

Sequential Ordering Problem을 위한 유전 연산자의 비교

이혜리, 이견명
충북대학교 컴퓨터 과학과

A Comparative Study of Genetic Operators for the Sequential Ordering Problem

Hye-Ree Lee, Keon-Myung Lee
Dept. of Computer Science, Chungbuk National University

요 약

Sequential Ordering Problem(SOP)은 여러 개의 도시를 방문함에 있어 '어떤 도시를 다른 도시보다 먼저 방문해야 한다'는 선행제약이 있는 비대칭 순회 세일즈맨 문제(Traveling Salesman Problem)로서, 주어진 선행 제약을 만족하면서 모든 도시를 한번씩만 경유하는 가장 짧은 경로를 찾는 NP-Complete에 속하는 문제이다. 유전자 알고리즘은 SOP와 같은 조합 최적화문제에 대해 유용한 메타휴리스틱의 한가지이다. 본 논문에서는 SOP에 유전자 알고리즘을 적용할 때, 선행제약을 만족하는 해를 생성하는데 사용할 수 있는 선행관계유지 유전 연산자를 소개하고 이를 비교한다. 비교하는 유전 연산자는 선행관계유지 교차연산자, 선행관계유지 순서기반 교차연산자, 최대부분순서/임의삽입 연산자, 선행관계유지 간선재결합 연산자이다.

1. 서 론

Sequential Ordering Problem(SOP)은 여러 개의 도시를 방문함에 있어 '어떤 도시를 다른 도시보다 먼저 방문해야 한다'라는 선행제약이 있는 비대칭 순회 세일즈맨 문제(Traveling Salesman Problem - TSP)로서, 주어진 선행 제약을 만족하면서 모든 도시를 한번씩만 경유하는 가장 짧은 경로를 찾는 NP-Complete에 속하는 문제이다. 비대칭 TSP는 도시 i 부터 도시 j 까지의 거리와 도시 j 부터 도시 i 까지의 거리가 다를 수 있는 TSP를 말한다.

지난 수년간 SOP를 해결하기 위해서 여러 가지 방법이 제안되어 왔다. 대표적인 방법으로는 Hybrid Ant System[1]이나 Lagrangean relax-and-cut 접근방법[2], Cutting Plane 접근방법[3], 유전자 알고리즘[5,6,7] 등이 있다. 이중 유전자 알고리즘은 NP-Complete 조합 최적화 문제에 쉽게 적용할 수 있는 메타휴리스틱의 하나이다. 본 논문에서는 SOP에 유전자 알고리즘을 적용할 때 선행제약을 만족하는 후보해를 생성하는데 사용할 수 있는 유전 연산자를 소개하고 이들의 성능을 비교한다. 비교하는 유전 연산자는 선행관계유지 교차(Precedence-preserving Crossover)연산자[5], 선행관계유지 순서기반 교차(Precedence-preserving Order-based Crossover)연산자[6], 부모 염색체의 공통성(Commonality)을 강조한 최대부분순서/임의삽입(Maximum Partial Order /Arbitrary Insertion) 연산자[7], TSP의 문제 해결을 위해 제안된 간선재결합(Edge Recombination) 연산자[8]를 SOP에 적용할 수 있도록 수정한 선행관계유지 간선재결합(Precedence-preserving Edge Recombination - PPER) 연산자 등이다.

2. 유전자 알고리즘

유전자 알고리즘은 여러 개의 개체들이 동시에 병렬적으로 주어진 환경에 따라서 최적의 상태를 향하여 진화생존의 방법으로 진화한다는 생체계의 진

화론에서 따온 문제해결방법으로 Holland[4]에 의해 제안된 것이다. 진화론에 의하면 부모 개체의 교배를 통하여 부모의 염색체와 다른 새로운 자손 염색체가 만들어져서 진화가 이루어지는데, 유전자 알고리즘에서는 유전 연산자가 새로운 지손 염색체를 생성하는 역할을 담당한다. 대표적인 유전 연산자로는 두 개의 염색체(후보해)로부터 일부분을 가져와 새로운 해를 구성하는 교차(Crossover)연산자와 어떤 염색체의 일부 성질을 임의로 바꾸어 새로운 해를 만드는 돌연변이(Mutation)연산자 등이 있다.

일반적으로 유전자 알고리즘은 다음과 같은 단계로 구성된다

- 단계 1. 후보해들로 이루어진 모집단을 초기화한 후에 적합도를 평가한다.
- 단계 2. 임의의 방법으로 선정된 후보해들에 대해 유전 연산자를 적용하여 새로운 후보해를 생성한다.
- 단계 3. 새로 생성된 후보해를 모집단에 넣기 위하여 기존의 모집단에서 적합도가 낮은 후보해를 제거한다
- 단계 4. 새로운 후보해를 평가한 후 모집단에 추가한다.
- 단계 5. 종료 조건을 만족하면 적합도가 가장 큰 후보해를 출력하고, 그렇지 않으면 단계 2로 돌아간다

3. SOP에 적용 가능한 유전 연산자

유전자 알고리즘을 어떤 문제에 적용하기 위해서는 후보해의 염색체 형태의 표현방법, 초기 모집단 구성방법, 후보해 생성을 위한 유전 연산자, 후보해의 평가 방법 등이 고안되어야 한다. 유전자 알고리즘을 SOP에 적용할 경우, 후보해의 표현은 도시 나열방법, 행렬표현방법, 간선표현방법 등이 있을 수 있고, 후보해의 평가는 문제 자체에서 주어진 후보해가 나타내는 경로의 길이를 계산하는 것이 된다. 초기 모집단은 문제 자체에서 주어진 도시간의 선행관계를 위상관계로 해석하여 무라위 순서의 위상정렬(Topological Sort) 방법으로 구성할 수 있다. 유전 연산자는 선행관계를 만족시키는 후보해를 생

성할 수 있는 것이어야 하는데, 적용되는 유전 연산자 특성에 따라 유전자 알고리즘의 성능에 영향을 많이 받는다. 이 절에서는 SOP에 적용될 수 있는 선행관계를 만족하는 후보해를 생성할 수 있는 유전 연산자를 소개한다. 이들 연산자는 선행관계유지 교차연산자[5], 선행관계유지 순서기반 교차연산자[6], 최대부분순서/임의삽입 연산자[7], 기존의 간선재결합 연산자[8]를 수정한 선행관계유지 간선재결합 연산자 등이다.

3.1 선행관계유지 교차연산자

선행관계유지 교차(Precedence-Preserving Crossover : PPX)연산자[5]는 다음과 같은 과정으로 새로운 염색체를 생성한다.

1. 두 개의 부모 염색체에 대해서 두 개의 절단위치를 무작위로 선택한다. 이 때 부모 염색체는 이미 방문순서 제약에 대한 선행관계를 만족하고 있다.
2. 부모 염색체 1의 첫 번째 절단위치의 왼쪽 부분과 두 번째 절단위치의 오른쪽 부분을 자손 염색체 1의 같은 위치에 복사한다. 부모 염색체 1의 두 절단위치의 가운데 부분에 있는 원소(유전자)들은 이들이 부모 2에서 나타나는 순서대로 자손 염색체의 가운데 부분에 복사한다

(그림 1)은 PPX가 어떻게 동작하는지 보여주는 예이다. |는 선택된 절단위치를 나타낸다

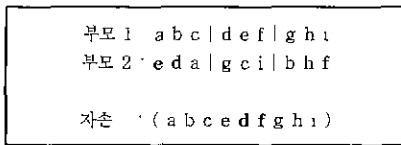


그림 1 PPX의 예

3.2 선행관계유지 순서기반 교차연산자

선행관계유지 순서기반 교차(Precedence-preserving Order-based Crossover : POX)연산자[6]가 새로운 염색체를 생성하는 과정이다

1. 부모 염색체 1에서 임의의 원소(유전자)를 선택한다
2. 선택한 유전자와 선행 제약이 있는 원소들을 찾는다.
3. 부모 염색체 1에서 선택된 원소들을 자손 염색체 1에서 부모 염색체1과 같은 지리에 복사한다
4. 나머지 원소들을 부모 염색체 2에서 나타난 순서대로 자손 염색체 1에 복사한다

(그림 2)는 POX가 어떻게 동작하는지 보여주는 예이다 임의로 선택된 원소는 e이고, e는 { b, e, h }의 선행제약이 있다고 가정할 것이다

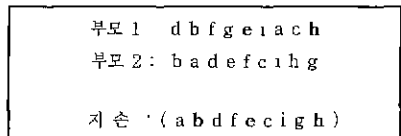


그림 2. POX의 예

3.3 최대부분순서/임의삽입 연산자

Chen[7]이 제안한 최대부분순서/임의삽입(Maximal Partial Order /Arbitrary Insertion : MPO/AI) 연산자는 자손 염색체를 생성할 때 부모 염색체의 공통된 구성블럭(building blocks 또는 schemata)을 최대한 많이 유지하도록 하는데 주안점을 두고 개발된 것이다. 다음은 두 부모 염색체의 공통된 부분순서 중에서 가장 긴 순서를 가지는 최대부분순서를 찾는 과정을 나타낸 것이다.

- 1 Boolean 행렬로 두 부모 염색체를 나타낸다.
- 2 2개의 행렬을 이용하여 교차행렬을 만든다
3. 각 원소의 선행자들을 일기 위하여 열을 합한다.

4. 부분 순서 그래프를 만든다

4.1 가장 적은 선행자들을 가지는 원소를 찾는다.

4.2 앞선(most ordered) 선행자들에게 연결 가능한 선행자를 연결한다.

5 단계 4를 반복하여 그래프를 완성한 후 가장 긴 길이(path)를 찾는다 이 과정을 통해서 최대부분순서를 찾은 후에 최대부분순서에 속하지 않는 임의의 원소(유전자)를 선택하여 주어진 SOP의 선행제약을 만족하면서 최소의 적합도를 가질 수 있는 위치에 그 원소를 삽입하여 자손을 생성한다.

3.4 선행관계유지 간선재결합 연산자

Whitley[8]가 TSP의 문제해결을 위해 개발한 간선재결합(ER) 연산자는 다른 유전 연산자들이 유전자의 위치나 순서를 고려하여 기존 염색체를 생성하는 것에 초점을 맞춘 것과는 달리, 유전자 사이의 간선을 고려하여 부모의 정보를 자손에게 전달한다 ER은 이 전달 과정에서 부모 염색체에서 지어오 하나의 원소와 인접되어 있는 원소를 나타내는 간선 목록(edge list)을 가지고 있는 테이블을 생성한다. ER의 자손 생성과정을 간략히 보면 부모의 첫 번째 원소 중에 하나를 임의로 선택한 후에 간선 목록을 이용하여 전체적인 자손 염색체를 생성한다. 선행관계유지 간선재결합(Precedence-Preserving Edge Recombination : PPER) 연산자는 간선 목록을 이용하나 선행제약이라는 추가적인 조건으로 인하여 자손을 만들어 가는 과정에서 하나의 원소를 선택할 때마다 선행제약을 만족하는지의 여부를 검사하는 과정이 추가된다.

(그림 3)은 PPER이 어떻게 동작하는지 보여주는 예로써, b->c->e의 선행제약이 있다고 가정할 때 자손염색체의 첫 번째 오는 원소는 c와 e를 제외한 나머지 원소 중에 하나여야 한다 (그림 3)에서는 자손 염색체의 첫 번째 원소로 a를 선택했으며, 다음에 오는 원소는 city a의 목록 중에서 선행제약을 만족해야 하므로 c와 e를 제외한 b와 f중에서 선택해야 한다. 그런데 b는 간선이 3개, f는 간선이 4개이므로 f를 선택한다. 이러한 방법으로 자손 염색체를 구성한다.

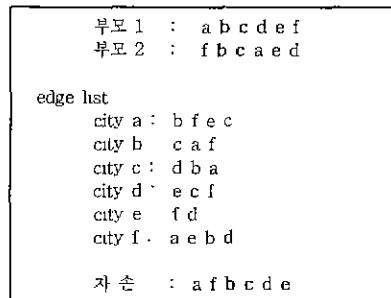


그림 3 PPER의 예

4. 실험 및 결과

본 실험에서는 Sun사의 UltraSparc에서 Whitley의 GENITOR[9]를 이용하여 SOP를 해결하기 위한 유전자 알고리즘을 구현하였고, 각 유전 연산자의 성능을 비교하기 위하여 두가지로 실험을 하였다. 처음 실험은 각 유전 연산자를 사용하는 유전자 알고리즘에 대해서 같은 매개 변수를 적용하여 10번 실험하였고, 다음 실험은 각 유전자 알고리즘에 대해서 근사 최적해를 찾을 수 있도록 매개변수를 변화시키면서 실험하였다 처음 실험에서 적용한 매개 변수들을 보면 부모 염색체 선택에 영향을 주는 선택편향(selection bias)은 17, 모집단의 크기는 500, 세대교체수는 300,000번이었다. 표 1은 첫 번째 실험에 대한 결과로서 Best는 10번의 전체 실험 결과 중에서 제일 좋은 해를 나타낸 것이며, Avg.는 10번의 각 실험에서 얻은 가장 좋은 해들의 평균을 나타낸 것이다. 두 번째 실험에서는 충분한 세대교체를 하면서 모집단의 크기와 선택편향값을 변화시키면서 실험하였다. 실험에서 사용한 데이터는 TSPLIB [10]에 있는 SOP에 대한 벤치마크 데이터 중 임의로 8개를 선택한 것이다 실험결과 MPO/AI가 실험 1이나 실험 2 모두에서 가장 좋은 성능을 보였다

으며, 모집단의 크기와 선택편향값의 변화에 큰 영향을 받지 않았다. PPER의 경우는 모집단의 크기가 비교적 높은 경우에 조금 더 좋은 결과를 나타내고 있으나, 표 1의 결과값과 많은 차이를 보이지 않았다. PPX와 POX는 매개변수의 변화로 성능이 향상되었고 높은 선택편향값과 모집단의 크기를 낮은 값으로 적용하는 경우에 더 좋지 않은 결과를 나타내고 있는 등, 두 유전 연산자가 비슷한 결과를 보이고 있다.

(그림 4)와 (그림 5)는 도시의 수가 각각 54, 71이면서 선행제약의 수가 증가할 때 각 유전 연산자의 수행 시간을 나타내고 있다. 두 그림에서 보면 대체적으로 같은 크기의 도시수에 대하여 선행제약의 수가 증가함에 따라 약간씩 시간이 증가하기는 했지만 거의 비슷한 수행시간을 보이고 있다. 실험 1과 실험 2에서의 유사한 결과값을 가졌던 PPX와 POX는 수행시간에서도 비슷한 결과를 보이고 있다. 결과적으로 MPO/AI가 가장 많은 시간이 걸렸고, 이 시간은 PPER의 약 2배, POX와 PPX의 약 20배나 되었다.

5. 결 론

본 논문에서는 SOP의 근사 최적해를 구하기 위한 유전자 알고리즘에 사용할 수 있는 항상 선행제약을 만족하는 염색체를 생성하는 4가지 유전 연산자를 소개하고 비교 실험하였다. 전체적인 실험결과를 보았을 때 MPO/AI가 가장 성능이 좋은 것으로 나타났으나 수행 시간이 너무 오래 걸리는 단점이 있었다. 그에 비해서 PPER은 성능 면에서는 MPO/AI와 그리 차이가 나지 않으면서 시간이 MPO/AI의 반정도나 걸렸다.

유전 연산자들 사이에서 나타난 주된 차이점은 재조합 과정에서 그들이 유지하려고 하는 정보가 있다. MPO/AI가 중요하게 생각하는 정보는 부모 염색체가 공통으로 가지고 있는 구성블럭이었고 PPER은 부모 염색체에서 어떤 유전자로부터 다른 유전자와의 연결을 나타내는 인접(adjacency) 정보였다. 이 정보들은 SOP의 문제해결에 있어 유리하게 이용되었고 실험 결과 좋은 성능을 나타냈다. PPX나 POX는 부모 염색체의 단순한 위치와 순서에 관한 정보를 중요하게 생각했다. 그러나 이 정보는 SOP의 문제해결에 성능이 좋지 않은 것으로 나타났다.

6. 참 고 문 헌

- [1] L. Gambardella, M. Dorngo, HAS-SOP: Hybrid Ant System for the Sequential Ordering Problem, *Technical Report IDSIA 11-97*, Lugano, Switzerland, 1997
- [2] L.F. Escudero, M. Guignard, K. Malik, A Lagrangean relax-and-cut approach for the sequential ordering problem with precedence constraints, *Annals of Operations Research*, Vol.50, pp 219-237, 1994
- [3] N. Ascheuer, L. Escudero, M. Grottschel, M. Stoer, A cutting plane approach to the sequential ordering problem (with applications to job scheduling in manufacturing), *SIAM Journal on Optimization*, Vol.3, pp 25-42, 1993
- [4] J. Holland, *Adaptation in Neural and Artificial Systems*, Vol.1, pp.3-22, 1989
- [5] 이경미, 이진명, 방문순서 제약이 있는 순회 세일즈맨 문제를 위한 유전자 알고리즘, *정보과학회 논문지*, Vol 4, No 2 pp 362-363, 1998
- [6] K.M. Lee, T. Yamakawa, K.-M. Lee, A Genetic Algorithm for General Machine Scheduling Problems, *Int'l Conf on Conventional and Knowledge-based Electronics Systems (KES'98)*, Australia, pp.60-65, 1998
- [7] S. Chen, S. Smith, Commonality and Genetic Algorithms, *CMU-RI-TR-96-27*, 1996
- [8] D. Whitley, T. Starkweather, D.A. Fuquay, Scheduling Problems and Traveling Salesman - The Genetic Edge Recombination Operator. *Proc of the Third Int'l Conference on Genetic Algorithm*, Morgan Kaufmann Publishes, San Mateo, CA, pp. 133-140, 1989
- [9] D. Whitley, J. Kauth, *GENITOR : A Different Genetic Algorithm*, *Proc. Rocky Mountain Conf. on Artificial Intelligence*, 1988
- [10] <http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/sop>

SOP Instance	Size	Constraints	MPO/AI		PPER		PPX		POX	
			Best	Avg.	Best	Avg.	Best	Avg.	Best	Avg.
ft53.1	54	12	7531	7531.0	8422	8955.3	10117	11868.2	11095	11983.7
ft53.2	54	25	8072	8091.2	9558	9833.3	11028	11922.8	11527	12118.2
ft53.3	54	48	10353	10370.3	12833	13429.6	13452	14051.8	13439	13951.1
ft53.4	54	63	14425	14425.0	15339	16005.0	16292	16685.5	16385	16641.0
ft70.1	71	17	39395	39280.2	43022	43240.3	49622	50364.4	49035	49878.6
ft70.2	71	35	40443	40454.0	44862	45756.3	47825	50374.2	49392	50319.3
ft70.3	71	68	42535	42547.7	47749	48745.6	51074	52012.7	50378	51514.6
ft70.4	71	86	53571	53581.8	56489	54100.5	57187	57822.3	57167	57985.2

표 1 8개의 SOP Instance를 사용한 4개의 유전 연산자의 실험 결과 (매개변수 고정)

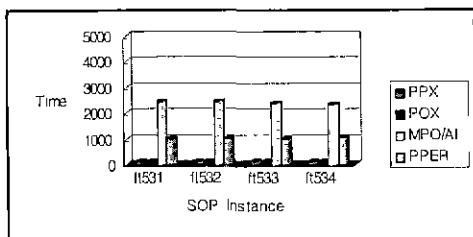


그림 4 각 유전 연산자의 수행 시간(도시수 54)

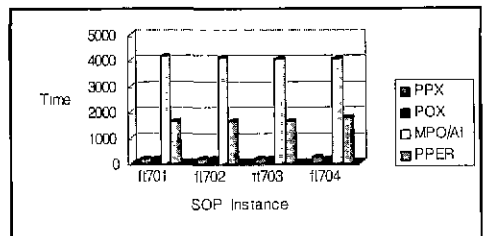


그림 5 각 유전 연산자의 수행 시간(도시수 71)