

# 개수 제한이 없는 2차원 절단문제를 위한 향상된 최적해법

지영근<sup>1</sup> · 강맹규<sup>2\*</sup>

<sup>1</sup>한양대학교 첨단도로연구센터 / <sup>2</sup>한양대학교 산업공학과

## An Improved Exact Algorithm for the Unconstrained Two-Dimensional Cutting Problem

Young-Gun G<sup>1</sup> · Maing-Kyu Kang<sup>2</sup>

<sup>1</sup>Advanced Highway Research Center, Hanyang University, Ansan, 425-791

<sup>2</sup>Department of Industrial Engineering, Hanyang University, Ansan, 425-791

This paper is concerned with the unconstrained two-dimensional cutting problem of cutting small rectangles (products), each of which has its own profit and size, from a large rectangle (material) to maximize the profit-sum of products. Since this problem is used as a sub-problem to generate a cutting pattern in the algorithms for the two-dimensional cutting stock problem, most of researches for the two-dimensional cutting stock problem have been concentrated on solving this sub-problem more efficiently. This paper improves Hifi and Zissimopoulos's recursive algorithm, which is known as the most efficient exact algorithm, by applying newly proposed upper bound and searching strategy. The experimental results show that the proposed algorithm has been improved significantly in the computational amount of time as compared with the Hifi and Zissimopoulos's algorithm.

**Keywords:** optimization, two-dimensional, cutting, packing

### 1. 서론

개수 제한이 없는 2차원 절단문제(unconstrained two-dimensional cutting problem; UTDC)는 하나의 큰 직사각형(자재)으로부터 서로 다른 가치를 갖는 여러 크기의 방향(orientation)이 고정된 작은 직사각형(제품)들을 가치의 합이 최대가 되도록 잘라내는 문제이다. 이때 잘라낸 각 제품의 개수에 제한이 없을 경우 이를 '개수 제한이 없는(unconstrained)' 문제라 하고, 각 제품별로 잘라낼 수 있는 개수에 제한이 있는 경우를 '개수 제한이 있는(constrained)' 2차원 절단문제(CTDC)라 한다. 일반적으로 두 종류의 2차원 절단문제 모두 2차원 자재절단문제(cutting stock problem; CSP)에서 각 절단패턴을 생성하기 위해 이용될 수 있다(Haessler and Sweeney, 1991).

본 논문에서 다루는 개수 제한이 없는 2차원 절단문제(TDC)를 이해하기 위해 먼저, 자재절단문제(CSP)를 이해할 필요가 있다. 자재절단문제란 일정한 규격을 갖는 하나 이상의 자재들로부터 서로 다른 크기의 작은 직사각형의 제품들을 잘라내는 문제로 각 제품들의 요구 수량을 만족시키면서 소요되는 자재 비용을 최소화하는 데 목적이 있다(Gilmore and Gomory, 1961). 따라서, 2차원 자재절단문제의 해는 여러 종류의 절단패턴들의 집합으로 구해지는데, 각각의 절단패턴들을 구하기 위해 2차원 절단문제를 풀어야 한다.

자재절단문제(CSP)를 위한 해법들은 크게 Gilmore와 Gomory (1961)의 선형계획적 접근법(linear programming approach)과 그 이외의 발견적 해법(heuristic algorithm)으로 나눌 수 있는데, 전자의 경우는 각각의 절단패턴을 생성하기 위하여 개수 제한이 없는 절단문제(UTDC)를 풀면 되지만, 후자의 경우는 개수 제

본 논문은 한국과학기술부, 한국과학재단에서 지원한 첨단도로연구센터의 연구수행 결과임.

\* 연락저자: 강맹규 교수, 425-791 경기도 안산시 사1동 1271 한양대학교 산업공학과, Fax : 031-409-2423, e-mail : dockang@logistex.hanyang.ac.kr  
2001년 4월 접수, 1회 수정 후 2001년 11월 게재 확정.

한이 있는 절단문제(CTDC)를 풀어야 한다 (Haessler and Sweeney, 1991). Gilmore와 Gomory의 선형계획적 접근법은 해법의 수학적 특성 때문에 최적해에 근사한 해를 구할 수 있다는 장점이 있으나, 대단히 많은 반복 주기마다 절단문제를 풀어야 하므로 절단문제의 해법이 매우 효율적이어야 한다. 지금까지 개수 제한이 없는 2차원 절단문제를 위한 몇몇 최적해법들이 개발되어 왔으나, 모두 선형계획적 접근법에 적용하기에는 비효율적이다 (Gilmore and Gomory, 1966; Beasley, 1985; Herz, 1972; Morabito *et al.*, 1992; Hifi and Zissimopoulos, 1996). 따라서, 개수 제한이 없는 2차원 절단문제(UTDC)의 최적해법을 효율적으로 만드는 것은, 2차원 자재절단문제의 선형계획적 접근법을 실제 문제에 적용하기 위해 대단히 중요하다.

본 논문은 개수 제한이 없는 이차원 절단문제(UTDC)의 해법에 이용할 수 있는 매우 효율적인 상한을 먼저 제안하고, 이러한 절단문제의 최적해법들 중 가장 효율적이라고 알려진 Hifi와 Zissimopoulos의 순환적 해법(recursive algorithm)에 적용하고 또한 이들의 탐색전략을 개선함으로써 해법의 성능을 향상시킨다. 끝으로, 실험을 통해 개선된 해법이 기존의 해법보다 매우 빠른 시간 내에 최적해를 구할 수 있음을 보인다.

## 2. 기존 해법

지금까지 개수 제한이 없는 2차원 절단문제를 위한 몇몇 최적해법들이 개발되어 왔다. 그 중 일부는 동적계획법(dynamic programming)을 적용한 것들이고, 일부는 트리 탐색법(tree search approach)을 적용한 것이다. 이 중 가장 우수하다고 알려진 것은 Hifi와 Zissimopoulos(1996)의 순환적 해법인데, 본 논문에서는 이 해법을 더욱 개선한 해법을 제안한다.

이 절에서는 개선한 해법을 제안하기 앞서, Hifi와 Zissimopoulos의 해법을 정리한다. Hifi와 Zissimopoulos의 해법은 트리 탐색법에 기반을 둔 해법으로, 트리 탐색을 위해 함수를 재귀 호출하는 방법을 사용한다.  $(x, y)$ 를 가로 길이  $x$ 와 세로 길이  $y$ 를 갖는 직사각형이라 하자. 자재의 크기가  $(l_i, w_i)$ 이고 제품  $i$ 의 크기와 가치가 각각  $(l_i, w_i)$ 와  $p_i$ 인  $n$ 종류의 제품을 갖는 개수 제한이 없는 2차원 절단문제를 정의하자.  $F(a, b)$ 를 자재조각  $(a, b)$ 에 대해 최적해를 구하는 함수라 하면,  $F(a, b)$ 는 다음과 같은 재귀함수이다(여기서  $\lfloor x \rfloor$ 는  $x$ 를 넘지 않는 최대 정수).

$$F(a, b) = \max \left\{ F(x, b) + F(a-x, b), \quad x = 1, \dots, \lfloor \frac{a}{2} \rfloor; \right. \\ \left. F(a, y) + F(a, b-y), \quad y = 1, \dots, \lfloor \frac{b}{2} \rfloor \right\}$$

$x$ 와  $y$ 를 각각  $\lfloor \frac{a}{2} \rfloor, \lfloor \frac{b}{2} \rfloor$ 까지만 변화시키는 이유는 자재조각이 둘로 절단될 때, 첫 번째 자재조각을 두 번째 자재조각보다 항상 작거나 같게 유지시킴으로써 불필요한 중복 탐색을 줄이기 위함이다. 다음의 집합  $P_{ab}$ 와  $Q_{ab}$ 는 각각 자재조각  $(a, b)$ 보다 작은 제품들의 가로 길이와 세로 길이 선형결합

으로 구할 수 있는 정수들의 집합이다(여기서  $N$ 은 비음의 정수 집합).

$$P_{ab} = \left\{ x : x = \sum_{i=1}^n l_i z_i \leq a, \quad z_i \in N, \quad w_i \leq b \right\} \\ Q_{ab} = \left\{ y : y = \sum_{i=1}^n w_i z_i \leq b, \quad z_i \in N, \quad l_i \leq a \right\}$$

Hifi와 Zissimopoulos는 탐색을 더욱 효율적으로 하기 위해, 집합  $P_{ab}$ 와  $Q_{ab}$ 에 포함되는  $x$ 와  $y$ 에 대해 자재조각을 절단한다. 이렇게 생성된 절단 패턴들을 정규화된(normalized) 패턴이라 한다(Herz, 1972; Christofides and Whitlock, 1977). 따라서, 함수  $F(a, b)$ 는 다음과 같이 재정의할 수 있다.

$$F(a, b) = \max \{ F(x, b) + F(a-x, b), \quad \forall x \in P_{\lfloor \frac{a}{2} \rfloor, b}; \\ F(a, y) + F(a, b-y), \quad \forall y \in Q_{a, \lfloor \frac{b}{2} \rfloor} \}$$

불필요한 탐색 과정을 줄이기 위해 Hifi와 Zissimopoulos는 하한(lower bound)과 상한을 구하여 이용할 것을 제안하였다. 하한은 Fayard와 Zissimopoulos(1995)의 발견적 해법(heuristic algorithm)을 사용하여 구하는데, 이 해법은 두 번에 걸쳐 배낭문제를 풀어 2단계 절단문제(two-stage cutting problem)의 해를 구하는 것이다(Gilmore and Gomory, 1965). 자재조각  $(a, b)$ 의 하한을 구하기 위해, 서로 다른 값을 갖는 제품의 세로 길이들을  $w_1 \leq w_2 \leq \dots \leq w_r$  ( $r \leq n$ )가 만족되도록 정렬한다. 그리고 다음의 배낭문제를 풀어 세로 길이  $i$ 에 대한 수평 띠조각 패턴을 구한다.

$$f_i(a) = \max \left\{ \sum_{j \in R_{ab}} p_j x_j : \sum_{j \in R_{ab}} l_j x_j \leq a, \quad x_j \in N \right\} \\ R_{ab} = \{ j : l_j \leq a, w_j \leq b \text{ for } j = 1, 2, \dots, n \}$$

그리고 이 값들을 이용하여 수평 방향의 2단계 절단문제의 해를 다음과 같은 배낭문제를 풀어 구한다.

$$L(a, b) = \max \left\{ \sum_{i=1}^r f_i(a) y_i : \sum_{i=1}^r w_i y_i \leq b, \quad y_i \in N \right\}$$

해법의 초기 단계에서 자재  $(l_i, w_i)$ 에 대한 하한은 수평 방향의 2단계 절단문제의 해  $L(l_i, w_i)$ 와 수직 방향의 2단계 절단문제의 해 중 큰 값을 취한다.

크기가  $(a, b)$ 인 자재조각( $a \leq L, b \leq W$ )의 상한을 구하기 위해 Hifi와 Zissimopoulos는 다음과 같은 수리모형을 사용하였다.

$$U(a, b) = \text{Maximize } \sum_{i \in R_{ab}} p_i x_i \\ \text{subject to } \sum_{i \in R_{ab}} (l_i w_i) x_i \leq ab \\ x_i \leq \lfloor \frac{a}{l_i} \rfloor \times \lfloor \frac{b}{w_i} \rfloor \\ x_i \geq 0, i \in R_{ab}$$

Hifi와 Zissimopoulos의 상한은 자재조각  $(a, b)$ 에 포함된 제품들의 면적의 합이 자재조각의 면적을 넘지 못한다는 제약하

에 제품 가치의 합을 최대로 하는 배낭문제에서 변수  $x_i$ 의 정수 제약을 제거한 것이다. 정수 제약을 제거한 이유는 해법의 탐색 과정에서 상한을 매우 빈번히 구해야 하는데, 그때마다 집합  $R_{ab}$ 를 구하여 배낭문제를 푸는 것은 해법의 계산량에 큰 부담이 되기 때문이다. 그러므로 정수 제약을 제거한 위의 수리모형을 빠른 시간 내에 풀 후, 그 결과를 정수화한  $\lfloor U(a, b) \rfloor$ 를 상한으로 사용하였다. 다음은 이상 설명한 하한과 상한을 이용한 Hifi와 Zissimopoulos의 해법이다. [Step 2]의 재귀 함수에서 세 번째 전달인자  $v_0$ 는 해를 구하고자 하는 자재조각  $(a, b)$ 의 하한(또는 목표값)을 의미한다. 따라서, 상한이 하한(또는 목표값)보다 작거나 같으면 자재조각  $(a, b)$ 의 해를 구할 필요가 없다.

Step 1. [initialize]

Construct the sets  $P_{l_0, w_0}$  and  $Q_{l_0, w_0}$ .  
 Set  $v_0 = L(l_0, w_0)$ .  
 Set  $Opt(x, y) = -1$  for  $1 \leq x \leq l_0, 1 \leq y \leq w_0$ .  
 $Opt(l_0, w_0) \leftarrow F(l_0, w_0, v_0)$  by calling the recursive function in step 2.  
 Go to step 3.

Step 2. [ $F(a, b, v_0)$ ]

$f \leftarrow U(a, b)$   
 If  $v_0 \geq f$  then  
     Return  $-1$ .  
 End if  
 Set  $a_0 = \max\{x: x \leq a, x \in P_{ab}\}$ .  
 Set  $b_0 = \max\{y: y \leq b, y \in Q_{ab}\}$ .  
 If  $Opt(a_0, b_0) \geq 0$  then  
     Return  $Opt(a_0, b_0)$ .  
 End if  
 $V \leftarrow L(a_0, b_0)$   
 If  $V \geq U(a_0, b_0)$  then  
     Return  $V$ .  
 End if  
 If  $\min\{w_i: l_i \leq a_0\} > b_0/2$  or  $\min\{l_i: w_i \leq b_0\} > a_0/2$  then  
     Return  $V$ .  
 End if  
 For all  $x \in R_{\frac{a}{2}, b}$  (ii)  
     If  $Opt(x, b_0) < 0$  then  
          $Opt(x, b_0) \leftarrow F(x, b_0, \max(V, v_0) - U(a_0 - x, b_0))$  (iii)  
 End if  
 If  $Opt(x, b_0) + U(a_0 - x, b_0) > V$  then  
     If  $Opt(a_0 - x, b_0) < 0$  then  
          $Opt(a_0 - x, b_0, \max(V, v_0) - Opt(x, b_0))$  (iv)  
 End if

$v_1 \leftarrow \max(0, Opt(x, b_0)) + \max(0, Opt(a_0 - x, b_0))$   
 If  $v_1 \geq U(a_0, b_0)$  then  
     Return  $v_1$ .  
 End if  
 If  $v_1 \geq V$  then  
      $V \leftarrow v_1$ .  
 End if  
 End if  
 End for  
 Repeat for all  $y \in Q_{\frac{a}{2}, b}$ . (v)  
 Return  $V$ .  
 Step 3. [finish]  
 Exit with  $Opt(l_0, w_0)$ .

3. 새로운 상한

Hifi와 Zissimopoulos의 상한은 단지 제품들의 면적만을 고려하기 때문에 우수한 상한을 구하지 못한다. 따라서, 이 절에서는 제품의 면적뿐만 아니라 기하학적 배치까지 고려할 수 있는 새로운 상한을 제안한다.

크기가  $(a, b)$ 인 자재조각에 대해, <그림 1>과 같이 자재의 아래 모서리에서부터  $j-1$ 만큼 떨어진 크기  $(a, 1)$ 인 자재조각을 수평띠  $j$ 라 하고  $x_{ij}$ 를 수평띠  $j$ 에 출현한 제품  $i$ 의 개수라 정의하면, 자재조각  $(a, b)$ 에 대한 2차원 절단문제는 다음과 같은 수리모형을 갖는다.

$$\begin{aligned} \text{Maximize } z &= \sum_{j=1}^b \sum_{i \in R_{ab}} \left( \frac{p_i}{w_i} \right) x_{ij} \\ \text{subject to } \sum_{i \in R_{ab}} l_i x_{ij} &\leq a \text{ for } \forall j \\ &\dots \\ \forall x_{ij} &\in \mathbb{N} \end{aligned} \tag{1}$$

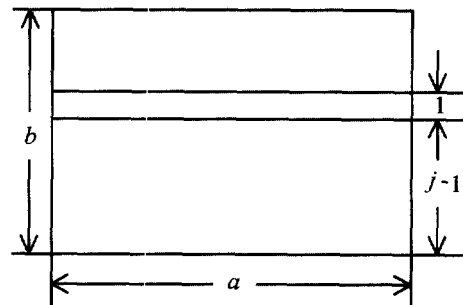


그림 1. 수평띠  $j$ .

다음과 같이 식 (1)의 제약조건을 완화한 수리모형을 고려하자.

$$\text{Maximize } z' = \sum_{j=1}^b \sum_{i \in R_{ab}} \left( \frac{p_i}{w_i} \right) x_{ij}$$

$$\begin{aligned} \text{subject to } \sum_{i \in R_{ab}} l_i x_{ij} &\leq a \text{ for } \forall j \\ &\dots \\ \forall x_{ij} &\in N \end{aligned} \quad (2)$$

식 (1)과 식 (2)의 최적해를 각각  $z^*$ 와  $z'^*$ 라 하면,

$$z^* \leq z'^* \quad (3)$$

이다.

임의의 수평띠  $j$ 에 대하여 다음의 수리모형을 정의하자.

$$\begin{aligned} \text{Maximize } z_j &= \sum_{i \in R_{ab}} \left( \frac{p_i}{w_i} \right) x_{ij} \\ \text{subject to } \sum_{i \in R_{ab}} l_i x_{ij} &\leq a \\ \forall x_{ij} &\in N \end{aligned} \quad (4)$$

식 (4)는 집합  $R_{ab}$ 가 포함하는 제품  $i$ 에 대하여,  $p_i/w_i$ 를 제품의 가치,  $l_i$ 를 제품의 무게로 하고  $a$ 를 배낭의 용량으로 하는 배낭문제이다. 즉, 식 (4)의 최적해  $z_j^*$ 는 임의의 수평띠에 포함될 수 있는 제품의 최대가치이므로 모든 수평띠  $j$ 에 대해 동일한 값을 가진다. 따라서,

$$z' = \sum_{j=1}^b z_j = z_1 + z_2 + \dots + z_b \leq bz_j^* \quad (5)$$

이다. 식 (3)과 식 (5)에 의해  $z^* \leq bz_j^*$ 이므로, 다음의  $u_k(a, b)$ 는 식 (1)의 상한이다.

$$u_k(a, b) = b \max \left\{ \sum_{i \in R_{ab}} \left( \frac{p_i}{w_i} \right) x_i; \sum_{i \in R_{ab}} l_i x_i \leq a, \forall x_i \in N \right\}$$

수직띠에 대해서도 이상과 유사한 방식으로 다음의 상한  $u_v(a, b)$ 를 정의할 수 있다.

$$u_v(a, b) = a \max \left\{ \sum_{i \in R_{ab}} \left( \frac{p_i}{l_i} \right) x_i; \sum_{i \in R_{ab}} w_i x_i \leq b, \forall x_i \in N \right\}$$

따라서,  $u(a, b) = \min \{ \lfloor u_k(a, b) \rfloor, \lfloor u_v(a, b) \rfloor \}$ 는 임의의 자재조각  $(a, b)$ 의 상한이다.

새로운 상한은 Hifi의 상한과 같이 완벽하게 제품 면적의 합을 고려하지는 못한다. 그러나 가로 또는 세로 방향으로 제품이 반복될 수 있는 횟수의 제한이 고려되기 때문에, 결국 상한에 이용된 제품 면적의 총 합이 자재조각의 면적을 넘을 수 없다. 따라서, 제품 면적의 합뿐만 아니라 가로 방향과 세로 방향의 제품의 기하학적 배치까지 고려하기 때문에 대부분의 자재 크기에 대해서 기존의 상한보다 낮은 값을 갖는다. 그러나 새로운 상한이 항상 기존의 상한보다 낮은 값을 갖는 것은 아니다.  $u_h(a, b)$ 를 예로 들면, 수평띠  $(a, 1)$ 의 최적패턴에 포함된 제품  $i$ 의 개수가  $\lfloor a/l_i \rfloor$  이고  $b/w_i$ 가  $\lfloor b/w_i \rfloor$  보다 크면,  $u_h(a, b)$ 에 포함된 제품  $i$ 의 총 개수는  $\lfloor a/l_i \rfloor \times \lfloor b/w_i \rfloor$ 를 넘는다. 따라서,  $u_h(a, b)$ 는  $U(a, b)$ 보다 크다.

임의의 자재조각  $(a, b)$ 에 대하여 제안한 상한을 구하기 위해서는 집합  $R_{ab}$ 를 구하고, 수평띠  $(a, 1)$ 과 수직띠  $(1, b)$ 에 대해 각각 한번씩 배낭문제를 풀어야 한다. 상한은 해법에서 다양한 크기의 자재조각에 대해 빈번히 구해야 하기 때문에, 그때마다 매번 배낭문제를 푼다는 것은 매우 큰 부담이 아닐 수 없다. 그러나 동적계획법을 이용하여 수평띠  $(L, 1)$ 에 대해 배낭문제를 한번만 풀면  $a \leq L$ 인 모든 수평띠  $(a, 1)$ 에 대한 배낭문제의 해가 구해진다(Dreyfus, 1977; Martello and Toth, 1990). 즉, 제품의 세로길이들 중 서로 다른 길이를 갖는 개수를  $r$ 이라 하자. 그리고 이 길이들을  $w_1 \leq w_2 \leq \dots \leq w_r$ 를 만족하도록 정렬했다고 가정하자.  $S_k = \{i: w_i \leq w_k \text{ for } i=1, 2, \dots, n\}$ 라 하면, 집합  $S_k$ 의 제품들로만 구성된 수평띠  $(a, 1)$ 의 최대가치  $x_k(a)$ 는 다음과 같다.

$$x_k(a) = \max \left\{ \sum_{i \in S_k} \left( \frac{p_i}{w_i} \right) x_i; \sum_{i \in S_k} l_i x_i \leq a, \forall x_i \in N \right\}$$

따라서, 임의의 자재조각  $(a, b)$ 에 대한 수평띠의 최대가치는  $w_k \leq b$ 를 만족하는 최대값  $k^*$ 에 대한  $x_{k^*}(a)$ 로 주어진다.  $x_k(a)$ 는 동적계획법을 적용하여  $x_{k-1}(a)$ 의 결과에 차집합  $S_k - S_{k-1}$ 에 포함된 제품들만을 추가로 고려하면 구할 수 있다. 바꾸어 말하면,  $x_r(a)$ 를 동적계획법을 이용하여 한번만 풀면,  $k < r$ 인 모든  $k$ 에 대해  $x_k(a)$ 의 결과를 구할 수 있다는 것이다. 그리고 앞에서 언급한 바와 같이  $x_k(l_0)$ 를 풀면  $a < L$ 인  $x_k(a)$  역시 구할 수 있다. 결국, 해법의 초기 단계에서  $x_r(l_0)$ 를 동적계획법을 사용하여 한번만 구하면, 어떠한 크기의 자재조각  $(a, b)$ 에 대해서도 수평띠의 최대가치를 쉽게 구할 수 있다. 이와 마찬가지로 수직띠  $(1, W)$ 에 대해서 한번만 배낭문제를 풀면  $b \leq W$ 인 모든 수직띠  $(1, b)$ 에 대한 해가 구해진다. 그러므로 해법의 초기 단계에 수평띠  $(L, 1)$ 과 수직띠  $(1, W)$ 에 대해 동적계획법을 사용하여 한번씩만 배낭문제를 풀면, 이후의 과정에서 구해야 하는 임의의 자재조각  $(a, b)$ 에 대한 상한을 쉽게 계산할 수 있다. 다음은 해법의 초기 단계에서  $x_r(l_0)$ 를 구하는 절차이다.

Step 1. [initialize]

Set  $f(x) = 0$  for  $0 \leq x \leq l_0$ .

Set  $S_0 = \phi$ .

Step 2. [calculate  $x_r(l_0)$ ]

For  $k=1, 2, \dots, r$

For  $x=1, 2, \dots, l_0$

For  $i \in S_k - S_{k-1}$

If  $x - l_i \geq 0$  then

If  $\frac{p_i}{w_i} + f(x - l_i) > f(x)$  then

$f(x) \leftarrow \frac{p_i}{w_i} + f(x - l_i)$

End if

End if

End for  
 End for  
 Set  $x_k(x) = f(x)$  for  $0 \leq x \leq L$ .  
 End for  
 Step 3. [finish]  
 Exit.

4. 탐색 전략

2절에서 설명한 바와 같이, Hifi와 Zissimopoulos는 탐색 과정의 중복을 피하기 위해 절단 패턴들이 정규화되어야 한다는 조건과 자재조각  $(a, b)$ 의 수직, 수평 방향의 절단 위치를 항상  $\lfloor \frac{a}{2} \rfloor, \lfloor \frac{b}{2} \rfloor$ 보다 작거나 같게 유지해야 한다는 조건을 만족하는 패턴만을 탐색한다. <그림 2>의 (가), (나), (다)는 모두 동일한 결과를 갖는 중복되는 패턴인데, 이렇게 함으로써 (나)와 (다)같이 중복되는 패턴을 피하고 (가)만을 탐색한다. (나)는 패턴이 정규화 되지 않았고, (다)는  $x_2$ 가  $\lfloor \frac{a}{2} \rfloor$ 를 넘기 때문이다.

<그림 3>의 (가), (나), (다) 역시 모두 동일한 결과를 갖는 중복 패턴이다. 그러나 Hifi와 Zissimopoulos의 탐색 전략으로는 세 경우 모두를 탐색하는 결과를 초래한다. (가)의 경우는  $x_1, x_2$ 의 순서로 절단하고, (나)의 경우는  $x_3, x_1$ , (다)의 경우는  $x_2, x_1$ 의 순서로 절단함으로써 앞의 두 조건이 모두 만족되기 때문이다. Christofides and Witlock(1977)은 대칭(symmetry)과 절단 정렬(cut ordering)의 개념을 사용하여 <그림 3>의 절단 패턴들 중 절단 길이  $x_1, x_2, x_3$ 가 오름차순으로 정렬된 (가)만을 탐색한다. 본 논문에서는 대칭과 절단정렬의 개념을 Hifi와 Zissimopoulos의 재귀 함수에 적용하기 위해  $F(a, b)$ 를 다음과 같이 수정한다.

$$F(a, b, x_0, y_0) = \max \{ F(x, b, x, 1) + F(a-x, b, x, 1) \\ \text{for all } x \geq x_0 \text{ in } P_{\lfloor \frac{a}{2} \rfloor, b}; \\ F(a, y, 1, y) + F(a, b-y, 1, y) \\ \text{for all } y \geq y_0 \text{ in } Q_{a, \lfloor \frac{b}{2} \rfloor} \}$$

이렇게 수정된 재귀 함수가 <그림 3>의 (가)와 같이 절단 길이가 정렬된 절단 패턴만을 탐색하는 이유는 세로(또는 가로) 방향의 절단을 살펴보면,  $x_0$ 는 자재조각이 세로 방향으로 절단될 때 첫 번째 자재조각의 가로길이에 해당한다. 따라서, 첫 번째 자재조각에 대해서는  $x_0$ 보다 크거나 같고  $\lfloor \frac{a}{2} \rfloor$ 보다 작거나 같은 절단 위치를 더 이상 찾을 수 없기 때문에 더 이상 세로 방향의 절단을 고려할 수 없다. 그리고 두 번째 자재조각은 반드시 첫 번째 자재조각의 가로길이보다 크거나 같은 위치부터 절단되어야 하므로, 이로부터 절단되는 두 개의 자재조각의 가로길이는 첫 번째 자재조각의 가로길이보다 항상 크거나 같다. 그러므로 수정된 재귀 함수에 의해 탐색되는 절단 패턴들은 항상 크기가 작은 자재조각부터 큰 순서로 배열된 형태를 갖는다. 따라서, <그림 3>의 (나)와 (다)에 해당하는 절단 패턴들은 탐색에서 제외되는 것이다. Hifi의 재귀 함수 대신 수정된 재귀 함수를 적용하면 2절의 해법의 (i)~(v)는 다음과 같이 수정된다.

- (i) Step 2. [ $F(a, b, x_0, y_0, v_0)$ ]
- (ii) For all  $x \geq x_0$  in  $P_{\lfloor \frac{a}{2} \rfloor, b}$
- (iii)  $Opt(x, b) \leftarrow F(x, b, x, 1, \max(V, v_0) - U(a_0 - x, b))$
- (iv)  $Opt(a_0 - x, b) \leftarrow F(a_0 - x, b, x, 1, \max(V, v_0) - Opt(x, b))$
- (v) Repeat for all  $y \geq y_0$  in  $Q_{a, \lfloor \frac{b}{2} \rfloor}$

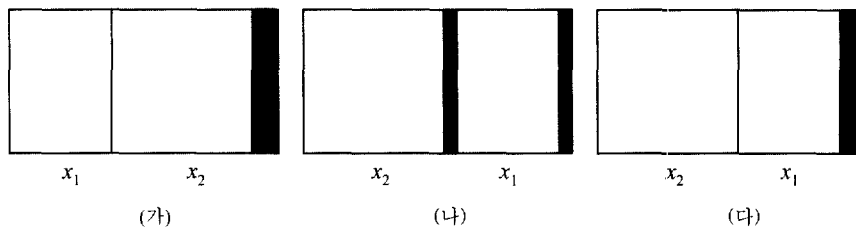


그림 2. Hifi와 Zissimopoulos의 해법이 고려하는 중복 패턴.

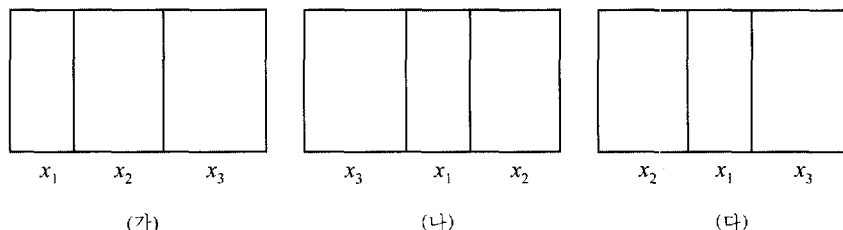


그림 3. 제안한 해법이 고려하는 중복 패턴.

5. 실험 결과

본 절에서는 제안한 해법의 효율성을 보이기 위해 Hifi와 Zissimopoulos가 다룬 동일한 문제에 대해 실험하여 비교한다. <표 1>은 Hifi와 Zissimopoulos의 논문에서 사용한 11개의 예제들에 대해 실험한 결과이다. 펜티엄 II를 장착한 개인용 컴퓨터를 사용하여 실험했음에도 불구하고, 모든 예제들에 대해 평균 0.03초 이내에 최적해를 구할 수 있었다. 그러나 <표 1>만을 보아서는 ±0.01초의 측정오차를 고려할 때, 새로 제안한 해법이 기존의 해법보다 더 향상된 결과를 보인다고 단정하기 어렵다. 새로 제안한 해법은 Hifi와 Zissimopoulos의 해법을 개선한 것이고, 이 해법은 재귀함수  $F(a, b, x_0, y_0)$ 를 반복 호출함으로써 해를 탐색한다. 따라서, 재귀함수  $F(a, b, x_0, y_0)$ 의 호출 횟수는 두 해법의 효율성을 비교하는 데 좋은 척도가 될 수 있다. <표 1>은 위에서 언급한 11개의 예제들에 대해 기존 해법과 제안한 해법을 적용하여 얻어진 재귀함수의 호출 횟수와 감소율을 보인 것이다. 제안한 해법이 기존의 해법에 비해 평균적으로 약 35%의 호출 횟수의 감소 효과가 있음을 관찰할 수 있다.

<표 2>는 Cung 등(2000)의 논문에서 언급된 인터넷 사이트 (<ftp://panoramix.paris1.fr/pub/CERMSEM/hifi/2Dcutting>)로부터 얻어진 11개의 예제(UW1~UW11)에 대해 실험한 결과이다. 이 예제들은 자재의 크기가 (500, 500)에서 (3500, 3650)까지로 <표 1>의 예제들에 비해 큰 값을 가지고 있다. 실험 결과를 살펴보면 기존의 해법에 비해 평균적으로 약 71%의 호출 횟수 감소 효과와 62%의 계산 시간 감소 효과를 관찰할 수 있다. 특히, 자재 (2645, 2763)의 예제에 대해서는 각각 97.2%, 90.9%의 감소율을 보인다. 이것은 제안한 해법이 문제의 크기가 클수록 더욱 우수한 성능을 나타냄을 의미한다.

<표 3>은 Cung 등(2000)의 논문에서 얻은 제품의 가치와 면적이 동일한 (unweighted) 11개의 다른 예제(UU1~UU11)에 대해 실험한 결과이다. 마지막 예제는 자재 크기 (3500, 3765)에 비해 너무 작은 (37, 710)의 크기를 갖는 제품으로 인해 최적해

표 1. Hifi와 Zissimopoulos(1996)의 11개 예제에 대한 실험 결과

(L, W)	최적값	제안한 해법		
		기존 해법의 호출 횟수	호출 횟수	감소율(%)
(127,98)	12348	239	194	18.8
(15,10)	249	72	36	50.0
(40,70)	3076	102	50	51.0
(40,70)	2240	204	69	66.2
(70,40)	2758	258	176	31.8
(70,40)	2776	81	69	14.8
(100,156)	15024	250	175	30.0
(253,294)	73176	487	302	38.0
(318,473)	142817	218	156	28.4
(501,556)	265768	394	286	27.4
(750,806)	577882	348	268	23.0

를 얻을 수 없었다. 이 결과에서도 제품의 가치가 따로 주어지지 않은 (weighted) 문제만큼은 아니지만, Hifi의 해법보다 약 20~45%의 호출 횟수와 약 13~30%의 계산 시간을 감소시킬 수 있었다.

<표 4>는 제안한 상한의 우수함을 보이기 위해, <표 2>와 <표 3>에서 실험한 각 예제들에 대해 무작위로 생성한 1000개의 자재 조각에 대한 기존의 상한과 제안한 상한을 구하여 비교한 것이다. 약 88~99%의 자재 조각 크기에 대해 제안한 상한이 기존의 상한보다 우수하거나 같은 값을 구해줌을 알 수 있다.

6. 결론

본 논문에서는 제품의 개수에 제한이 없는 2차원 절단문제에 대한 효율적인 해법을 제안하였다. 개수 제한이 없는 2차원 절단문제는 2차원 자재절단문제를 선형계획적 접근법으로 풀 때, 해법의 매 반복주기마다 풀어야 한다. 선형계획법으로 구한 해는 매우 우수한 값을 갖지만 대단히 많은 반복주기를 거쳐야 하기 때문에 2차원 자재절단문제에 적용하기에는 어려

표 2. Cung 등(2000)의 11개 예제에 대한 실험 결과

(L, W)	최적값	기존 해법		제안한 해법			
		호출 횟수	시간(초)	호출 횟수	감소율(%)	시간(초)	감소율(%)
(500,500)	6036	1426	0.3	686	51.9	0.2	40.7
(555,632)	15747	17862	3.9	7287	59.2	1.9	52.2
(560,750)	8468	6682	1.4	3463	48.2	0.9	36.9
(700,650)	5964	3451	0.8	1420	58.9	0.3	55.9
(1245,1015)	8326	2790	0.9	173	93.8	0.2	79.8
(1100,1450)	7780	1779	0.5	349	80.4	0.2	57.2
(1750,1542)	6615	6886	2.2	1448	79.0	0.7	66.4
(2250,1875)	10464	12735	4.6	5534	56.5	2.1	54.0
(2645,2763)	7692	13531	6.5	376	97.2	0.6	90.9
(3000,3250)	7038	3249	2.1	924	71.6	0.9	57.6
(3500,3650)	7507	26326	12.4	3127	88.1	1.9	84.5

표 3. Cung 등(2000)의 제품 가치가 없는 11개 예제에 대한 실험 결과

(L, W)	최적값	기존 해법		제안한 해법			
		호출 횟수	시간(초)	호출 횟수	감소율(%)	시간(초)	감소율(%)
(500,500)	242919	641	0.8	467	27.1	0.6	28.6
(750,800)	595288	614	0.7	404	34.2	0.5	24.2
(1100,1000)	1072764	853	1.0	617	27.7	0.8	21.2
(1000,1200)	1179050	3015	3.8	2392	20.7	3.4	12.8
(1450,1300)	1868999	3897	4.7	2340	40.0	3.4	29.0
(2050,1457)	2950760	586	1.0	464	20.8	0.8	17.2
(1465,2024)	2930654	11524	15.4	8427	26.9	12.6	17.9
(2000,2000)	3959352	3559	4.6	2481	30.3	3.6	21.7
(2500,2460)	6100692	2491	3.2	1366	45.2	2.3	28.7
(3500,3450)	11955852	11231	16.3	7605	32.3	13.1	19.9
(3500,3765)	—	—	—	—	—	—	—

움이 많았다(G, 2000). 선형계획적 접근법으로 2차원 자재절단문제를 현실적인 시간 내에 풀기 위해서는 개수 제한이 없는 2차원 절단문제를 위한 매우 효율적인 해법이 절실히 요구된다.

본 연구에서 제안하는 해법은 기존의 해법 중 가장 우수하다고 알려진 Hifi와 Zissimopoulos의 재귀함수를 이용한 해법을 새로운 상한과 탐색전략의 개선을 통해 더욱 향상시킨 것이다. 기존의 상한이 단지 제품의 면적의 합만을 고려하는 것과는 달리, 새로운 상한은 면적의 합뿐만 아니라 가로 방향과 세로 방향의 제품의 기하학적 배치까지 고려하기 때문에

기존의 상한보다 대부분의 자재 크기에 대해 낮은 값을 갖는다. 그리고 개선된 탐색전략은 항상 크기가 작은 자재 조각부터 오름차순으로 정렬된 절단 패턴만을 탐색하기 때문에 기존의 해법에 비해 작은 해공간을 탐색함으로써 최적해를 구할 수 있다.

실험을 통해 Hifi와 Zissimopoulos의 해법과 비교한 결과, 제안한 해법이 평균적으로 51% 정도의 효율 향상과 최대 약 90%의 계산 시간의 감소를 보였다. 이와 같은 효과는 문제의 크기가 클수록 더욱 두드러짐을 관찰할 수 있는데, 이것은 현장에서 풀어야 하는 자재절단문제의 자재 크기가 약 (1200,

표 4. 기존 상한(Hifi)과 제안한 상한(Proposed)의 비교

(L, W)	Hifi > Proposed	Hifi = Proposed	Hifi < Proposed	Rate(%) of Hifi ≥ Proposed
(500,500)	570	379	51	89.5
(555,632)	572	340	88	87.8
(560,750)	547	348	105	94.8
(700,650)	509	369	122	91.4
(1245,1015)	308	640	52	89.5
(1100,1450)	294	620	86	89.9
(1750,1542)	380	515	105	90.3
(2250,1875)	411	488	101	88.6
(2645,2763)	462	441	97	89.5
(3000,3250)	504	382	114	96.7
(3500,3650)	519	376	105	95.7
(500,500)	558	409	33	97.7
(750,800)	666	291	43	96.7
(1100,1000)	588	389	23	97.4
(1000,1200)	563	404	33	96.9
(1450,1300)	584	390	26	98.8
(2050,1457)	553	416	31	98.7
(1465,2024)	610	378	12	98.1
(2000,2000)	597	390	13	98.0
(2500,2460)	592	389	19	98.5
(3500,3450)	581	399	20	97.7
(3500,3765)	629	356	15	96.9

2400)의 큰 문제임을 고려할 때 매우 바람직한 현상이다(G, 2000). 따라서, 2차원 자재절단문제에 대해 선형계획적 접근법을 현실화하기 위해 본 연구에서 제안한 해법이 성공적으로 적용될 수 있을 것이다.

## 참고문헌

- Beasley, J. E. (1985), Algorithms for unconstrained two-dimensional guillotine cutting, *Journal of the Operational Research Society*, **36**, 297-306.
- Cung, V.-D., Hifi, M. and Le Cun, B. (2000), Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm, *International Transactions in Operational Research*, **7**, 185-210.
- Christofides, N. and Whitlock, C. (1977), An algorithm for two-dimensional cutting problems, *Operations Research*, **25**, 30-44.
- Dreyfus, S. E. and Law, A. M. (1977), *The Art and Theory of Dynamic Programming*, Academic Press, Inc., Orlando.
- Fayard, D. and Zissimopoulos, V. (1995), An approximation algorithm for solving unconstrained two-dimensional knapsack problems, *European Journal of Operational Research*, **84**, 618-632.
- G, Y. G. (2000), *An Efficient Algorithm for the Two-Dimensional Guillotine Cutting Stock Problem with Various Constraints and Integration with Production Information System*, A Doctoral Thesis, Hanyang University, Seoul, Korea.
- Gilmore, P. and Gomory, R. (1961), A linear programming approach to the cutting stock problem, *Operations Research*, **9**, 848-859.
- Gilmore, P. and Gomory, R. (1965), Multistage cutting problems of two and more dimensions, *Operations Research*, **13**, 94-119.
- Gilmore, P. and Gomory, R. (1966), The theory and computation of knapsack functions, *Operations Research*, **14**, 1045-1074.
- Haessler, R. W. and Sweeney, P. E. (1991), Cutting stock problems and solution procedures, *European Journal of Operational Research*, **54**, 141-150.
- Herz, J. C. (1972), A recursive computing procedure for two-dimensional stock cutting, *IBM Journal of Research and Development*, **16**, 462-469.
- Hifi, M. and Zissimopoulos, V. (1996), A recursive exact algorithm for weighted two-dimensional cutting, *European Journal of Operational Research*, **91**, 553-564.
- Martello, S. and Toth, P. (1990), *Knapsack Problem: Algorithms and Computer Implementations*, John Wiley & Sons, Chichester.
- Morabito, R., Arenales, M. and Arcaro, V. (1992), An AND/OR graph approach for two-dimensional cutting problems, *European Journal of Operational Research*, **58**, 263-271.
- Viswanathan, K. V. and Bagchi, A. (1993), Best-first search methods for constrained two-dimensional cutting stock problems, *Operations Research*, **41**, 768-776.