분할과 병합을 이용한 새로운 공진화 알고리즘-SMGA

박성진° , 김명원
숭실대학교 컴퓨터학과
comart@it.soongsil.ac.kr, mkim@computing.soongsil.ac.kr

SMGA: A New Coevolutionary Algorithm based on Species Splitting and Merging

Soungjin Park, Myungwon Kim
School of Computing, Soongsil University

요 약

진화 알고리즘은 현재까지 다양한 최적화 문제들에 사용되어 왔고, 또한 이러한 최적화 문제들을 효율적으로 해결하기 위하여 많은 진화 알고리즘이 개발되어 왔다. 그러나 이러한 진화 알고리즘들의 공통적인 문제점은 탐색공간의 확대에 대하여 전반적으로 탐색시간이 오래 걸린다는 것이다. 실제로 최적화 해야 할 변수의 증가에 따라 탐색 차원이 증가하므로 탐색 시간도 기하급수적으로 늘어난다. 따라서 최근의 진화 알고리즘에 대한 연구는 탐색공간의 축소나, 진화 속도의 향상에 초점이 맞추어져 있었고, 이러한 경향에 따라 많은 연구성과가 있었다. Potter와 DeJong의 협력 공진화와, Weicker의 적응적 공진화가 바로 그것이다. 그러나 이 방법들도 최적화 해야 할 변수들이 서로 강한 의존성을 가지고 있는 경우나, 대부분의 변수가 서로 의존성을 가지고 있는 경우에는 그다지 좋은 결과를 보이지 못하는 문제점을 가지고 있다. 본 논문에서는 이러한 연구들을 기반으로 하여 각 방법의 단점들을 보완함으로써 효율을 향상시킨 새로운 진화 알고리즘을 제안한다.

1. 서론

나양한 최적화 문제들을 효율적으로 풀기 위해 현재까지 많은 진화 알고리즘들이 개발되어 왔고, 이러한 진화 알고리즘들의 공통적인 문제점은 탐색공간의 확대에 대해 전반적으로 탐색시간이 오래걸린다는 것이다. 최적화 문제들은 문제의 특성상 최적화 해야 할 변수의 증가에 따라 탐색공간이 기하급수적으로 늘어나며, 그것에 비례하여 최적해의 탐색시간도 기하급수적으로 늘어난다. 따라서 최근의 진화 알고리즘에 대한 연구는 진화 속도의 향상에 초점이 맞추어져 있고, 실제로 많은 성과가 있었다. 특히, 1994년 Potter와 DeJong이 제안한 협력 공진화(cooperative coevolution)방법은 최적화 해야 할 각각의 변수에 대하여 종족(species)개념을 도입함으로써, 탐색 공간이 충분히 클 때에도 빠른 수렴속도를 보인다[1]. 그러나, 이 방법도 특성 문제—탐색해야 할 변수들 간의 의존성(variable dependency)이 강한 문제—에 대해서는 오히려 일반적인 진화 알고리즘보다도 성능이 떨어진다. 이에 대하여 1999년 Weicker가 제안한 적응적 공진화(adaptive cooperative coevolution) 알고리즘은 변수들간의 의존성이 존재하면 그 변수들을 표현하는 종족들을 병합함으로써 이 문제를 해결 하였다[2]. 본 논문에서 제안하는 진화 알고리즘은 Weicker의 알고리즘을 개선하여 종족의 병합 후에도 지속적으로 관찰하면서 필요한 경우 병합된 종족을 다시 각각의 변수를 표현하는 다른 종족으로 분할이 될 수 있도록 하였다. 즉, 탐색점이 국소 최적해를 벗어나 전역 최적해의 끌림유역(basin of attraction)에 유입되었을 경우, 다시 종족을 분할하여 빠른 진화속도를 낼 수 있도록 하였다.

본 논문은 다음과 같은 순서로 구성되어 있다. 2장에서는 관련연구들을 설명하고, 3장에서는 제안하는 알고리즘에 대하여 설명하며, 4장에서는 실험 결과를 보인다. 마지막으로 5장에서는 결론과 향후연구를 제시하였다.

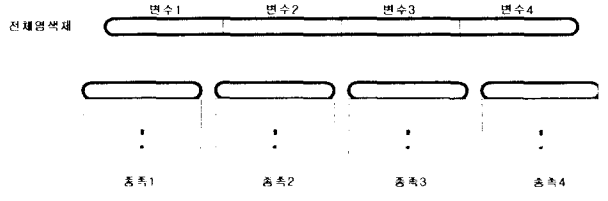
2. 관련연구

(1) 협력 공진화(Cooperative Coevolution)[1]

1994년 Potter와 DeJong이 개발한 알고리즘으로서 최적화 해야할 변수들을 각각 따로 진화 시키면서 서로의 정보를 공유하도록 하여 탐색공간이 큰 경우에도 빠른 진화가 가능하도록 하였다.

-CCGA1(Cooperative Coevolution Genetic Algorithm)

이 알고리즘은 일반적인 진화 알고리즘에서의 염색체를 각각의 변수 단위로 잘라내어 종족을 만들어 탐색공간을 줄임으로써 진화 속도의 향상을 꾀하였다<그림 1>.



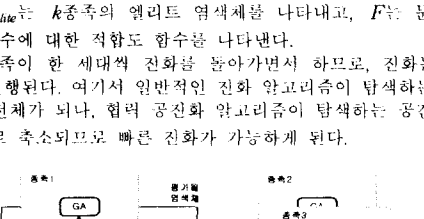
<그림 1> 협력 공진화에서의 염색체 형태

각각의 종족은 염색체 평가를 제외하고는 일반적인 진화 알고리즘이 적용된다. 염색체의 평가는 <그림 2>와 같이 다른 종족에서의 엘리트 염색체들을 모아서 현재 종족의 염색체를 평가하게 된다. 즉, 최적화 해야할 변수가 s개 일 때, k종족의 i번째 염색체의 적합도는 (식 1)과 같다.

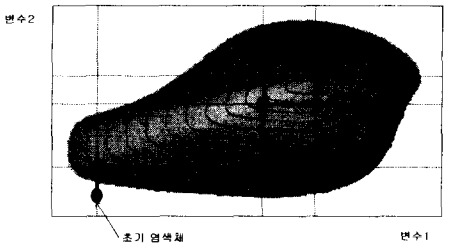
$$F^{CCGA1}(c_i^k) = F(c_{elite}^1, c_{elite}^2, \ldots, c_i^k, \ldots, c_{elite}^s) \quad (식\ 1$$

여기서 c_{elite}^k는 k종족의 엘리트 염색체를 나타내고, F는 문제 영역에서 전체 변수에 대한 적합도 함수를 나타낸다.

각각의 종족이 한 세대씩 진화를 돌아가면서 하므로, 진화는 <그림 3>과 같이 진행된다. 여기서 일반적인 진화 알고리즘이 탐색하는 공간은 2차원 공간 전체가 되나, 협력 공진화 알고리즘이 탐색하는 공간은 그림상의 점선들로 축소되므로 빠른 진화가 가능하게 된다.



<그림 2> 협력 공진화에서의 염색체 평가



<그림 3> 협력 공진화 에서의 진화 과정

- **CCGA2**

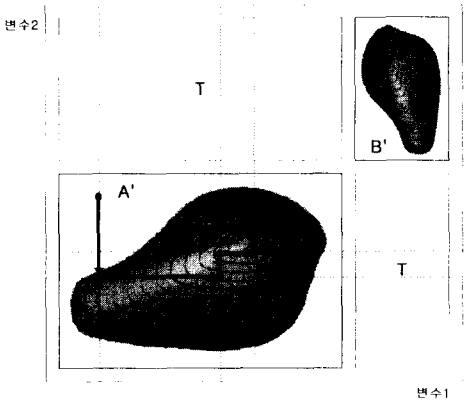
CCGA1은 진화 속도는 빠르나 <그림 4>와 같은 특정 문제—내쉬 평형점(Nash equilibrium point)[3]이 많은 문제—에 대해서는 일반적인 진화 알고리즘보다 성능이 떨어진다[4]. <그림 4>에서 극값은 A와 B가 존재한다. CCGA1의 측면에서 봤을 때, A'영역은 극값 A의 끌림유역에 해당되고, B'영역은 극값 B의 끌림유역에 해당된다. CCGA1에서는 진화 과정중 다른 변수들은 고정시키고 변수 하나씩 돌아가면서 진화시키기 때문에 탐색점을 지나면서 변수 축에 수직이 되는 직선들 만을 탐색하게 된다. 따라서 한번 어떤 극값의 끌림유역에 들어가게 되면, 그 끌림유역을 벗어나기가 힘들어진다. 만약 <그림 4>에서 극값 B가 전역 최적해일 때 A'영역에서 탐색을 시작했다면 절대 B'영역으로 갈 수가 없게 된다. 이런 상황에서는 내쉬 평형점의 정의에 의해, 변수1과 변수2를 동시에 변화시켜야지만 B'영역으로 갈 수 있다. 이럴 때, 변수1과 변수2에는 변수 의존성(variable dependency)가 존재한다고 한다. 이것을 식으로 나타내면 <식 2>와 같다. 적합도 값이 높은 개체가 좋은 개체라고 가정할 때, $\Delta f^a(c)$를 원래의 염색체 c에서 변수 x_a를 어떤 값으로 변화시켰을 때와 변화시키기 전의 염색체 c의 적합도 차이라고 가정하면, 변수 x_a와 변수 x_b에 의존성의 여부를 나타는 함수 $D(a,b)$는 (식 2)와 같이 정의된다[5].

$$D(a,b)=\begin{cases} False & \text{, if } \forall case-of\ \{\Delta f^a(c)+\Delta f^b(c)\} \rightarrow \\ & \quad \Delta f^a(c)+\Delta f^b(c) \geq \Delta f^{ab}(c) \\ True & \text{, if } \exists case-of\ \{\Delta f^a(c)+\Delta f^b(c)\} \rightarrow \\ & \quad \Delta f^a(c)+\Delta f^b(c) < \Delta f^{ab}(c) \end{cases} \quad \text{(식 2)}$$

CCGA1의 이러한 문제점을 해결하기 위하여 Potter와 DeJong은 CCGA2를 제안하였다. CCGA2는 CCGA1과 진화 방법은 유사하나, 평가 방법을 달리 함으로써 CCGA1과 구별된다. CCGA2에서 최적화 해야할 변수가 s개 일 때, k종족의 i번째 염색체의 적합도는 (식 3)과 같다.

$$F^{CCGA2}(c_i^k)=\max\left\{\begin{matrix} F(c^1_{elite}, c^2_{elite}, \ldots, c_i^k, \ldots, c^s_{elite}), \\ F(c^1_{rand}, c^2_{rand}, \ldots, c_i^k, \ldots, c^s_{rand}) \end{matrix}\right\} \quad \text{(식 3)}$$

여기서 c^k_{elite}는 k종족의 엘리트 염색체를 나타내고, c^k_{rand}는 k종족에서 무작위로 선택된 염색체를 나타내며, F는 문제 영역에서 전체 변수에 대한 적합도 함수를 나타낸다. 이와 같이 염색체를 평가할 때 다른 종족에서 무작위로 선택된 개체들과도 계산함으로써 특정 끌림유역을 벗어날 수 있도록 하였다.



<그림 4> 변수간에 의존성이 존재하는 문제

(2) 적응적 공진화(Adaptive Cooperative Coevolution)[2]

이 알고리즘은 K. Weicker와, N. Weicker가 CCGA2를 개선하여 1999년에 제안한 알고리즘으로서, 평가 방법과 의존도 행렬을 사용한다는 것을 제외하고는 CCGA와 같다. 적응적 공진화 에서의 적합도는 다음과 같이 계산된다. 최적화 해야할 변수가 s개 일 때, k종족의 i번째 염색체의 적합도는 (식 4)와 같다.

$$F^{ACC}(c_i^k)=\max\left\{\begin{matrix} F(c^1_{elite}, c^2_{elite}, \ldots, c_i^k, \ldots, c^s_{elite}), \\ F(c^1_{elite}, \ldots, c^j_{rand}, \ldots, c_i^k, \ldots, c^s_{elite}) \end{matrix}\right\} \quad \text{(식 4)}$$

여기서 c^k_{elite}는 k종족의 엘리트 염색체를 나타내고, c^j_{rand}는 $j(1 \leq j \leq s,\ j \neq k$ 인 임의의 수)종족에서 무작위로 선택된 염색체를 나타내며, F는 문제 영역에서 전체 변수에 대한 적합도 함수를 나타낸다. 이와 같이 계산할 때, (식 4)의 max에서 두 번째 항이 선택되어 적합도로 될 경우, 의존도 행렬에서 변수 x_j와 x_k사이의 의존도를 나타내는 값을 증가하도록 하여, 이 값이 주어진 값을 초과할 경우, j종족과, k종족을 병합하도록 하였다. 변수들 사이에 의존성이 없다면 (식 2)에 의해 (식 4)의 max에서 항상 첫 번째 항이 적합도로 선택이 될 것이므로, CCGA1의 알고리즘과 같이 진행되며, 둘 이상의 변수가 서로 의존성이 존재한다면, 두 변수를 나타내는 종족들은 서로 병합되고, 병합된 종족은 일반적인 진화 알고리즘이 적용되므로 병합된 둘 이상의 변수가 동시에 변화하여, 국소 최적해의 끌림유역을 벗어날 수 있게 된다.

3. 제안하는 알고리즘(SMGA: Split & Merge Genetic Algirithm)

종족이 병합되어 진화하는 어느 순간에는 두 가지 상황이 가능하다. 즉, 엘리트가 국소 최적해의 끌림유역에 존재할 경우와, 또는 엘리트가 전역 최적해의 끌림유역에 존재할 경우이다. 만약, 국소 최적해의 끌림유역을 탐색하고 있다면 계속 병합되어 있는 것이 효율이 좋겠지만, 이미 국소 최적해의 끌림유역을 벗어나서 전역 최적해의 끌림유역을 탐색중이라면 두 종족이 병합되어 있는 것 보다는 다시 분할되는 것이 효율이 좋아진다. 그러나 이같은 상황을 탐색 도중 알 수 없으므로, 본 논문에서는 병합된 종족이 일정시간동안 엘리트의 향상에 기여를 하지 못할 경우 분할을 하도록 하였다.

본 논문에서 제안하는 SMGA는 진화 과정중 병합단계와 분할단계의 두 가지 단계가 반복된다. 병합단계는 적응적 공진화에서의 병합 방법과 같은 방법을 사용하였으며, 병합 후, 병합된 종족이 주어진 세대동안 전체 엘리트 염색체(각 종족에서 엘리트만 모아놓은 염색체)의 개선에 기여를 못 할 경우 다시 분할을 하도록 하였다. 분할 후에도 의존도 행렬에 의해 다시 병합될 수 있도록 하여, 분할과 병합이 반복되게 하였다. 따라서 의존도가 높은 변수쌍일 경우 병합상태가 오래 지속되며, 의존도가 낮은 변수쌍일 경우 분할상태가 오래 지속되므로 효율적인 진화가 가능하다.

(1) 병합방법

병합단계에서는 적응적 공진화에서의 방법과 같다. 즉, j번째 종족과 k번째 종족이 병합할 경우 새로운 종족 jk의 개체군 Pop_{jk}는 다음과 같이 정의된다.[2]

$$\begin{matrix} Pop_j = \langle c_1^j, \ldots, c_n^j \rangle \\ Pop_k = \langle c_1^k, \ldots, c_n^k \rangle \\ \rightarrow \\ Pop_{jk} = \langle c_1^{jk}, \ldots, c_n^{jk} \rangle \end{matrix} \quad \text{(식 5)}$$

여기서 새로운 개체군 Pop_{jk}의 i번째 개체 c_i^{jk}는,

$$c_i^{jk}=\begin{cases} c^j_{elite} \circ c^k_{elite} & \text{, for } i=1 \\ c^j_{elite} \circ c^k_{rand} & \text{, for } 2 \leq i \leq \lceil \frac{n}{2} \rceil \\ c^j_{rand} \circ c^k_{elite} & \text{, otherwise} \end{cases} \quad \text{(식 6)}$$

와 같이 정의된다.

(2) 분할방법

$Extract^i(c_i^{jk})$를 j종족과 k종족이 병합된 개체군 중 i번째 염색체

c_i^{jk}에서, j종족에 해당되는 염색체의 부분만을 추출해내는 함수라고 가정하면, 개체군 Pop_{jk}에서 Pop_j의 i번째 개체 c_i^j와 Pop_k의 i번째 개체 c_i^k는 다음과 같이 정의된다.

$$c_i^j = \begin{cases} Extract^j(c_{elite}^{jk}) & , \text{for } i=1 \\ Extract^j(c_{rand}^{jk}) & , otherwise \end{cases}$$

$$c_i^k = \begin{cases} Extract^k(c_{elite}^{jk}) & , \text{for } i=1 \\ Extract^k(c_{rand}^{jk}) & , otherwise \end{cases} \quad (식\ 7)$$

4. 실험

모든 실험에서는 <표 1>과 같은 매개변수를 사용하여 실험하였다.

매개변수	값
개체군 크기	100
교배확률	0.6
돌연변이확률	0.1
선택방법	적합도비례선택
교배방법	2점교배
실험회수	50회
염색체 길이	16bit

<표 1> 실험에 사용된 매개변수

(1) Ackley 함수

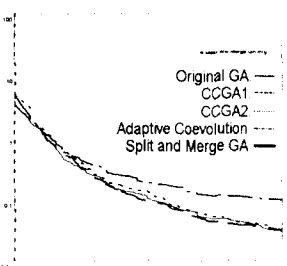
Ackley함수는 자체만으로는 변수간에 의존성이 존재하지 않는다. 따라서 축 회전(coordinate rotation)[6]을 하여 변수들 간에 의존성이 생기게 하였다. 최적화 하는 변수를 30개로 하여, 탐색공간을 크게 하였다.

Ackley 함수의 원형은 (식 8)과 같다.

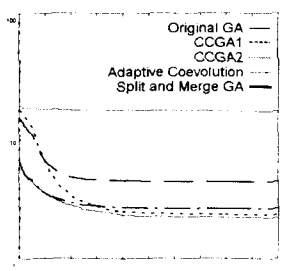
$$F(\vec{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e \quad (식\ 8)$$

실험 결과들 보면, 일반적인 진화 알고리즘은 탐색공간의 확대로 인하여 거의 진화가 이루어지지 않고 있다. 그리고, CCGA1은 국소 최적해에서 내쉬 평형이 이루어지고 있으며, CCGA2는 진화 초기에는 변수 의존성을 고려하는 방법들보다 진화 속도가 떨어지는 것을 알 수 있다.

그리고, ACC(Adaptive Cooperative Coevolution)와 SMGA는 진화 초기에는 흡사하게 진화 되지만, 진화 중반으로 갈수록 ACC의 종족들이 병합이되어, 진화 말기쯤에는 모든 종족이 하나로 병합되어 일반적인 진화 알고리즘과 같이, 방대한 탐색공간으로 인하여 진화 속도가 저하되는 것을 알 수 있다. 그에 비해 SMGA는 병합과 분할을 반복함으로써 계속적인 진화가 가능하다는 것을 알 수 있다.



<그림 5> Ackley 함수에 대한 실험 결과



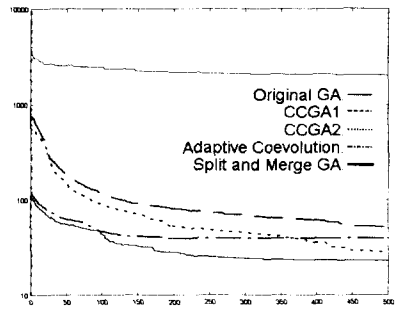
<그림 6> 변수축을 회전시킨 Ackley함수에 대한 실험 결과

(2) Rosenbrock 함수

Rosenbrock 함수는 좌표축 회전이 없이도 자체적으로 변수 의존성을 가지고 있는 함수이다. 함수의 형태는 다음과 같다.

$$F(\vec{x}) = \sum_{i=1}^{n/2}\left[100(x_{2i} - x_{2i-1}^2)^2 + (1 - x_{2i-1}^2)^2\right] \quad (식\ 9)$$

실험 결과를 보면 Ackley 함수 실험에서 보인것과 마찬가지로 CCGA1 국소 최적해에서 내쉬평형이 이루어지고 있고, CCGA2는 진화 초기에 변수 의존성을 사용하는 방법들보다 진화 속도가 떨어지며, ACC는 진화 중반에 모든 종족이 병합되어버려서 진화 말기에는 진화속도가 현저히 감소되는 것을 볼 수 있다.



<그림 7> Rosenbrock 함수에 대한 실험 결과

5. 결론 및 향후 연구

본 논문에서는 진화 알고리즘의 성능을 향상시키기 위하여, 기존의 방법들을 개선한 새로운 방법을 제시하였다. 실험 결과 제시한 알고리즘이 대상이 된 실험 함수들에 대하여 기존의 알고리즘보다 좋은 성능을 보였다. Potter와 DeJong이 제안한 CCGA1은 내쉬 평형점이 많은 문제들에 대해서는 그 중의 하나의 내쉬 평형점으로 수렴하여, 다른 점들을 탐색하지 못함으로써, 그다지 좋은 성능을 보이지 못하였고, CCGA2는 CCGA1의 이런 문제점을 개선 하였으나, 변수 의존성을 진화 상황에 효율적으로 반영하지 못함으로써 그다지 좋은 성능을 보이지 못하였다. 그리고 적응적 공진화는 진화 도중 병합만을 고려함으로써 진화 말기에는 모든 종족이 하나로 병합되어 진화 속도가 저하됨으로써, 좋은 결과를 보이지 못하였다. 그러나 본 논문에서 제안한 SMGA는 변수 의존성을 고려하면서도, 진화 도중 병합과 분할을 반복함으로써, 적응적 공진화의 문제점을 해결하여 다른 알고리즘들 보다 좋은 성능을 보여주었다.

지금까지 벤치마크 함수(benchmark function)들에 대해서만 실험을 했기 때문에, 제안하는 알고리즘의 다당성과, 일반성을 검증하기 위해서 실세적인 문제에 적용을 해 보아야 할 것이다. 그리고, 분할의 순간을 결정할 때, 현재는 전체적인 엘리트 염색체의 진화에 기여를 하는 것을 관찰하여 결정하였으나, 이것을 개선하여 국소 최적해의 끌림유역을 벗어나, 전역 최적해의 끌림유역에 들어갔을 경우를 알아내어 이 시점을 분할의 시점으로 하여야 할 것이다.

6. 참고문헌

[1] Potter, M. A. and K. A. DeJong (1994). A cooperative coevolutionary approach to function optimization. In Y. Davidor and H.-P Schwefel (Eds.), *Proceedings of the Third Conference on Parallel Problem Solving from Nature*, pp. 249-257. Springer-Verlag

[2] Weicker, K. and Weicker N. (1999). On the improvement of coevolutionary optimizers by learning variable interdependencies. *Congress on Evolutionary Computation, CEC99*, pp. 1627-1632

[3] Nash, J. (1951). Non-cooperative games. *Annals of Mathematics* 54(2), 286-295

[4] Potter, M. A. (1997). The design and analysis of a computational model of cooperative coevolution. Ph. D. thesis, George Mason University, Fairfax, Virginia.

[5] Munetomo, M. and Goldberg, D. E. (1999). Identifying Linkage Groups by Nonlinearity/Non-monotonicity Detection, In Banzhaf, W. et al. (Eds.). *Proceedings of the Genetic and Evolutionary Computation Conference 1999 (GECCO-99)*. San Fransisco, CA: Morgan Kaufmann.

[6] Salomon, R. (1996). Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions. *BioSystems* 39, 210-229