# 분산 시뮬레이티드 어닐링을 이용한 복합 재료 재단
## (Composite Stock Cutting using Distributed Simulated Annealing)

홍 철 의 †
(Chul-Eui Hong)

**요 약** 복합 재료로 구성된 원판으로부터 여러 가지 패턴을 버려지는 부분이 최소화되게 배치시킨 후 절단하는 문제를 복합 재료 재단 문제라 부른다. 본 논문은 목적 함수의 비용 오류를 감내하는 영역 분할 분산 시뮬레이티드 어닐링 알고리즘을 MPI 환경하에서 복합 재료 재단 문제에 적용한다. 비용 오류 감내 기법은 최적해 접근 특성을 유지하기 위하여 스트림 길이를 동적으로 변화하며 상태변환을 비동기적으로 수행한다. 또한 복합 재료 재단 도구 개발을 위한 여러 가지 모양을 가진 패턴의 정보 및 친화도 생성, 목적함수, 상태변환 방법, 어닐링 스케쥴 및 이를 위한 효율적인 자료 구조에 대하여 정의한다. 배치될 패턴은 정형이나 convex 다각형으로 제한되어 있지 않고 어떠한 모양도 가능하며 원판은 복합 재료의 성격상 2 또는 4 방향으로 고정되어 있다.

**키워드** : 시뮬레이티드 어닐링, 재료재단, 분산처리, 최적화

***Abstract*** The composite stock cutting problem is to allocate rectangular and/or irregular patterns onto a large composite stock sheet of finite dimensions in such a way that the resulting scrap will be minimized. In this paper, the distributed simulated annealing with the new cost error tolerant spatial decomposition is applied to the composite stock cutting problem in MPI environments. The cost error tolerant scheme relaxes synchronization and chooses small perturbations on states asynchronously in a dynamically changed stream length to keep the convergence property of the sequential annealing. This paper proposes the efficient data structures for representation of patterns and their affinity relations and also shows how to determine move generations, annealing parameters, and a cost function. The spatial decomposition method is addressed in detail. This paper identifies that the final quality is not degraded with almost linear speedup. Composite stock shapes are not constrained to convex polygons or even regular shapes, but the rotations are only allowed to 2 or 4 due to its composite nature.

**Key words** : Simulated Annealing, stock cutting, optimazation

## 1. 서 론

시뮬레이티드 어닐링은 금속 응고 시의 어닐링 프로세스를 조합 최적화 문제에 적용한 알고리즘으로 Kirkpatrick 등에 의하여 처음 소개되었다[1]. 금속 응고에서의 어닐링은 금속을 융해점 근처의 높은 온도로 가열한 후 에너지의 평형 상태를 유지하면서 서서히 식히면 바닥 상태인 에너지가 가장 낮은 상태에 도달하게 된다. 시뮬레이티드 어닐링에서도 이와 마찬가지로 에너지에 대한 비용 함수를 정의한 다음 높은 온도에서 평형 상태를 유지하면서 서서히 온도를 낮추면 에너지가 높은 상태 즉, 비용이 높은 상태도 방문하면서 모든 가능한 상태를 고려하게 되므로 전역 최적해에 도달하게 된다<표 1>. 비용이 높은 상태로의 변환을 hill-climb 상태변환이라 한다.

표 1 금속 응고와 시뮬레이티드 어닐링

| 금속 응고 | 최적화 문제 |
|---|---|
| 결정 구조 | 상태(Configuration) |
| 에너지 | 비용 |
| 상 변태 | 상태변환(Move) |
| 바닥 상태 | 최적해 |
| 급속냉각(Quenching) | 순환향상 |
| 저속냉각(Annealing) | 시뮬레이티드 어닐링 |

· 본 연구는 2000학년도 상명대학교 자연과학연구소 연구비 지원으로 이루어졌음.
† 정 회 원 : 상명대학교 정보통신학부 교수
hongch@sangmyung.ac.kr
논문접수 : 2000년 8월 10일
심사완료 : 2001년 10월 15일

시뮬레이티드 어닐링은 조합 최적화 문제를 해결하는 데 순환향상 기법(iterative improvement)에서 발생하는 지역 최소값을 탈피하지 못하는 문제를 해결하여 비용 함수의 전역 또는 근사 전역 최소값에 도달하게 한다. 그러나 시뮬레이티드 어닐링은 Markov 프로세스로 최적해에 도달하는데 수행 시간이 오래 걸린다. 이를 해결하기 위하여 시뮬레이티드 어닐링 과정의 병렬화에 대한 많은 연구가 진행되어 왔다[2]. 시뮬레이티드 어닐링을 최적화 문제에 적용하기 위해서는 (1) 적합한 비용 함수 식, (2) 평형 상태를 유지하는 어닐링 스케줄, (3) 가능한 모든 상태를 방문할 수 있는 상태변환 구조, 마지막으로 (4) 수행속도를 빠르게 하기 위한 어닐링 프로세스의 병렬화가 요구된다.

산업에 사용되는 복합 재료는 방향성을 가진 얇은 복합체를 여러 장 겹쳐서 만든다. 복합 재료의 두께, 유리섬유의 방향성, 겹쳐진 판(plies or nests)의 개수 등이 강도 및 방향에 대한 무결성에 영향을 미친다. 겹치는 작업이 끝난 후 복합 재료에 포함된 공기 방울을 제거한 다음 오븐에서 소성 작업에 들어가서 복합 재료를 완성한다. 복합 재료는 특성상 방향성을 가지고 있으므로 주어진 패턴의 회전이 2 또는 4 방향으로 제한된다. 복합 재료는 일반적으로 고가이고 한 번에 많은 양의 판이 필요로 하므로, 여러 가지 모양을 가진 패턴을 최적으로 배치하는 작업은 원가 절감 및 생산성의 증가, 공정 시간의 단축에 필수적이다. 또한, 사각형 및 원과 같은 정형의 패턴만이 배치되는 것이 아니라 임의의 부정형의 패턴도 함께 배치되어야 하므로 이를 최적화하는 작업은 더욱 더 힘들어 진다. 복합 재료의 재단은 항공 및 철강, 자동차 등의 산업에서 중요한 기계적 공정이다. 복합 재료로 구성된 원판으로부터 여러 가지 패턴을 버려지는 부분이 최소화되게 배치시킨 후 절단하는 복합 재료 재단 문제는 방향성을 가진 일종의 2차원 2진 패킹 문제로 NP-Complete 군으로 분류된다[3]. 따라서, 본 논제를 주어진 시간 안에 해결할 다항식 알고리즘은 존재하지 않으므로 본 논문은 경험적 알고리즘의 하나인 시뮬레이티드 어닐링 기법을 사용하여 항상 전역 최적 값은 아니더라도 사용 가능한 배치를 가지는 근사 최적해를 찾는다.

제2절에서는 재료 재단 문제에 대한 기존의 연구를 정리한다. 제3절에서 복합 재료 재단문제의 자료구조를 정의하고, 제4절에서 시뮬레이티드 어닐링의 병렬화에 대하여 설명한다. 마지막으로 제5절에서 실험 결과를 분석한다.

## 2. 재료 재단 문제에 대한 연구 동향

일반적으로 선형 프로그래밍 방법은 최적화될 목적함수 및 목적 함수의 변수가 취할 수 있는 값의 범위를 제한하는 제약 조건에 대하여 수학적 모델을 개발하여 최적해를 구한다[4, 5]. 목적 함수 및 제약 조건은 독립변수를 갖는 선형 함수로서 정의된다. 그러나, 재료 재단 문제는 목적 함수의 변수 및 제약 조건이 수백에서 수천에 이르는 복잡한 수학적 모델로 나타낸다. 따라서, 특정한 구조를 갖는 문제에 대하여 최적해를 구할 수 있을 뿐 실제로 존재하는 다양한 종류의 문제에는 본 기법의 적용이 곤란하다.

동적 프로그래밍 기법은 경험적 알고리즘의 일종으로 문제를 먼저 모델링 한 후 그 모델을 일련의 상태변환 문제로 바꾼다. 이러한 변환은 궁극적으로 최적해는 첫 번째의 선택이 무엇이건 간에 다음의 선택은 앞선 선택의 결과에 대하여 최적인 선택을 내림으로써 얻어진다는 특성에 기초를 하고 있다. 문제는 최적의 선택을 어떻게 하면 빨리 내릴 수 있느냐 이다. 따라서, 동적 프로그래밍 기법은 모든 선택을 예시한 후 그 중에서 가장 최선의 값을 찾아내는 문제로 귀결될 수 있다. 이는 지수 함수의 복잡도를 가지고 있다[6].

다른 경험적 알고리즘으로 트리 탐색 기법은 모든 가능한 해를 트리 구조로 열거하여 트리를 효과적으로 탐색 함으로서 최적해를 구한다. 트리 탐색 알고리즘은 임의의 경로로부터 시작하여 최적해라 여기지는 해를 발견하였거나 본 경로가 최적해를 가지고 있지 않다고 판단될 때 종료한다. 트리 탐색 기법에서의 문제점은 처음에 경로를 어떻게 선정하며, 일단 경로가 선정되었을 때, 본 경로가 탐색할 가치가 있는지를 결정하여 탐색을 시도하던지 다른 경로로 바꿀지를 결정하는 문제이다[7]. 이를 이용한 branch-and-bound 기법은 [8]에 의하여 연구되었다.

Bottom-up Left-justified(BL) 알고리즘은 패킹할 패턴들의 순서를 리스트로 표현한 후 리스트에 나타난 패턴을 순서대로 하나씩 될 수 있는 한 원판의 가장 아래쪽 위치에 배치시킨 후 왼쪽으로 배치하는 작업을 반복한다(그림 1). 패킹할 패턴을 사각형으로 제한하면 하나의 사각형 당 2개의 방향이 존재하고 패턴의 개수를 $n$개로 정의하면 순서 리스트의 최대 한계는 $2^n n!$이며 BL 알고리즘을 사용하였을 때의 높이는 최적상태 높이의 3배 이하라는 것이 증명되어 있다[9].

BL 알고리즘의 순서 리스트는 모든 가능한 배치를 나타내므로 이를 유전자 알고리즘의 염색체로 보고 선택, 교배, 돌연변이의 유전 연산자를 적용하여 높이가 가장 적은 최적의 배치를 찾고자 하였다[10, 11, 12]. 그러나 BL 알고리즘은 패턴이 사각형으로 제한되어 있

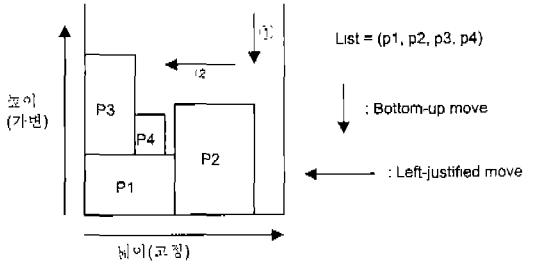


그림 1 BL 알고리즘의 예

으므로 임의의 다각형 패턴에는 적용할 수가 없다. 이러한 단점을 극복하기 위해서 다각형 패턴을 포함하는 최소 사각형을 만들어 최적의 순서 리스트를 유전자 알고리즘을 이용하여 찾아낸 후 각 패턴들 사이의 공간을 없애는 방법을 사용하였다[13]. 또한, [14]에서는 복합 재료 재단 문제에 순차 시뮬레이티드 어닐링을 적용하여 어닐링 스케줄 및 비용 함수, 상태변환 방법이 제시되었으나, 실제 산업 현장에서 요구되는 많은 수의 패턴을 배치하지는 못하였다.

이와 같이 위의 방법은 전역 최적 상태를 찾지 못하거나 현실적으로 다양한 응용 분야에 적용될 수 없기 때문에 본 논문에서는 영역 분할 분산 시뮬레이티드 어닐링 기법을 이용하여 항상 최적 배치는 아니더라도 현실적으로 받아들일 수 있는 근사 최적 배치를 찾는다.

## 3. 복합 재료 재단 문제의 자료 구조

복합 재료 재단 문제의 순서를 살펴보면, 어닐링 프로세스에 들어가기 전에, 각 패턴에 대하여 허용된 방향성을 고려하여 비트맵을 만든 다음 모든 한 쌍의 패턴에 대하여 친화도(affinity)를 미리 계산한다. 원판은 복합 재료로 구성되어 있기 때문에 각 판에 대해서는 0, 90, 180, 270도와 같은 미리 정해진 방향성이 존재한다. 비트맵은 두 패턴 사이의 중복을 쉽게 판별하기 위하여 사용하며 임의의 두 패턴에 대한 친화도는 두 패턴의 접근 정도에 따라서 패킹에 미치는 공헌도를 나타낸다.

### 3.1 패턴의 비트맵 정보 생성

여러 가지 모양을 가진 패턴을 방향성을 고려하여 효율적으로 두 패턴의 중복 및 친화도를 계산하기 위하여 비트맵을 생성한다. 비트맵 생성 시 임의의 패턴을 포함하는 최소 사각형을 bounding box또는 Minimum Bounding Rectangle(MBR)로 정의하여, bounding box 내에서 패턴의 내부 및 경계, 외부에 대한 비트 값을 각기 달리하여 임의의 두 패턴에 대한 친화도 및 중복을 손쉽게 계산한다(그림 2). Bounding box의 최소 xy좌표에 해당하는 비트를 해당 패턴의 기준좌표로 정한다.

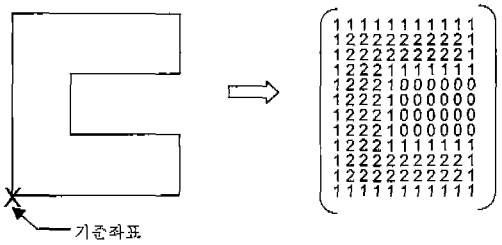


그림 2 임의의 패턴에 대한 비트맵의 예

### 3.2 친화도 생성

패킹 밀도는 두 패턴의 경계선이 될 수 있는 한 많이 겹칠수록, 두 개의 패턴을 포함하는 최소 사각형(bounding box)의 면적이 작을수록 높아질 가능성이 크므로 임의의 두 패턴 $i$와 $j$에 대한 친화도는 다음과 같이 정의된다.

$$a_{ij} = \alpha\left(a_1 + \frac{a_2}{2}\right) + \beta\rho \qquad (1)$$

$a_1$과 $a_2$는 두 패턴의 경계선이 겹쳐지는 정도를 나타내며 $\rho$는 두 패턴을 포함하는 최소 사각형의 밀도를 나타낸다(그림 3). $\alpha$와 $\beta$는 각 항의 가중치를 나타내며 음이 아닌 1보다 작은 수로 정의된다. $a_1$과 $a_2$는 비트맵 방식을 사용할 때 두 패턴의 경계선을 나타내는 비트가 겹쳐져 있는 수 및 거리 1만큼 떨어져 있는 비트의 수를 계산하여 얻는다. 패턴의 경계가 2이상 떨어져 있는 경우는 친화도에 영향을 미치지 않는다.

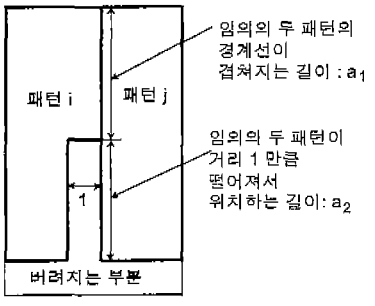


그림 3 친화도 계산

(그림 4)는 두 패턴의 경계가 겹치는 정도(edgewise adjacency)를 최대로 하는 것이 두 패턴을 포함하는 bounding box를 최소로 하지 못하는 경우를 보여 준다.

따라서, bounding box의 밀도($\rho$)를 친화도 계산에 포함시켜야 한다. 모든 패턴이 배치되었을 때 복합 재료 원판의 밀도는 두 패턴을 포함하는 bounding box의 넓이가 작을수록 커질 가능성이 높기 때문이다.

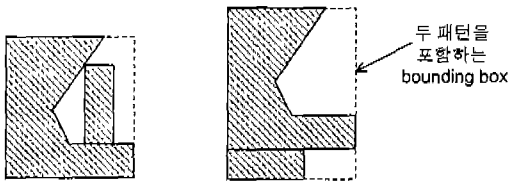


그림 4 Edgewise adjacency와 bounding box가 패킹에 미치는 영향

시뮬레이티드 어닐링은 일반적으로 시간이 오래 걸리므로 상태변환 및 비용 함수를 효율적으로 계산할 수 있는 quadtree나 pyramid와 같은 자료구조에 대한 연구가 앞으로 필요하다. 또한, 비용 함수 계산 시 친화도를 매 상태변환마다 계산한다면 많은 시간이 소요되므로 임의의 두 패턴의 허용된 각 방향에 대하여 친화도를 미리 계산하여 저장하고 있으면 실행 시간을 현저하게 줄일 수 있다.

### 3.3 패턴의 중복도 계산

시뮬레이티드 어닐링의 상태변환 시 두 패턴의 중복 정도를 효과적으로 계산할 수 있는 방법으로 비트맵을 사용하여 중복 정도를 효율적으로 알아 낼 수 있다(그림 5).

```
PROCEDURE FIND_OVERLAP
  begin
    Pick a pattern i and a position (x,y) randomly;
    for (all patterns j !=i) do
      if (bounding boxes of pattern i and j are
      overlapped)
        compute the number of overlap between
        boundary map of a pattern i and a whole
        map of a pattern j;
        compute the number of overlap between
        boundary map of a pattern j and a whole
        map of a pattern i;
  end
```

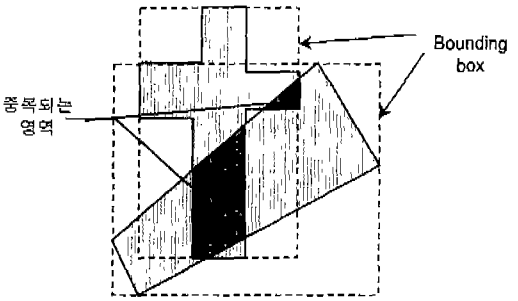


그림 5 비트맵을 이용한 두 패턴의 중복도 계산

## 4. 병렬 어닐링 스케줄

시뮬레이티드 어닐링은 일반적으로 광범위한 NP 문제에 적용이 가능하나, 문제 특성에 따라서 어닐링에 관계된 상태변환 방법, 비용 함수 계산, 냉각 스케줄 및 비용 함수 및 상태변환을 효율적으로 계산할 수 있는 자료 구조를 연구하여야 한다.

### 4.1 상태변환 방법

패턴의 새로운 장소로의 이동(displacement), 두 패턴의 위치 교환(exchange), 패턴의 회전(rotation)과 같은 3가지 상태변환 방법이 존재 할 수 있다.

4.1.1 이동(Displace Move)

이동 상태변환에서는 하나의 패턴을 임의로(랜덤 추출) 선택한 후 선택된 패턴의 기준 좌표를 허용된 이동 범위 안의 임의의 장소로 이동한다. 새로운 상태의 허용률에 따라서 이동 범위를 동적으로 조절하여 허용률을 40%~50%의 범위를 유지하게 하였다. 높은 온도에서는 이동성이 활발하게 보장되는 것이 좋으나, 낮은 온도로 갈수록 너무 광범위한 이동성은 큰 비용 변화를 초래하여 새로운 상태가 허용될 확률이 적어지므로 바람직하지 못하다. 또한, 패턴의 새로운 장소로의 이동 후 이동된 위치에서 허용된 모든 방향에 대하여 회전을 수행하여 패턴의 중복이 발생하지 않는 방향을 선택함으로써 상태변환 허용률을 높이도록 하였다.

4.1.2 위치 교환(Exchange Move)

위치 교환 상태변환에서는 두개의 패턴이 임의로 선택된 후, 두 패턴의 기준 좌표를 서로 교환한다. 두 패턴의 위치 교환은 높은 비용의 변화를 초래하므로 온도에 따라서 이동 대 교환 비율을 적절하게 조절하여야 한다. 이동 대 교환 비율은 높은 온도에서 낮은 온도로 갈수록 20:1에서 5:1로 변화 시키는 것이 새로운 상태에 대한 수락율이 잘 보장되었다.

4.1.3 회전(Rotate Move)

회전 상태변환에서는 하나의 패턴과 그 패턴에 적용할 회전 각의 크기를 임의로 선택한다. 복합 재료 재단 문제에서는 패턴의 방향성이 2 또는 4로 고정되어 있으므로 각 패턴에 따라 허용된 방향 각각에 대하여 비트맵을 미리 준비하여 회전 상태변환 수행 시간을 단축한다.

### 4.2 비용 함수

최소값을 최적해로 하는 비용 함수($C$)는 아래와 같이 정의 된다.

$$C = -\alpha \sum_{i} \sum_{j} a_{ij} + \beta \sum_{i} d_i + \gamma \sum_{i} \sum_{j} o_{ij} \qquad (2)$$

$\alpha$, $\beta$, $\gamma$는 각 항의 가중치를 나타낸다. 먼저, 식 (1)에서 구해진 친화도($a_{ij}$)의 항은 친화도가 높은 두 패턴이 근접할수록 최적해에 도달하므로 패턴 $i$와 $j$ 사이의 거리를 나타내는 $d_{ij}$에 반비례한다. 즉, 친화도항은 친화도가 높은 패턴들을 서로 이웃 하여 배치시키는 기능이 있다. 또한 $d_i$는 패턴 $i$가 원판의 기준점으로부터 떨어져 있는 거리를 나타내므로, 두 번째 항인 클러스터 항은 모든 패턴을 될 수 있는 한 원판의 기준점에 근접하도록 하여 모든 패턴을 포함하는 최소 사각형의 높이가 작아지도록 한다. 마지막으로, 현실적으로 수용 가능한 배치를 얻기 위해서는 모든 패턴은 중복되어서는 안되므로, 세 번째 항은 중복에 대한 페널티를 임의의 두 패턴의 중복 정도($o_{ij}$)에 대한 합으로 나타낸다. 세 번째 항의 가중치는 온도가 낮아질수록 커지거나, 최종 배치 시 겹쳐지는 부분이 없어지도록 하였다.

### 4.3 냉각 스케줄

냉각 스케줄은 문제의 특성에 따라 민감하게 작용하므로 복합 재료 문제에 대한 적절한 변수 선택은 중요하다. 시뮬레이티드 어닐링을 수행할 때 다음과 같은 변수가 정의된다.

- 초기 온도 : $T_0$
  초기 온도는 Markov 체인의 길이 $L_k$ 동안 상태변환 수락율이 90% 이상인 온도로 설정한다.
- 최종 온도 : $T_f$
  최종 온도는 5~10회의 온도 변화 동안 비용의 변화가 발생하지 않는 온도로 설정한다
- 임의의 온도 $k$에서의 Markov 체인 길이 : $L_k$
  Markov 체인의 길이는 입력 패턴의 개수의 4배 이상으로 설정한다.
- 온도 감소율 : $T_{k+1} = T_k$
  온도 감소율 $\alpha$는 0.95에서 0.98로 정의하였다.

최적해에 영향을 크게 미치는 임계 영역은 비용 오류 감내 알고리즘을 이용하여 찾아내었다.

### 4.4 비용 오류 감내 알고리즘

시뮬레이티드 어닐링 알고리즘은 전 단계에 대한 전역 상태 값이 필요한 Markov 프로세스로 정의되므로 분산 메모리를 가진 다중 컴퓨터 시스템 및 분산 환경에서 실행이 어렵다. 분산 환경에서 전역 상태를 유지하기 위해서는 상태변환 시마다 정보 전달이 필요하며, 이는 성능에 대한 병목 현상으로 작용한다. 이러한 병목 현상을 완화 시키는 방법으로는 각각의 프로세서가 상태변환을 동시에 여러 번 수행한 후 변화된 상태 값을 한 번에 다른 프로세스에 전달함으로써 정보 전달 횟수를 줄인다. 그러나, 이러한 방법은 임의의 상태에 대한 비용을 계산하는데 오류를 발생하며, 발생된 오류는 최적해의 도달을 방해한다. 그러나 시뮬레이티드 어닐링의 hill-climb 특성에 의하여 접근 특성이 유지되는 허용 가능한 오류 범위 내에서는 어닐링 스케줄의 능동적 조정 및 병렬 수행이 가능하여 순차 시뮬레이티드 어닐링을 사용한 때와 같은 품질의 최적해에 빠르게 도달할 수 있다.

비용 오류에 대한 한계를 전역 갱신 빈도 즉, 스트림 길이, $s$로 나타낼 수 있다. 스트림 길이란 모든 프로세서의 상태정보에 대한 전역 갱신 사이에 발생하는 각 프로세서의 비동기 상태변환의 개수를 말한다. 즉, 스트림 길이가 커지면 상태정보의 전역 갱신 사이에 비동기 상태변환이 많이 발생하므로 비용 오류가 커지며, 반대로 스트림 길이가 작으면 비용 오류도 작아진다. 비용 오류가 지수 분포를 가지고 있다는 것을 증명하여 스트림 길이 $s$에서의 비용 오류는 다음과 같이 정의되었다 [15].

$$E = (\Delta C_e \pm s \cdot \alpha \cdot |\langle E \rangle|) \cdot \Pr[\Delta C_e > 0] \cdot \left| e^{\frac{C}{T}} \cdot \left( e^{+\frac{s \cdot \alpha \cdot |\langle E \rangle|}{T}} - 1 \right) \right| \qquad (3)$$

여기서 $\Delta C_e$는 측정된 비용의 변화를 나타내며, $s$는 스트림 길이, $\alpha$는 상태변환에 대한 수락 율, $|\langle E \rangle|$는 하나의 수락된 상태변환에 대한 평균 비용 오류 값, $T$는 온도를 나타낸다.

비용이 높은 상태로의 상태변환 능력, 즉 hill-climb power는 비용 오류에 비례해서 감소하며, 감소된 hill-climb power를 보상하기 위해서 Markov 체인의 길이를 증가해야 한다. 따라서, 스트림 길이 $s$에 대하여 비용 오류가 발생하는 비동기 수행 시 비용 오류가 없는 순차 알고리즘과 같은 hill-climb power를 얻기 위해서 $su$의 스트림 길이가 필요하다.

$$e^{\frac{d_r(s)}{T}} \le \frac{1}{s} \cdot \frac{1}{u} \qquad (4)$$

여기서 $d_r(s)$은 스트림 길이 $s$를 가진 비용 오류가 없

는 시뮬레이티드 어닐링에서의 hill-climb power를 나타내며, $u$는 더 추가된 스트림 길이의 증가 분으로 여기서는 1보다 큰 수이다.

다음은 비용 오류 감내 알고리즘을 보여준다.

**PROCEDURE ERROR-TOLERANT SIMULATED ANNEALING**

begin

**INITIALIZE**; /* set the initial temperature $T_0$ */

$k$= 0;

repeat

**calculate** $|<E>|$ in $T_k$ /* Average cost error in one accepted move */

$E$ (s) = 0 /* $E$ (s) is the total amount of cost error in a given stream length */

repeat

**PERTURB**(config. $i$ config. $j, \Delta C_{ij}$);

if $\Delta C_{ij}$ <= 0 then accept

else

$E(s) = E(s) + C_{ij}$

$$E(s) = E(s) + \Delta C_{ij} \cdot e^{-\frac{C_{ij}}{T}\left(\frac{e^{1-\alpha\cdot|<E>|}}{T_k}-1\right)} \qquad (5)$$

/* $i$ is the $i^{th}$ iteration in the stream length */

/* is the acceptance ratio */

if exp($-\Delta C_{ij} / T_k$ ) > **random** [0,1] then accept;

if accept then

**UPDATE**(config. $j$);

until **equilibrium is approached sufficiently closely**; /* perturb states in $L_k$ times */

$T_{k+1} = \alpha \times T_k$; /* temperature decrement ratio=0.95 */

$k$= $k$+1;

if ( $E(s) \leq T_k \log u$ ) then /* $u$ = 1.05 */

**increase stream length**;

else

**decrease stream length**;

until **stop criterion == true (system is 'frozen')**;

/* final temperature $T_f$ is reached */

end.

### 4.5 영역 분할 분산 알고리즘

식 (2)의 비용 함수의 세 항 중 친화도에 대한 항만이 전역 상태를 요구하므로 비용 오류는 친화도에 대한 항에서만 발생하고 친화도는 두 패턴 사이의 거리에 반비례하므로, 가까이 위치한 패턴들을 같은 프로세서에 속하도록 패턴들이 속해 있는 영역에 따라 각 프로세서에 균등하게 분할함으로써 비용 함수 계산 시 발생하는 비용 오류를 최소화할 수 있다. 각 프로세서가 관할하는 영역의 $y$-방향의 길이는 원판의 넓이로 고정되어 있고, $x$-방향의 길이는 모든 패턴을 수용하는 원판의 길이를 프로세서 수로 나눈 길이로 영역을 할당한다(그림 6). 각 프로세서는 자신의 영역에 패턴의 기준 좌표가 속하는 패턴에 대하여 상태변환을 시도한다. 각 패턴의 기준 좌표는 그 패턴을 포함하는 bounding box의 최소 ($x$, $y$)-좌표 값으로 한다.

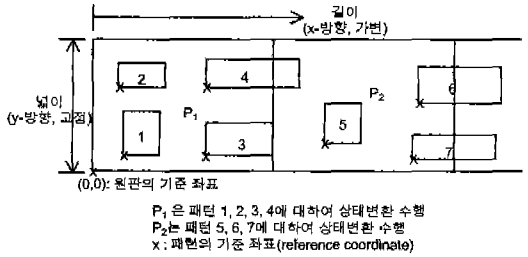


그림 6 영역 분할의 예

어닐링 프로세스가 시작하기 전 각 패턴들은 원판의 임의의 부분에 할당되나, 어닐링 프로세스가 진행됨에 따라 각 패턴들은 원판의 기준 좌표($x$-좌표 = 0) 부근으로 패킹 되어, 프로세서의 관할 영역은 점점 작아진다. 상태변환은 많아야 이웃한 2개의 프로세서만이 관여하도록 함으로서 프로그램을 간단하게 하고 프로세서의 비동기 수행을 원활하게 한다. 서로 협력하는 프로세서를 이웃한 2개 프로세서 이하로 제한하는 것은 패턴의 이동성을 제약하나 실험 결과 하나의 프로세서가 4개 이상의 패턴에 대하여 상태변환을 수행한 때 최적해 결과에 영향을 미치지 않는 것으로 나타났다.

서로 협력하는 프로세서를 이웃한 2개 프로세서로 제한하였으므로 패턴의 이동(displacement)은 프로세서내 이동과 프로세서간 이동으로 구별된다. 프로세서내 이동은 선택된 패턴의 전체 비트맵이 하나의 프로세서 영역 안에 존재하고, 이동 후에도 전체 비트맵이 같은 프로세서 영역 안에 존재한다. 따라서 이동에 대한 비용 변화는 해당 프로세서만이 계산하게 된다. 반면에 프로세서간 이동은 이동 전이나 이동 후에 선택된 패턴의 비트맵이 이웃한 2개의 프로세서 영역에 걸쳐 있을 때 발생하며 걸쳐있는 영역에 해당하는 2개의 프로세서는 비트맵의 중복에 의한 페널티 항을 정확하게 계산함으로써 비용 계산 시 오류를 최소화한다(그림 7).

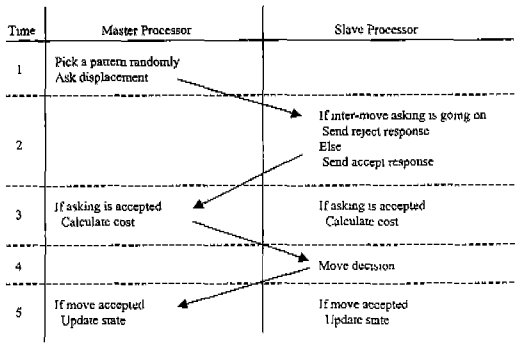


그림 7 프로세서간 이동

프로세서간 이동 시 주 프로세서는 자신의 영역에 속하는 임의의 패턴을 선택한 후 그 패턴의 이동 위치를 결정하고 이에 대한 정보를 해당되는 이웃 프로세서에 전달한다. 이 때 이웃 프로세서는 종속 프로세서로서 전달된 정보를 받아 이동의 진행을 결정한다. 이웃한 두 프로세서가 동시에 프로세서간 이동을 수행할 경우 상대방으로부터 메시지를 기다리는 데드락(deadlock) 상태에 빠지게 된다. 이러한 경우 종속 프로세서는 프로세서간 이동 요청을 거절하여 데드락을 해결한다.

이동에 대한 요청이 수락되면 주 프로세서는 클러스터항의 비용 변화와 주 프로세서 영역을 포함하여 주 프로세서 쪽에 위치한 모든 패턴의 친화도, 주 프로세서 영역 내의 패턴의 중복에 대한 페널티 비용을 계산하며, 종속 프로세서는 종속 프로세서 영역을 포함하여 종속 프로세서 방향에 위치한 패턴의 친화도 및 종속 프로세서 영역에서 발생하는 패턴의 중복에 대한 페널티 비용을 계산한다. 다음 종속 프로세서는 주 프로세서에서 계산된 비용을 합한 후 상태변환에 대한 수락 여부를 결정한다. 상태변환이 수락되면 새로운 상태로 갱신한다.

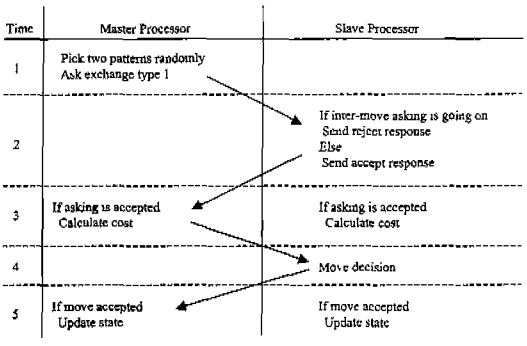


그림 8 프로세서간 교환 형태 1

패턴의 교환도 하나의 프로세서 영역 안에서 발생하느냐, 아니면 두 프로세서 영역에 걸쳐서 발생하느냐에 따라 프로세서내 교환(intra-processor exchange)과 프로세서간 교환(inter-processor exchange)으로 분류된다(그림 8, 9). 프로세서간 교환은 두 가지 형태가 존재할 수 있는데 첫번째 형태는 하나의 프로세서에서 교환될 두 패턴을 선택하였으나 교환 전 또는 후에 패턴의 위치가 이웃한 두 프로세서 영역에 걸쳐 나타나는 경우이고, 두 번째 프로세서간 교환 형태는 두 이웃한 프로세서에서 각각 하나의 패턴을 임의로 선택한 후 두 패턴을 교환하는 경우이다. 프로세서간 교환의 경우도 이웃한 2개의 프로세서만이 참여하도록 두 패턴의 선택을 제한한다.

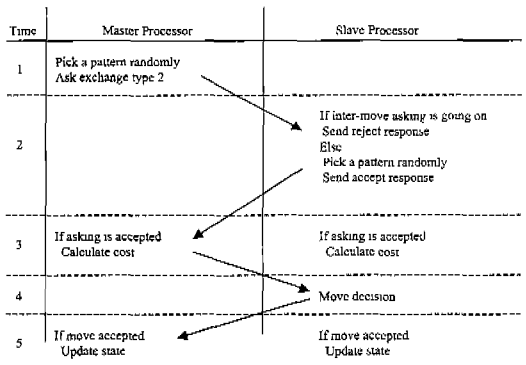


그림 9 프로세서간 교환 형태 2

각 온도마다 정해진 스트림 길이 동안 모든 프로세서는 다른 프로세서와는 무관하게 상태변환을 비동기적으로 수행한다. 비동기 상태변환의 회수가 늘어날수록 즉, 스트림 길이가 증가할수록 친화도항에 대한 비용오류 발생이 커진다. 따라서 스트림 길이만큼의 비동기 상태변환 후에 모든 패턴의 위치 정보에 대하여 전역 갱신을 시도한다.

상태변환에 참여하는 작업 프로세서들은 주어진 스트림 길이의 비동기 상태변환 수행 후, 상태변환 종료 메시지를 호스트 프로세서(프로세서 번호 0)로 전송한다. 호스트 프로세서는 모든 작업 프로세서로부터 상태변환 종료 메시지를 수령한 후 전역갱신 메시지를 모든 작업 프로세서에게 전달한다. 전역갱신 메시지를 받은 작업 프로세서들은 tree-reduction 알고리즘에 의하여 자신의 영역에 있는 패턴의 위치 정보를 다른 작업 프로세서들에게 전송한다. 호스트 프로세서는 각 프로세서에서 계산된 비용변화의 합을 실제 비용변화와 비교하여 식 (3)

에 의하여 비용 오류를 계산하고, 식 (4)에 의하여 새로운 스트림 길이를 계산한 후 이를 모든 작업 프로세서에 브로드캐스트한다.

## 5. 실험 결과

복합 재료 재단 문제를 위한 영역 분할 분산 시뮬레이티드 어닐링 알고리즘을 MPI 환경에서 실험하였다. 본 논문에 사용된 비용 오류 감내 분산 시뮬레이티드 어닐링 알고리즘은 각 온도에서 스트림 길이를 동적으로 조절하여 프로세서간 비동기적 수행으로 발생하는 비용 오류를 감내한다. 스트림 길이에 대한 계수는 5%로 고정하였다. 즉, 식 (4)에서 $u$=1.05로 하였다.

먼저 16개의 비정형 패턴을 4개의 프로세서에서 Markov 체인 길이는 500으로 고정하여 순차 알고리즘과 분산 알고리즘을 비교하였다. 식 (2)의 비용 오류는 친화도항에서만 발생하므로 비용 오류가 어닐링 알고리즘에 미치는 영향을 알아보기 위하여 본 실험에서는 친화도항에 대한 가중치를 클러스터 항에 대한 가중치보다 높게 정하였다. <표 2>에서 보듯이 분산 알고리즘과 순차 알고리즘의 최종 결과는 비슷하게 나타났다.

표 2 분산 및 순차 알고리즘의 최종 비용 값

| | 평균 | 표준편차 | 최악 | 최선 |
|---|---|---|---|---|
| 분산 어닐링 | -423.3 | 8.16 | -423.1 | -436.2 |
| 순차 어닐링 | -425.6 | 7.94 | -417.8 | -436.2 |

(그림 10)은 각 온도에서의 어닐링 곡선과 스트림 길이를 나타낸다. 임계 영역에서의 스트림 길이는 2까지 줄어 들었으나, 고온 및 저온 부위에서는 125까지 증가하였다. 즉, 어닐링 곡선의 변화에 따라서 최적해 접근 특성을 유지하면서 스트림 길이가 동적으로 변화하는 것을 알 수 있다. 이러한 사실은 임계영역에서는 비용 오류가 최적해 접근 특성에 미치는 영향이 크나, 그 이외의 부분에서는 영향이 적게 미치는 것을 말해주고 있다. 또한 이러한 사실은 임계영역 이외의 부분에서 어닐링 프로세스를 빠르게 진행할 수 있으나, 임계영역에서는 어닐링 프로세스를 천천히 진행하여야 최적해에 접근한다는 사실과도 부합한다.

두 번째 실험에서는 패턴의 수를 16개에서 160개까지 증가하였으며, 초기 온도는 200,000으로, 온도 감소율은 0.98에서 0.99, Markov 체인 길이는 5,000에서 20,000까지 변화 하였다. 클러스터 항의 가중치를 친화도항의 가중치와 같게 하여 최적의 최종 결과를 얻도록 하였다.


그림 10 어닐링 곡선에 따른 스트림 길이의 동적 변화

또한, 프로세서 수 변화에 대한 수행 시간의 향상(speedup)을 (그림 11)에 표시하였다. <표 3>은 프로세서 수에 따른 최종 비용을 보여준다.

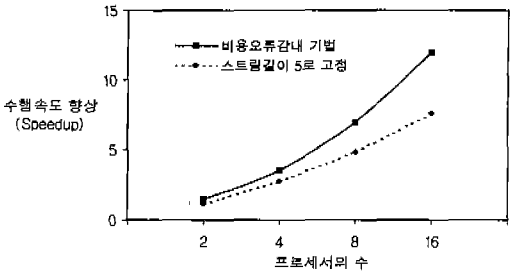


그림 11 분산 기법의 수행속도 향상

표 3 프로세서 수에 따른 최종 비용

| 프로세서 수 | 평균 | 표준편차 |
|---|---|---|
| 1(순차) | 583848 | 8469 |
| 2 | 581401 | 7973 |
| 4 | 579776 | 7840 |
| 8 | 586483 | 7661 |
| 16 | 583251 | 7496 |

세 번째 실험에서는 스트림 길이에 대한 비용 오류의 특성을 알아보기 위하여 스트림 길이를 변화 시켜 보았다. 본 실험은 16개의 프로세서에 대하여 128개의 비정형 패턴을 사용하였으며 각 항목 당 5번씩 수행하여 평균을 구하였다. <표 4>에서 알 수 있듯이 스트림 길이가 증가함에 따라 수행시간은 감소하나 최종 비용은 증가하여 최적해에 도달하지 않는 것으로 나타났다. 반면에 본 논문에서 사용한 스트림 길이를 동적으로 조절하는 비용 오류 감내 기법은 짧은 시간 안에 최적의 결과를 보여 주었다. 즉, 스트림 길이가 고정된 종래의 기법

표 4 스트림 길이를 고정하는 기법과 비용 오류 감내 기법의 성능 비교

| 스트림 길이 | 수행시간(msec) | 비용 |
|---|---|---|
| 625 | 6.96378e+06 | 797019 |
| 208 | 7.05910e+06 | 791015 |
| 125 | 7.31633e+06 | 790672 |
| 89 | 7.38138e+06 | 784053 |
| 69 | 7.48652e+06 | 794623 |
| 57 | 7.18042e+06 | 791005 |
| 45 | 7.52592e+06 | 787917 |
| 35 | 7.83123e+06 | 786564 |
| 26 | 8.44653e+06 | 784167 |
| 21 | 8.56733e+06 | 787604 |
| 16 | 9.43514e+06 | 788615 |
| 10 | 1.09276e+07 | 788676 |
| 5 | 1.74531e+07 | 783721 |
| 평균 | 8.73649e+06 | 78896 |
| 오류감내기법 | 7.20755e+06 | 786792 |
| 순차기법 | 9.34506e+07 | 783421 |

에 비하여 향상된 최적 값을 유지하면서 6.3배의 전역 갱신을 절약하는 효과를 보였다. 수행시간 향상(speedup) 측면에서는 스트림 길이가 고정된 기법이 10.6이고 새로운 비용 오류 감내 기법이 12.9로 좋은 결과를 얻었다.

(그림 12)는 복합 재료 재단 문제에 대한 실제 예를 보여 준다. 대부분의 실제 문제에서 정확한 전역 최적해를 알 수 없으므로, <표 3>에서 보듯이 여러 가지 예에 대한 오류감내 기법을 이용한 분산 어닐링의 패킹 밀도는 76%에서 91%로 순차 어닐링에서의 결과와 비슷하게 나타났다.

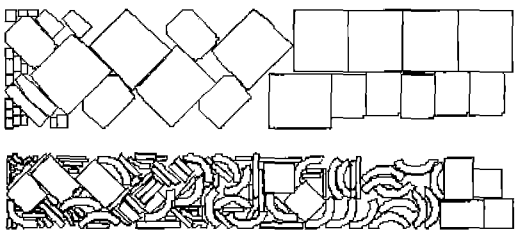

그림 12 복합 재료 재단 문제의 예

## 6. 결 론

본 논문에서는 복합 재료 재단에 관한 문제 정의 및 이에 대한 분산 어닐링 스케줄에 대하여 기술하였다. 복합 재료 재단 문제에 대한 도구 설계 시 고려하여 할 사항을 체계적으로 접근하였으며, 복합 재료 재단 문제에 특성화된 어닐링 스케줄을 연구함으로써 새로운 응용 분야에 대한 시뮬레이티드 어닐링의 적용 가능성을 타진하였다. 실험 결과 시뮬레이티드 어닐링을 이용한 복합 재료의 패킹 밀도는 어닐링의 온도를 0으로 하여 hill-climb 상태변환을 허용하지 않는 순환 향상 기법에 비하여 평균 10% 정도의 향상을 보였다.

본 논문에서 사용한 비용 오류 감내 기법은 speedup 측면에서의 이익과 함께 고정된 스트림 기법에 비하여 최적 스트림을 찾기 위한 수많은 실험이 필요하지 않다. 즉, 고정된 스트림 기법에서는 최적의 스트림 개수를 찾기 위해서 많은 수의 전 단계 실행이 필요하나, 제안한 비용 오류 감내 기법은 고온과 저온 부분에서는 스트림 길이가 크며, 임계 영역 부분에서는 스트림 길이가 줄어서, 온도에 따라 스트림 길이가 동적으로 변화됨을 볼 수 있다.

앞으로, 임계 영역에 대한 자동 탐지 문제, 온도 감소율이 최적 값에 미치는 영향을 조사하여 어떠한 종류의 재단 문제에 대해서도 자동적으로 어닐링 스케줄을 조정할 수 있게 한다. 또한, 시뮬레이티드 어닐링의 단점인 입력에 따른 어닐링 변수의 변화에 대한 문제를 해결하여 시뮬레이티드 어닐링을 실생활에 사용하도록 한다.

## 참 고 문 헌

[1] Kirkpatrick, S., Gellatt Jr. C. D., and Vecci, M. P., Optimization by simulated annealing, *Science*, Vol.220, pp. 975-986, 1983.

[2] Lee, S. and Lee, K., Synchronous and asynchronous parallel simulated annealing with multiple Markov chains, *IEEE Trans. on Parallel and Distributed Systems*, Vol.7, No.10, pp. 993-1008, 1996.

[3] Fowler, R.J., Paterson, M.S., and Tanimoto, S.L., Optimal packing and covering in the plane are NP-complete, *Information Processing Letters*, Vol.12, No.3, pp. 133-137, 1981.

[4] Balas, E., Ceria, S., Cornuejols, G., and Natraj, N., Gomery cut revisited, Operations *Research Letters* 19, pp. 1-9, 1996.

[5] Degraeve, Z. and Vandebroek, M., A mixed integer programming model for solving a layout problem in the fashion industry, *Management Science* 44, pp. 301-310, 1998.

[6] Stoyan, Y.G., Novozhilova, M.V., and Kartashov, Mathematical model and method of searching for a local extremum for the non-convex oriented polygons allocation problem, *European Journal of*

*Operational Research* 92, pp. 193-210, 1996.
[7] Beasley, J.E. An exact two-dimensional non-guillotine cutting tree search procedure, *Operations Research*, Vol.33, No.1, pp. 49-64, 1985.
[8] Morabito, R., and Arenales, M.N., Staged and constrained two-dimensional guillotine cutting problems: An AND/OR-graph approach, *European Journal of Operational Research* 94, pp. 548-560. 1996.
[9] Baker, B.S., Coffman, E.G., and Rivest, R.L., Orthogonal packings in two dimensions, *SIAM Journal on Computing*, Vol.9, No.4, pp. 846-855, 1980.
[10] Bean, J.C., A multiple-choice genetic algorithm for a nonlinear cutting stock problem, *Computing in Science and Engineering*, Vol.2, No.2, pp. 80-83, 2000.
[11] Dagli, C.H., and Poshyanonda, P., New approaches to nesting rectangular patterns, *Journal of Intelligent Manufacturing*, Vol.8, pp. 177-190, 1997.
[12] Petridis V., Kazarlis, S., and Bakirtzis, A., Varying fitness functions in genetic algorithm constrained optimization The cutting stock and unit commitment problems, *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol.28, No.5, pp. 629-640, 1998.
[13] Jakobs, S., On genetic algorithms for the packing polygons, *European Journal of Operational Research* 88, pp. 165-181, 1996
[14] Lutfiyya, IL, B. McMillin, P. Poshyanonda, and C. Dagli, Composite stock cutting through simulated anncaling, *Mathl. Comput. Modeling*, Vol.16, No.1, pp. 57-74, 1992.
[15] 홍철의, 김영준, 시뮬레이티드 어닐링에서의 비용오류 측정 및 분석, *한국정보처리학회 논문지*, 제7권 제4호, pp. 1141-1149, 2000.

홍 철 의
1985년 한양대학교 금속공학과 졸업(학사). 1989년 미국 New Jersey Institute of Technology 전산학(석사). 1992년 미국 Univ. of Missouri-Rolla 전산학(박사). 1992년 ~ 1997년 한국전자통신연구원 선임연구원. 1997년 ~ 현재 상명대학교 정보통신학부 조교수. 관심분야는 분산 및 병렬 알고리즘, 최적화 기법