마이크로 유전자 알고리즘을 이용한 2차원 부재의 최적화 배치

김한중, 조범준
조선대학교 컴퓨터 공학과

Optimal Two-Dimemsional Layout using Micro-Genetic Algorithms

Han-Joung Kim, Beom-Joon Cho
Dept. of Computer Engineering, Chosun Univ.
E-mail : dosaking@lycos.co.kr, bjcho@mail.chosun.ac.kr

요 약

산업현장에서 생산원가의 절감에 관심이 많아지면서 넓은 원자재에서 크기가 다양한 작은 부재를 절단하여 소재를 얻는 경우, 넓은 원자재에 소재를 배치하는 상태에 따라 원자재의 사용비율은 달라지게 된다. 같은 양의 제품을 만들 때 원자재의 사용량을 줄일 수 있으면 그 만큼 원가를 절약할 수 있다. 따라서 본 논문에서는 원자재에 소재를 배치하는 기본 논리와 방법을 고찰하고, 적당하며 실용적인 방법을 제시하며, 이러한 기반을 토대로 Micro-Genetic Algorithms(μGAs)을 이용한 2차원적 부재의 최적화 배치 효율을 측정하여 기존의 배치알고리즘과 비교하였다.

위의 과정을 컴퓨터 시뮬레이션을 이용하여 구현하였으며 기존의 연구보다 좋은 배치 효율을 얻을 수 있었다. 이렇게 하여 얻어진 배치의 결과는 최적에 가까운 상태를 얻을 수 있음을 확인하였으며 원자재의 사용을 최소화 할 수 있었다.

1. 서론

산업현장에서 넓은 원자재에서 크기가 다양한 작은 부재를 절단하여 소재를 만드는 경우, 부재를 배치하는 요령에 따라 쓰이는 원자재의 량이 달라질 수 있다. 보통 경험에 의한 값으로 결정하여 사용하는데, 노련한 숙련자의 경우 경험에 의하여 넓은 부재를 먼저 잘라내고 작은 소재를 나중에 배치해 원자재의 사용량을 줄인다. 이것은 넓은 부재와 부재의 사이의 공간에는 작은 부재를 설치할 공간이 비교적 마련되기 쉽기 때문이다.

이러한 경험적, 실험적 논리를 이용하여 이와 유사한 모든 배치의 문제에 큰 부재를 먼저 배치하는 방법이 사용되고 있다. 하지만 큰 부재부터 배치하는 경험적 방법이 유리하다 하더라도 최적의 배치의 해를 얻는지는 못한다. 부재에 있어 배치의 효율은 일정한 순서만으로 결정될 수 없기 때문이다. 또한 2차원 배치의 경우 소재의 모양과 크기 형태의 조건을 제한하여 변수를 위치와 각도로 획일화시키더라도 발생할 수 있는 값의 범위는 특정 값이나 경계 치로 구분할 수 없다. 이것은 경우의 수가 무한대임을 의미한다. 또한 2차원의 부재배치에 대한 최적의 해를 구하는 알고리즘은 현재 존재하지 않는다.

이런 경우의 해에 가까운 값을 구하는 방법으로 여러 가지 방법이 쓰이고 있으며 그중 유전자 알고리즘(Genetic Algorithms)이 유용하게 쓰일 수 있다. 현재 많은 분야서 효율적인 배치 알고리즘이 연구되고 있

으며 외국의 몇몇 보고서에 의하면 최적의 해를 구하는 객체지향 알고리즘이 연구되고 있다.

따라서 본 연구에서는 Micro-Genetic Algorithms(μGAs)[1, 2, 3]을 이용하여 2차원 부재의 최적화 배치를 하고 그 결과를 입증하고자 한다.

2. 기하학적 형상 배치 전략

배치전략이란 원자재에 소재를 배치하기 위한 기준을 정의하는 작업이다. 배치작업을 일원화하여 Flow-chart에 대입될 수 있는 기준을 설정하는 작업으로 다음과 같은 배치 전략을 세웠다.

(1) 부재의 배치는 왼쪽에서 오른쪽으로 진행한다.
(2) 같은 부분일 경우 하단에서 상단으로 배치한다.
(3) 부재의 배치효율은 원자재의 크기 및 사용된 부재의 크기의 비로 결정한다.

2.1 교차점 탐색

그림 1과 같은 두 직선($a_1x_1+b_1y_1+c_1=0$, $a_2x_2+b_2y_2+c_2=0$)간의 교차점(x_0 , y_0)을 식 2-1과 같이 계산한다.

여기에서 만일 $a_1b_2-a_2b_1$ 이 0(zero)이면 이 두 직선은 평행하고 교차점이 존재하지 않는다. (x_1, y_1) 과 (x_2, y_2) 이 직선1의 임의의 점이고 (x_3, y_3) 과 (x_4, y_4) 이 직선2의 임의의 점이면 그때의 교차점을 (x_0, y_0) 이다.

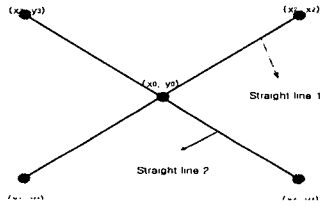


그림 1. 두 직선간의 교차점

$$x_0=\frac{b_1c_2-b_2c_1}{a_1b_2-a_2b_1},\quad y_0=\frac{c_1a_2-c_2a_1}{a_1b_2-a_2b_1} \qquad (식\ 1)$$

계수 ($a_{1,2}, b_{1,2}, c_{1,2}$)들은 다음과 같이 계산된다.

$$a_1=-\frac{y_2-y_1}{x_2-x_1},\quad a_2=-\frac{y_4-y_3}{x_4-x_3}$$
$$b_1=1.0\quad b_2=1.0 \qquad (식\ 2)$$
$$c_1=\frac{y_2-y_1}{x_2-x_1}\cdot x_1-y_1,\quad c_2=\frac{y_4-y_3}{x_4-x_3}\cdot x_3-y_3$$

2.2 Polygon 면적 계산

정점 $\{(x_i, y_i);\ i=1,\cdots,m\}$을 갖는 Polygon의 면적은 다음과 같은 식(Green's theorem)을 이용하여 구한다.

$$Area=\frac{1}{2}\cdot\left|\sum_{i=1}^{m}\{x_i\times(y_{i+1}-y_i)-y_i\times(x_{i+1}-x_i)\}\right| \qquad (식\ 3)$$

2.3 Polygon상의 경계 판정

일반적으로 어떤 점에서 Polygon에 무한 연장선을 가상하여 연장선과 Polygon이 교차하는 교차점의 숫자를 2로 나누어 그 숫자가 0이면 외부에 존재하는 점이며, 1이면 내부에 존재하는 점임을 알 수 있다. 그리고 Polygon의 라인선상에 존재하면 이 값은 0이나 2가 될 수 있는데 Point로부터 Polygon에 연장선(Extension Line)을 그리는 방향이 중요하게 된다. 이 경우 그 연장선이 Polygon의 정점을 지나면 이 알고리즘은 사용할 수 없게 되어 연장선은 Polygon의 정점을 피하여 그어져야 한다. 따라서 다음과 같은 알고리즘을 고안하였다.

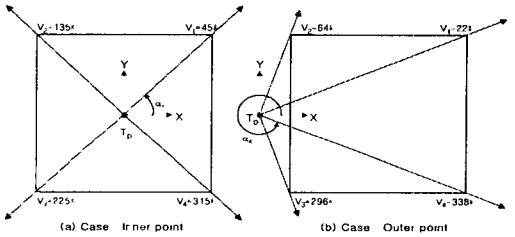


그림 2. Polygon상의 경계 판정 예

표 1. 그림 2의 Vector각도 계산 결과

경우 : 내부 점		
변수	Vector 각도	누적 각도
V_2 V_1	135° - 45° = 90°	90°
V_3 V_2	225° - 135° = 90°	180°
V_4 V_3	315° - 225° = 90°	270°
V_1 V_4	45° - 315° = -270° ⇒ 360° - 270° = 90°	**360°**
경우 : 외부 점		
변수	Vector 각도	누적 각도
V_2 V_1	64° - 22° = 42°	42°
V_3 V_2	296° - 64° = 232° ⇒ 232° - 360° = -128°	86°
V_4 V_3	338° - 296° = 42°	44°
V_1 V_4	22° - 338° = -316° ⇒ 360° - 316° = 44°	**0°**

우리는 이 알고리즘을 Angle Detection이라 부르기로 하였다. 표 1과 그림 2에서와 같이 연장선을 그리지 않고 정점의 위치를 감별하는 것으로 Polygon의

각도를 측정하는 것만으로 Inside와 Outside의 위치를 판단할 수 있었으며 점이 Polygon의 선상에 있는지의 여부는 정점으로부터 각도가 180도가 나오면 Polygon의 선상에 있는 점으로 규정할 수 있다.

2.4 Polygon Direction 판정

작성된 Polygon의 Vertex(정점)의 위치와 배열을 통하여 Polygon의 방향성(시계방향 또는 반시계방향)을 알아내는데 이것은 Polygon의 합집합, 차집합, 교집합을 결정하는데 중요한 요인이다.

Polygon상의 2개이 Vertex를 취하여 Vertex가 가지는 방향과 일치하는 중점에서 오른쪽으로 미세한 거리의 점을 P(x,y)라 하고 이점이 Polygon의 내부에 있으면 CW(시계방향), 외부에 있으면 CCW(반시계방향)으로 결정된다.

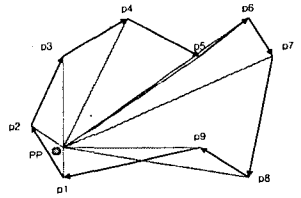


그림 3. Polygon 방향 판정

위 그림 3에서 p1에서 p2의 중심 방향의 오른쪽에 가상점(PP)을 생각하고 이점에서의 내각의 합이 360도(내부점) 임으로 이 Polygon은 반시계방향(CCW)으로 판정한다. 반대로 0도(외부점)의 경우 CW로 판정한다.

2.5 Polygon Sliding

Polygon의 설정위치를 정하는 알고리즘으로 2개의 Polygon의 면적을 일치시키기 위해 사용한다. Poly1의 Vertex에서 Poly2의 Vertex를 이동시켜 최소이동거리를 계산하는 방법이다.

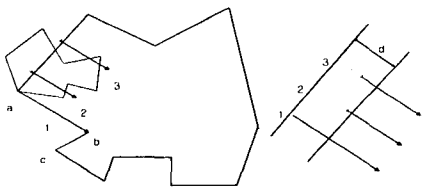


그림 4. Sliding 이동거리 결정

그림 4에서 보는 바와 같이 Polygon A의 정점 (p1, p2, p3,...)에서 a-b점의 각도를 따라 연장한 라인 1, 2, 3...을 추출할 수 있다. 이중에서 이동거리가 가장 긴 2번 라인의 점선 부위는 2개의 Polygon이 a에서 b의 선상으로 Sliding할 때 Polygon A의 면적이 Polygon B의 면적에 정확히 병합되는 이동거리(d)를 얻는 값을 얻을 수 있다.

2.6 Polygon Clipping

2개의 Polygon A와 B를 연결하여 2개의 Polygon 간의 교집합, 합집합, 차집합 등을 구한다. 그림 5는 그 간략한 예를 설명한다.

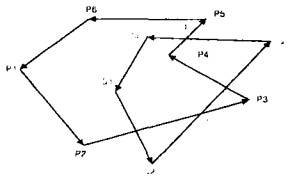


그림 5. Polygon Clipping 예

Polygon A = { P1, P2, ... PN },

Polygon B = { Q1, Q2, ... QM }으로 규정할 때.

Polygon A와 B의 교차점 = { I1, I2 .... IK } 이다.

Polygon A ∩ B = { P4, I4, Q4, Q1, I1, I2, I3 }

Polygon A - B = { P1, P2, I1, Q1, Q4, I4, P5, P6 }

Polygon A ∪ B = { P1, P2, I1, Q2, I2, P3, I3, Q3, I4, P5, P6 }

Polygon A와 B가 2.4에서 얻은 값으로 방향(CCW)이 일치할 때를 기준으로 다음과 같이 계산한다.

1 단계: Polygon A와 B에 교차된 Vertex(I1, I2, ... In)을 추가시켜 Polygon A'와 B'를 만든다.

Polygon A' = { P1, P2, I1, I2, P3, ... PN },

Polygon B' = { Q1, I1, Q2, I2, I3, Q3, ... QM }

2 단계: 2.3의 경계검출에 의하여 Polygon A'에서 B'의 각 정점에 대한 위치를 파악하여 Polyline을 만든다.

Polygon A'에서 B'	
P1-P2	외부
P2-I1	외부
I1-I2	내부
I2-P3	외부
P3-I3	외부
I3-P4	내부
P4-I4	내부
I4-P5	외부
P5-P6	외부
P6-P1	외부

Polygon B'에서 A'	
Q1-I1	내부
I1-Q2	외부
Q2-I2	외부
I2-I3	내부
I3-Q3	외부
Q3-I4	외부
I4-Q4	내부
Q4-Q1	내부

3 단계: 얻어진 Polyline을 병합하여 새로운 얻고자 하는 Polygon을 얻을 수 있다.

예) Polygon A ∪ B는 A′ B′에서 검출한 모든 외부 정점은 연결한다. 만일 검색 도중 교차점인 I의 위치에서 Polygon의 검색을 A′에서 B′로 또는 B′에서 A′로 변경하며 선상의 점으로 발견될 때는 A′를 그대로 유지한다. 만일 A′에서 B′로 이동 전에 A′의 다음 정점이 외부이면 A′를 B′로 이동하지 않는다.

Polygon A ∪ B = { P1, P2, I1, Q2, I2, P3, I3, Q3, I4, P5, P6 }

똑같은 의미로 Polygon A - B인 경우는 A′의 외부와 B′의 내부를 검색하고 Polygon A ∩ B의 경우는 두 Polygon A′와 B′의 내부를 검색한다.

3. Genetic Algorithms

Genetic Algorithms는 진화연산(Evolutionary Com -putation)에 바탕을 둔 알고리즘으로 우수한 자료를 변형한 자료는 결과가 우수하게 나올 가능성이 많은 관계로 우수한 부모를 선택하여 발전시킴으로 좀더 우수한 후손을 얻는 일종의 확률론적 방법이다.

일반적인 GA의 흐름도는 그림 6과 같다.

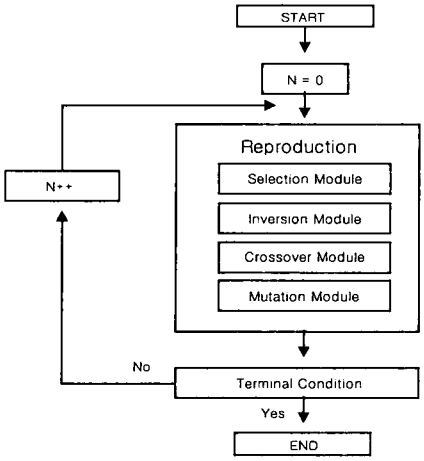


그림 6. 유전자 알고리즘의 흐름도

일반적인 유전자 알고리즘(GAs)은 약 30 ~ 200개의 개체 수를 가지고 수행하지만 본 연구에서 개발된 μ GAs는 5개의 개체 수를 가지고 사용하였다. 이러한 개체수의 축소로 인하여 유전 발전의 속도를 증가시키는 알고리즘이다. 아래에서 언급되어지는 4개의 Module로 구성된 재생산기법(Reproduction)은 이전 세대에서 일정수의 우수한 종들을 선택하여 새로운 세대의 개체군을 만드는 과정이다.

3.1 Selection Module

본 연구에서는 Selection Module로서 Air-Borne Selection Module[4]이 개발되었다. 이 선택 모듈은 전체 해의 공간에서 더 나은 최적 해를 찾기 위하여 일반적 유전자 알고리즘의 Ranking Select Module보다 더 빠른 탐사를 수행한다. 선택된 개체군들은 이전 세대의 개체군과 유전 연산자가 적용되는 현세대의 개체군으로부터 최상의 값을 갖는 개체군을 선택한다.

단계 1: 이전세대와 현재 세대로부터 모든 개체군을 수집한다.

단계 2: 수집된 개체군을 순위에 따라 분류한다.

단계 3: 분류된 개체군에서 최상의 개체로부터 다음 세대를 만든다.

일반적인 GAs와 μGAs의 선택과정의 그림 7과 같다.

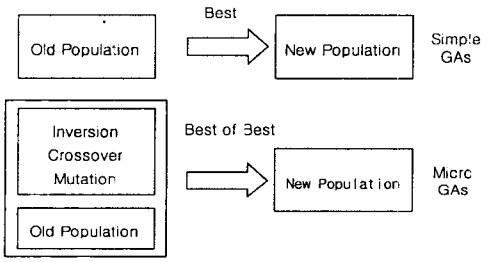


그림 7. GAs와 μGAs의 선택과정 비교

3.2 Inversion Module

이 모듈은 그림 8에서 보는 바와 같이 하나의 부모로부터 Randomize하게 선택된 유전자 배열의 일부분(3-2-6-4)을 떼어 역순으로(4-6-2-3)만들어 새로운 개체의 유전자를 형성한다.

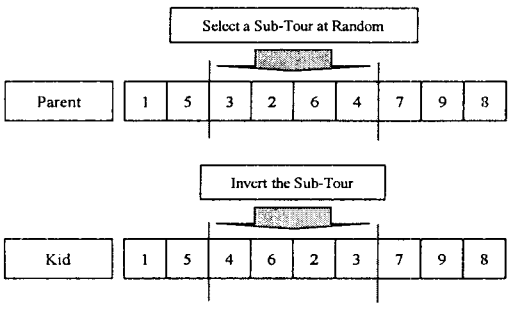


그림 8. Inversion Module의 설명

3.3 Crossover Module

순열조합문제에 가장 적절하게 적용되는 Order Crossover방법이 사용되었다. 이 방법은 우선 선택된 2개의 부모 유전자 배열의 일부분을 그대로 자식 유전자 배열에 이식한다. 그리고 한 부모 유전자의 배열을 보존된 유전자 위치의 뒤를 시점으로 재배열한 (8-7-5-4-2-1-6-9-3)후 다른 부모 유전자의 보존된 유전자에 있는 중복된 번호(9-7-4-8)를 제외시켜 새로운 유전자 배열(5-2-1-6-3)을 생성한다. 이렇게 생성된 배열은 자식의 비여 있는 유전자 위치에 이식한다. 자세한 연산과정은 그림 9에 묘사되었다.

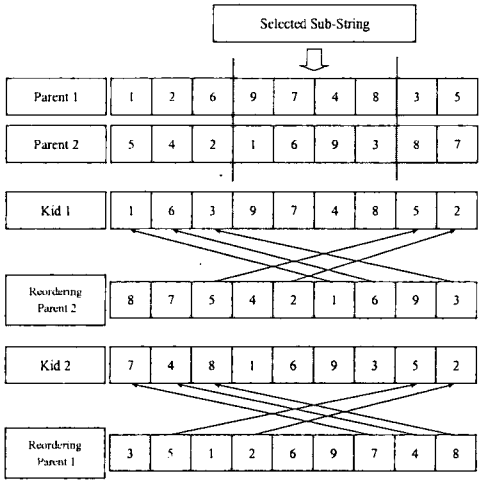
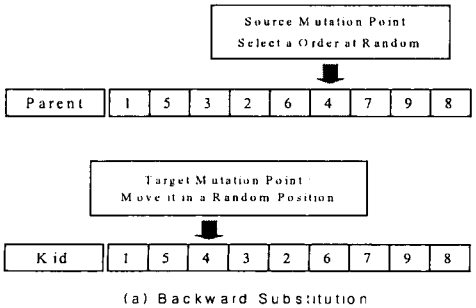


그림 9. Order Crossover Module의 설명

3.4 Mutation Module

유전자 배열중 임의의 한 점을 떼어내어 무작위로 선택된 다른 위치로 이동시켜 돌연변위된 유전자를 생성한다. 이때 선택된 위치에 따라 앞(Forward)이나 뒤(Backward)방향에 돌연변위가 수행된다[그림 10].



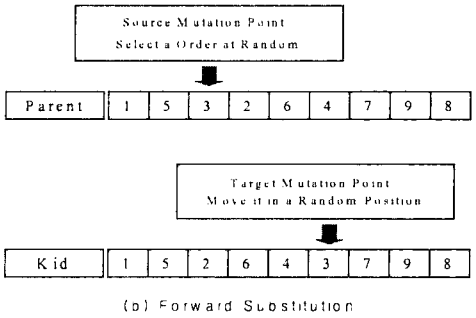


그림 10. 개성된 Mutation Module의 설명

4. Simulation 결과

그림 11과 같이 원재료를 나누어 분리시키고 다시 원재료의 같은 크기의 면적에 대입함으로서 이 알고리즘을 평가하고자 한다. 본 연구의 배치전략중 각도에 대한 처리부분은 부재가 놓여진 각도를 기준으로 4개의 각도(0° , 90° , 180° , 270°)만을 고려하여 수행하였다. 그림 12(a, b, c)의 결과는 기존의 보고된 결과를 바탕으로 참조하였다. 그림 13은 본 연구의 수행된 중간단계(0, 200, 350 세대수)와 최종결과이다.

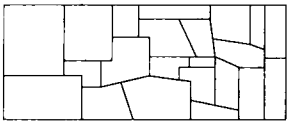
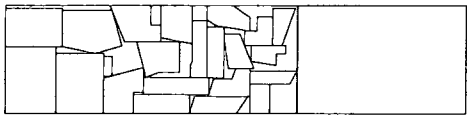
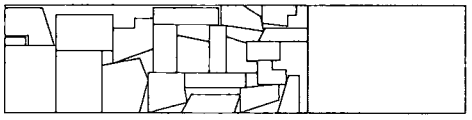


그림 11. Origin shape



(a) Result of the NestLib



(b) Result of the SigmaNEST

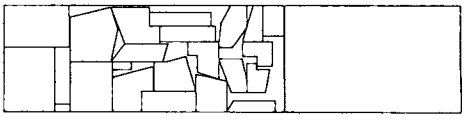
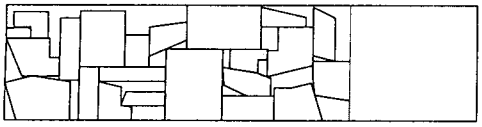
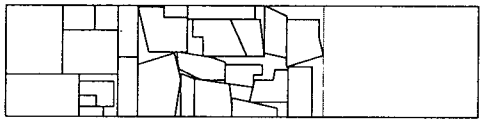


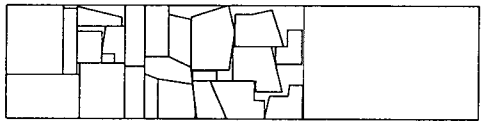
그림 12 (c) Result of the NestPlus[5]



Case 0 : Generation



Case : 200 Generation



Case 350 : Generation

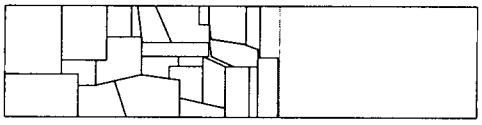


그림 13. Case 500 : Generation

평가의 기준을 삼기 위하여 표 2와 같은 Parameter Values를 사용하였다.

표 2. Parameter Value using μGAs

Parameter	Value
Crossover rate	0.4
Mutation rate	0.009
Inversion rate	0.009
Total Generation	100

표 3. 기존의 결과와 본 연구의 성과 비교

	Used Length	Waste ratio(%)
NestLib	483.67	18.21
SigmaNEST	520.13	21.22
NestPlus	468.05	15.50
Present	414.60	13.73

5. 결론

마이크로 유전자 알고리즘(Micro-GAs)은 배치의 효율을 증가시켜 최적의 효율을 얻는데 유용하게 이용될 수 있었다. 위의 Simulation의 결과로 최적에 가까운 배치효율로 만족스러운 결과를 얻었다 그러나 이러한 효율적인 배치 효과를 얻기 위하여 적용된 유전자 알고리즘의 수행 시간은 많이 소요(약 5시간) 되었다.

따라서, 앞으로의 연구과제는 보다 더 빠른 연산 시간 내에 최적에 근접한 결과를 도출시키기 위한 실용적 알고리즘을 구현하는 것이다.

[참고문헌]

[1] D.E. Goldberg, "Sizing Populations for Serial and Parallel Genetic Algorithms", In J. David Chaffer, editor, Proceedings of the third International Conference on Genetic Algorithms, San Mateo, California, Morgan Kaufmann Pub., pp. 70-79, 1989.

[2] K. Krishnakumar, "Micro-Genetic Algorithms for Stationary and Non-Stationary Function Optimization", In SPIE Proceedings: Intelligent Control and Adaptive Systems, Philadelphia, PA, Vol. 1196, pp. 289-296, 1989

[3] Carlos A. Coello Coello and Gregorio Toscano Pulido, "A Micro-Genetic Algorithm for Multiobjective Optimization", Lania-RI-2000-06, Laboratorio Nacional de Informatica Avanzada, 2000

[4] Yunyoung Kim, Masahiro Toyosada, Koji Gotoh, and Jewoong Park, "Air-Borne Selection in Micro-Genetic Algorithms for Combinatorial Optimisation", Inter. Conf. on Control, Automation and Systems(ICCAS 2001), Cheju National Univ., Korea, pp. 854-858, 2001.

[5] 한윤근, "임의 형상 부재의 자동 네스팅 시스템에 관한 연구", 서울대학교 공학박사 학위 논문, 2000.