOEA
평균	1.62744	0.128647	165.718	202.232

에도 표 3을 볼 때, 선택된 최종 DNA 서열 집합을 볼 때는 h-measure도 우수하며 다른 나머지 목적 함수들에 대해서도 좋은 결과를 보이는 것을 알 수 있었다. 그리고, 계산 시간을 비교하기 위해서 그림 7에 function evaluation 횟수에 따른 convergence 정도를 그려보았다. DTLZ2의 함수 때와 마찬가지로 ε-MOEA가 NSGA-II보다 훨씬 빨리 좋은 결과를 보여주며 지속적으로 점점 좋은 결과를 찾아내는 것을 재확인할 수 있었다.

표 5와 표 6에는 [18]의 DNA 서열에 대한 비교 분석

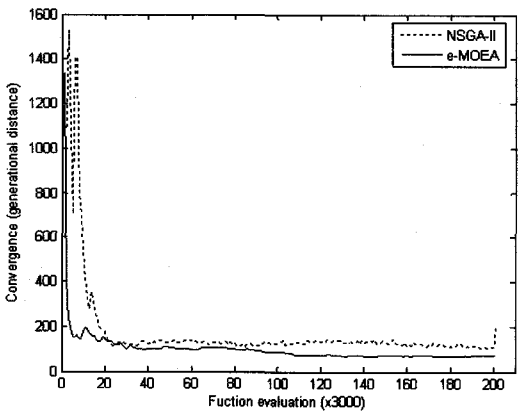

그림 7 [16]의 서열에서 function evaluation 횟수에 따른 수렴 정도 비교

결과가 설명되어 있다. Tanaka 등은 simulated annealing을 이용하여 14개의 길이 20인 DNA 서열을 디자인하였다[18]. 역시 표 5에 최종 DNA 서열이 제시되어 있고, 표 6에 비교 결과가 나타나 있다. [18]의 DNA 서열을 재디자인하는 경우에도 Deaton 등의 경우와 유사하게 convergence는 좋으나 diversity는 조금 떨어지는 현상을 재확인할 수 있었다. 역시 그림으로 보여주지는 않았으나 수렴 속도도 역시 ε-MOEA가 빠른 것을 볼 수 있었다.

그리고, 마지막으로 실제적인 문제 해결에 사용될 수 있다는 것을 보이기 위해서 [19]에서 해결한 순환 외판원 문제(TSP)를 위해 설계된 DNA 서열을 ε-MOEA를 이용하여 재설계해 보았다. [19]에서는 단순한 유전자 알고리즘을 사용하여 서열을 생성했었는데, 표 7에 새로이 생성한 DNA 서열이 소개되어 있다.

그림 8에 7-순환외판원 문제를 위한 DNA 서열을 생성하는 데 필요한 function evaluation 횟수를 그래프로 표시하여 보았는데, 그림 8에서 알 수 있는 것처럼 ε-MOEA가 절반의 시간만을 사용하고도 수렴하는 것을 알 수 있고, 계속 세대가 진행될수록 더욱 최적해에 가까운 값들을 계속 찾아나가는 것을 확인할 수 있다. 알고리즘 속도뿐만 아니라 해의 품질과 다양한 해를 찾는 정도도 ε-MOEA가 우수한 것을 확인할 수 있었는데, 표 8에 설명되어 있다. Convergence에서는 ε-MOEA가 훨씬 좋은 결과를 보여주고, diversity에서도 ε-

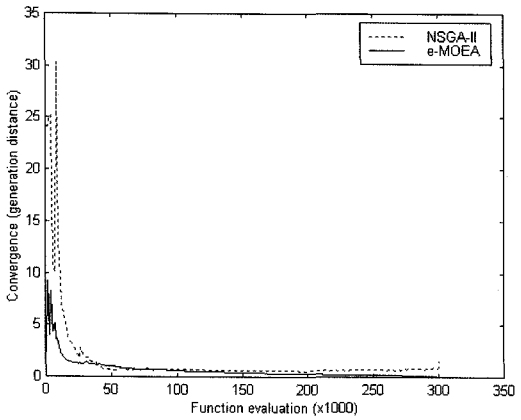

그림 8 7-순환외판원 문제 DNA 서열 생성을 위한 NSGA-II와 ε-MOEA의 function evaluation 횟수 비교

MOEA가 NSGA-II보다 우수한 결과를 보여주는 것을 확인할 수 있다. 7-순환외판원 문제 서열 디자인에 대해서 표 2를 분석한 방법과 동일하게 측정해 본 결과, convergence에서 ε-MOEA는 NSGA-II보다 $\frac{1.62744-0.128647}{1.62744}$ ≒0.921, 즉 약 92.1 %의 성능 향상을 보였고, diversity에서는 $\frac{202.232-165.718}{165.718}$ ≒0.22, 즉 22%의 정도의 성능 향상을 보이는 것을 확인할 수 있었다.

5. 결론 및 토의

본 논문에서는 기존에 DNA 서열 디자인에 사용한 NSGA-II의 단점을 해결하기 위해서 ε-MOEA를 이용해 DNA 서열을 디자인해 보았다. ε-MOEA의 우수성을 보다 자세히 NSGA-II와 비교하기 위해서 우선 DTLZ2 벤치마크 문제에 적용하여 보았다. 목적함수의 개수가 작은 경우에는 큰 차이가 없었으나, 목적함수의 개수가 많을 경우(6개 이상) ε-MOEA가 NSGA-II에 대해서 convergence와 diversity에 대해서 각각 70%, 73% 향상된 성능을 보여주었고, 계산 시간도 목적 함수의 개수가 많을 수록 비약적으로 단축시키는 것을 확인할 수 있었다, 즉, DNA 서열 디자인 문제에 대해서 ε-MOEA는 기존에 사용한 NSGA-II의 문제점인 느린 계산 시간을 극복할 수 있을 뿐 아니라, convergence와 diversity에서도 성능 향상을 보여줄 수 있는 것이 확인되어, ε-MOEA를 이용하여 DNA 서열을 디자인하여 보았다. 기존의 연구에서 NSGA-II가 다른 여러 가지 DNA 서열 디자인 방법론들(단순한 유전자 알고리즘, simulated annealing)보다 우수한 것을 확인하였는데, ε-MOEA를 사용한 결과 NSGA-II를 능가하는 것을 알 수 있었고, 특히 실제 DNA 컴퓨팅으로 해결한 7-순환외판원 문제에 사용한 DNA 서열에 대해서 NSGA-II보다 convergence와 diversity 측면에서 유사한 결과를 2배 이상 빨리 발견하였고, 동일한 계산 시간을 이용해서는 22% 정도 보다 다양하게 해를 발견하였으며, 92% 우수한 최적해를 발견하는 것을 확인하였다. 특히 DNA 서열 디자인 문제의 경우 디자인해야 하는 서열의 종류가 많거나, 서열이 길어질 경우 function evaluation 시간이 커지므로 적은 세대를 이용해 수렴할 수 있다면 그만큼 계산 시간을 절약할 수 있는 장점이 있다. 또한 같은 function evaluation 횟수를 사용하더라도 ε-MOEA의 경우 NSGA-II에서보다 알고리즘 내부에서의 dominance 관계 계산 횟수가 적기 때문에 population의 크기가 큰 경우에는 실제 계산 시간에서 큰 이득을 볼 수 있다.

이러한 결과를 바탕으로 본 연구팀에서 개발 중인 DNA 서열 디자인 프로그램인 NACST/Seq 의 알고리즘을 기존의 NSGA-II에서 ε-MOEA로 변경하였는데, 특히 ε-MOEA의 빠른 convergence 속도로 인해 DNA 서열을 디자인하는 데 소비되는 시간을 대폭 단축시킬 수 있을 것으로 기대된다.

참 고 문 헌

[1] Garzon, M. H. and Deaton, R. J., "Biomolecule Computing and Programming," IEEE Transactions on Evolutionary Computation, Vol.3, No.3, pp. 236-250, 1999.

[2] Reif, J. H., "The Emergence of the Discipline of Biomolecular Computation in the US," New Generation Computing, Vol.30, No.3, pp. 217-236, 2002.

[3] Maley, C. C., "DNA Computation: Theory, Practice, and Prospects," Evolutionary Computation, Vol.6, No.3, pp. 201-229, 1998.

[4] Shin, S.-Y., Lee, I.-H., Kim, D., and Zhang, B.-T., "Multi-Objective Evolutionary Optimization of DNA Sequences for Reliable DNA Computing," IEEE Transactions on Evolutionary Computation, Vol.9, No.2, pp. 143-158, 2005.

[5] Laumanns, M., Thiele, L., Deb, K., and Zitzler, E., "Combining Convergence and Diversity in Evolutionary Multi-Objective Optimization," Evolutionary Computation, Vol.10, No.3, pp. 263-282, 2002.

[6] Deb. K., Mohan, M., and Mishra, S., "A Fast Multi-Objective Evolutionary Algorithm for Finding Well-Spread Pareto-Optimal Solutions," KanGAL Report No. 2003002, 2003.

[7] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A Fast and elitist multiobjective genetic

algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, Vol.6, pp. 182-197, 2001.
[8] Garzon, M. and Deaton, R., "Codeword design and information encoding in DNA ensembles," Natural Computing, Vol.3, No.3, pp. 253-292, 2004.
[9] Deb, K., Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Ltd., 2001.
[10] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E., "Scalable Test Problems for Evolutionary Multi-Objective Optimization," KanGAL Report No. 200101, 2001.
[11] Deb, K. and Agrawal, R. B., "Simulated Binary Crossover for Continuous Search Space," Complex Systems, Vol.9, No.2, pp. 115-148, 1995.
[12] Deb. K. and Goyal, M., "A Combined Genetic Adaptive Search(GeneAS) for Engineering Design," Computer Science and Informatics, Vol.26, No.4, pp. 30-45, 1996.
[13] Veldhuizen, D. V., "Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations," Ph. D. Thesis, Dayton, OH:Air Force Institute of Technology, Technical Report No. AFIT/DS/ENG/99-01, 1999.
[14] Zitzler, E., "Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications," Ph. D. Thesis, Zürich, Switzerland:Swiss Federal Institute of Technology(ETH)(Dissertation ETH No. 13398), 1998.
[15] Purshouse, R. C. and Fleming, P. J., "Evolutionary Many-Objective Optimisation: An Exploratory Analysis," Proceedings of the 2003 Congress on Evolutionary Computation, pp. 2066-2073, 2003.
[16] Deaton, R., Murphy, R. C., Garzon, M., Franceschetti, D. R., and Stevens Jr., S. E., "Good encoding for DNA-based solutions to combinatorial problems." Proceedings of the Second Annual Meeting on DNA Based Computers, pp. 247-258, 1996.
[17] S.-Y. Shin, "Multi-Objective Evolutionary Optimization of DNA Sequences for Molecular Computing," Ph. D. Thesis, Seoul National University, 2005.
[18] Tanaka, F., Nakatsugawa, M., Yamamoto, M., Shiba, T., and Ohuchi, A., "Developing support system for sequence design in DNA computing," Proceddings of the 7th International Workshop on DNA Based Computers, pp. 340-349, 2001.
[19] Lee, J. Y., Shin, S.-Y., Park, T. H., and Zhang, B.-T., "Solving Traveling Salesman Problems with DNA Molecules Encoding Numerical Values," BioSystems, Vol.78, pp. 39-47, 2004.

신 수 용
1998년 2월 서울대학교 컴퓨터공학과 학사. 2000년 2월 서울대학교 컴퓨터공학부 석사. 2005년 8월 서울대학교 전기,컴퓨터공학부 박사. 관심분야는 진화연산, DNA 컴퓨팅, 생물정보학, 기계학습

이 인 희
2001년 2월 서울대학교 컴퓨터공학부 학사. 2001년 3월~현재 서울대학교 전기, 컴퓨터공학부 석박사 통합과정. 관심분야는 다중목적 진화연산, DNA 컴퓨팅, Molecular theorem proving method

장 병 탁
정보과학회논문지 : 소프트웨어 및 응용 제 32 권 제 11 호 참조