論文
3-6

무제한3차원절단문제를위한새로운분지한계법

ANewbranchandboundalgorithmforunconstrained three-dimensionalcuttingproblems

성영조(Young-Jo Seong)[1] 강맹규(Maing-Kyu Kang)[2]

요약

무제한 3차원 절단문제는 커다란 직육면체 자재에서 작은 직육면체 제품을 잘라낼 때, 잘라낸 제품의 가치가 최대가 되도록 하는 절단패턴을 찾는 문제이다. 이 때, 잘려지는 각각의 제품은 고유의 크기와 가치를 갖는다. 두 개의 절단패턴을 결합해서 새로운 절단패턴을 만들어 내는 상향식 접근법은 무제한 3차원 절단문제를 푸는 해법으로 사용할 수 있다. 본 논문에서는, Wang이 제안한, 2차원 문제에서 사용되는 두 가지 결합 외에 새로운 결합을 정의하고, 지 등이 무제한 2차원 절단 문제를 풀기 위해 제안한 분지전략을 3차원 문제에서 사용할 수 있도록 새롭게 재구성한다.

Abstract

An unconstrained three-dimensional cutting problem describes the process of finding the cutting pattern that yields the maximum total profit-sum for the small parallelepipeds pieces cut from a large parallelepiped box and there is no limit to the number of pieces to be cut. The problem is a classic NP-hard. The bottom-up approach, which generates all of the feasible cutting patterns by combining two other cutting patterns, can be applied to the three-dimensional problem. We introduce a build and new branching strategies for the unconstrained three-dimensional cutting problem. The strategies are all generalized from the branching strategies proposed by G et al. to solve unconstrained two-dimensional cutting problems.

논문접수 : 2004. 3. 10.
심사완료 : 2004. 3. 25.

1) 정회원 : 한양대학교 대학원 산업공학과
2) 정회원 : 한양대학교 대학원 산업공학과

1.Introduction

A three-dimensional cutting (3DC) problem describes the process of finding the cutting pattern that yields the maximum total profit-sum for the small parallelepiped pieces cut from a large parallelepiped box [4]. Each piece has its own size and profit. A constrained cutting problem limits the number of each piece produced if the number of each piece is unlimited, then the problem is termed an unconstrained problem. In this paper, we deal with the unconstrained 3DC (U3DC) problem. Generally, the cutting problem includes two restrictions: i) only guillotine-type cuts are used, *i.e.*, each cut produces two sub-parallelepipeds; and ii) all pieces have a fixed orientation, *i.e.*, each piece cannot be rotated.

A two-dimensional cutting (TDC) problem is a special case of the 3DC problem and has been extensively introduced in many researches. The TDC problems have been solved exactly using dynamic programming [1, 5], tree search procedures [2, 6, 8] and bottom-up approach [7, 10]. The unconstrained TDC (UTDC) problem was successfully solved using the bottom-up approach by G et al. [3]. Despite the abundance of the TDC literature, the 3DC literature is very limited [9].

In the bottom-up approach, a new cutting pattern is obtained from a horizontal or vertical build of two sub-patterns [11]. Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be the cutting patterns of ($x\boldsymbol{A}$, $y\boldsymbol{A}$) and ($x\boldsymbol{B}$,$y\boldsymbol{B}$), respectively, where (x, y) denotes the size of a cutting pattern with length x and width y. Let $g(\boldsymbol{A})$ and $g(\boldsymbol{B})$ be the profits of $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively. A new cutting pattern of ($x\boldsymbol{A}$+$x\boldsymbol{B}$, max{$y\boldsymbol{A}$, $y\boldsymbol{B}$}) with profit $g(\boldsymbol{A})$ + $g(\boldsymbol{B})$ can be obtained from the horizontal build of $\boldsymbol{A}$ and $\boldsymbol{B}$, ($\boldsymbol{A}$, $\boldsymbol{B}$)h. Similarly, the vertical build of $\boldsymbol{A}$ and $\boldsymbol{B}$, ($\boldsymbol{A}$, $\boldsymbol{B}$)v, generates another new cutting pattern of (max{$x\boldsymbol{A}$, $x\boldsymbol{B}$}, $y\boldsymbol{A}$+$y\boldsymbol{B}$) with profit $g(\boldsymbol{A})$ + $g(\boldsymbol{B})$.

In this paper, we introduce a build named 'parallel build' besides the two builds proposed by Wang and redefine the three builds for the 3DC problems. To solve the U3DC problem, we also introduce new branching strategies, which are all generalized from the branching strategies proposed by G et al. [3] to solve UTDC problems.

2.Bottom-up approach for thehree-dimensionalcuttingproblem

In the bottom-up approach for the 3DC problem, a new cutting pattern is obtained from a horizontal or vertical or parallel build of two sub-patterns. Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be the cutting patterns of ($x\boldsymbol{A}$, $y\boldsymbol{A}$ $z\boldsymbol{A}$) and ($x\boldsymbol{B}$,$y\boldsymbol{B}$,$z\boldsymbol{B}$), respectively, where (x, y, z) denotes the size of a cutting pattern with length x, width y, and height z. Let $g(\boldsymbol{A})$ and $g(\boldsymbol{B})$ be the profits of $\boldsymbol{A}$and $\boldsymbol{B}$, respectively.

As illustrated in Figure 1, a cutting pattern of ($x\boldsymbol{A}$ + $x\boldsymbol{B}$, max{$y\boldsymbol{A}$, $y\boldsymbol{B}$}, max{$z\boldsymbol{A}$, $z\boldsymbol{B}$}) with profit $g(\boldsymbol{A})$ + $g(\boldsymbol{B})$ can be obtained fromthe horizontal build of $\boldsymbol{A}$ with $\boldsymbol{B}$, ($\boldsymbol{A}$, $\boldsymbol{B}$)h. Similarly, the vertical build of $\boldsymbol{A}$ with $\boldsymbol{B}$, ($\boldsymbol{A}$, $\boldsymbol{B}$)v, also generates a new cutting pattern of (max{$x\boldsymbol{A}$, $x\boldsymbol{B}$}, $y\boldsymbol{A}$ + $y\boldsymbol{B}$, max{$z\boldsymbol{A}$, $z\boldsymbol{B}$}) with profit $g(\boldsymbol{A})$ + $g(\boldsymbol{B})$. The parallel build of $\boldsymbol{A}$ with $\boldsymbol{B}$, ($\boldsymbol{A}$, $\boldsymbol{B}$) p generates another new cutting pattern of (max{$x\boldsymbol{A}$, $x\boldsymbol{B}$}, max{$y\boldsymbol{A}$, $y\boldsymbol{B}$}, $z\boldsymbol{A}$ + $z\boldsymbol{B}$) with profit $g(\boldsymbol{A})$ + $g(\boldsymbol{B})$. The shaded volumes in the figure represent waste.

We use a best-first search method based upon the bottom-up approachto solve the U3DC problem. Each iteration of the method generally consists of the following procedure:

i) selecting a leaf node (i.e. not yet branched node) with the highest upper bound and moving it from leaf nodes set into branched nodes set, ii) branching out child nodes from the selected node, and iii) adding the child nodes with upper bounds greater than the current optimal value into the leaf nodes set. In the best-first search method, each node represents a cutting pattern, and branching refers to the construction of the horizontal, vertical and parallel builds. Therefore, child nodes (*i.e.*, new cutting patterns) of a selected node are generated by the builds between the selected node and all of the branched nodes (including the selected node).

Consider a U3DC problem instance with a parallelepiped box of (L, W, H) and n types of parallelepiped pieces, such that the size and profit of the ith piece are (li, wi, hi) and pi, respectively. Assume that L, W, H, li, wi, hi and pi for all pieces are positive integers. Let $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$ and $g(\boldsymbol{R})$ be the size and profit of a node $\boldsymbol{R}$, respectively. Then, the upper bound at the node $\boldsymbol{R}$, $U(\boldsymbol{R})$ is represented as

$$U(\boldsymbol{R}) = g(\boldsymbol{R}) + u(\boldsymbol{P}) \quad (1)$$

where $u(\boldsymbol{P})$ is the upper bound for the complementary volume $\boldsymbol{P}$ for $\boldsymbol{R}$ shown in Figure 2.

At each node, $g(\boldsymbol{R})$ in equation (1) can be easily obtained by summing up the profits of the pieces contained in $\boldsymbol{R}$, but $u(\boldsymbol{P})$ must be calculated to obtain $U(\boldsymbol{R})$.

3. Branching strategies

In this section, new branching strategies as well as some useful propositions are proposed for the U3DC problem. These strategies and propositions are generalized from those proposed by G et al. [G03] to solve the UTDC problems. Consider a best-first branch and bound algorithm where the leaf node set is **O** and the branched node set is **C**. Let (a, b, c) (or or =) (d, e, f) denote a (or or =) d, b (or or =) e and c (or or =) f.

Definition 1. Given an instance of an unconstrained three-dimensional cutting problem, a node (cutting pattern) $\boldsymbol{R}$ is a dominated node (dominated cutting pattern) if the optimal solution does not change when $\boldsymbol{R}$ is not considered.

Proposition 1. Given a node $\boldsymbol{R}$, if there exists a node $\boldsymbol{R}$ such that

$$(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R}) \quad (l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R}) \text{ and } g(\boldsymbol{R}) \quad g(\boldsymbol{R}) \quad (2)$$

then $\boldsymbol{R}$ is dominated by $\boldsymbol{R}$.

Proof. For any pattern including $\boldsymbol{R}$, $\boldsymbol{R}$ can replace $\boldsymbol{R}$ without resizing the pattern because $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$ $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$. Since $g(\boldsymbol{R})$ $g(\boldsymbol{R})$, the profit of the replacement is larger than or equal to the original profit.

***Strategy* 1**. When moving a node $\boldsymbol{R}$ from **O** into **C**, if there exists a node $\boldsymbol{R}$ in **C** satisfying (2) then $\boldsymbol{R}$ is pruned.

Strategy 1 not only prunes the node $\boldsymbol{R}$ but also removesa lot of nodes (cutting patterns) that would be branched from other nodes by combining with$\boldsymbol{R}$ in the future.

Proposition2. For any two nodes $\boldsymbol{R}$in **O** and $\boldsymbol{R}$ in **C**, if $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$ $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$ then the inequality $g(\boldsymbol{R})$ $g(\boldsymbol{R})$ holds.

Proof. In the best-first branch and bound algorithm, $U(\boldsymbol{R})$ $U(\boldsymbol{R})$, namely, $g(\boldsymbol{R})+u(\boldsymbol{P})g(\boldsymbol{R})+u(\boldsymbol{P})$ where $\boldsymbol{P}$ is the complementary volume for $\boldsymbol{R}$. From the inequality $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$ $(l\boldsymbol{R}, w\boldsymbol{R}, h\boldsymbol{R})$, $u(\boldsymbol{P})$ $u(\boldsymbol{P})$. Therefore, $g(\boldsymbol{R})$ $g(\boldsymbol{R})$ holds.

Proposition3. For any two nodes $\boldsymbol{R}$and $\boldsymbol{R}$ in **C**, if (lR, wR, hR) (lR, wR, hR) then the inequality $g(\boldsymbol{R})$ $g(\boldsymbol{R})$ holds.

Proof. Proposition 3 is valid by Strategy 1 and Proposition 2.

The second strategy is used when child

nodes are branched on a node ***R*** in **O**. The child nodes of ***R*** are generated by horizontal, vertical and parallel builds between the node ***R*** and all of the elements in **C**(including ***R***). The following Strategy 2 reduces the number of these builds.

Strategy **2**. In the horizontal or vertical or parallel build procedure, all dominated patterns identified by the following Propositions 4 and 5, are removed.

Proposition 4. Given a node ***R*** in **O**, if there exists any two nodes ***R*** and ***R*** in **C** such that *wR* *wR* *wR* and *hR* = *hR* = *h* and *lR* = *lR* = *l*
then ***H*** = (***R***, ***R***)*h* is dominated by ***H*** = (***R***, ***R***)*h*.

Proof. ***R***, ***R*** **C** and (*lR*, *wR*, *hR*) (*lR*, *wR*, *hR*), thus *g*(***R***) *g*(***R***) by Proposition 3.
(*lH*, *wH*, *hH*) = (*lR* + *lR*, max{*wR*, *wR*}, max{*hR*, *hR*}) = (*lR* + *l*, *wR*, max{*hR*, *h*}),
(*lH*, *wH*, *hH*) = (*lR* + *lR*, max{*wR*, *wR*}, max{*hR*, *hR*}) = (*lR* + *l*, *wR*, max{*hR*, *h*})
and
g(*H*) = *g*(***R***)+ *g*(***R***),
g(*H*) = *g*(***R***)+ *g*(***R***) *g*(***R***)+ *g*(***R***) = *g*(*H*).
Thus,
(*lH*, *wH*, *hH*) = (*lH*, *wH*, *hH*) and *g*(*H*) *g*(*H*).
Therefore, ***H*** dominates ***H*** by Proposition 1.
Similarly, the vertical case can be applied to the nodes ***R*** in **O** and ***R***, ***R*** in **C** such that
hR *hR* *hR* and *lR* = *lR* = *l* and *wR* = *wR* = *w*.
The parallel case can also be applied to the nodes ***R*** in **O** and ***R***, ***R*** in **C** such that
lR *lR* *lR* and *wR* = *wR* = *w* and *hR* = *hR* = *h*.

Proposition 5. Given a node ***R*** in **O**, if there exists a node $\bar{R}$ such that

$w_{\bar{R}}$ = min {*wR*| *wR* *wR*, *hR* = *hR* = *h*, *lR* = *lR* = *l*, ***R*** **C**}

then (***R***, ***R***)*h* is dominated by ($\bar{R}$, ***R***)*h*, where ***R*** is an arbitrary node such that *wR* $w_{\bar{R}}$, ***R*** **C**.

Proof. $R \in \mathbf{O}, \bar{R} \in \mathbf{C}$ and $(l_R, w_R, h_R) \le (l_{\bar{R}}, w_{\bar{R}}, h_{\bar{R}})$ thus the inequality $g(R) \le g(\bar{R})$ holds by Proposition 2. The two horizontal builds, (***R***, ***R***)*h* and $(\bar{R}, R')^h$ have the same dimension of (*l* + *lR*, *wR*, max{*h*, *hR*}), and the inequality $g(R)+g(R')$ $g(\bar{R})+g(R')$ holds between their profits. Therefore, (***R***, ***R***)*h* is dominated by $(\bar{R}, R')^h$ by Proposition 1 and can be eliminated since $(\bar{R}, R')^h$ already exists in **O**.

Similarly, this proposition can be applied to the vertical and parallel cases. Therefore, when generating the child nodes of ***R*** using horizontal or vertical or parallel build, a number of dominated builds can be eliminated by Propositions 4 and 5.

The following two strategies are used when a horizontal or vertical or parallel build, ***q*** (a newly generated child node), is moved into the leaf node set **O**.

Strategy **3**. When moving the child node ***q*** into **O**, if there exists a node ***R*** in **O** such that

(*lq*, *wq*, *hq*) = (*lR*, *wR*, *hR*) and *g*(***q***) *g*(***R***) (3)

then ***q*** is removed.

If there exists ***R*** in **O** satisfying (3), ***q*** is dominated by ***R*** by Proposition 1.

Since the proposed algorithm terminates when there remains no element in **O** and Strategy 3 reduces the number of the element in **O**, the strategy can improve the algorithm performance.

The proposed branching strategies require only a· maximum of 2*LWH* of memoryspace.

The space required for the branched node set **C** is less than or equal to *LWH* by Strategy 1 and the space for the leaf node set **O** is also less than or equal to *LWH* by Strategy 3. Therefore, the space complexity of the proposed algorithm is O(*LWH*).

4. Conclusion

We introduced a best-first branch and bound algorithm to solve U3DC problems. While previous exact algorithms for the unconstrained cutting problem generally used top-down approach, the branch and bound algorithm was based upon a bottom-up approach using horizontal, vertical, and parallel builds.The proposed branching strategies were generalized from the strategies proposed by G et al. to solve UTDC problems. Therefore, the bottom-up approach can be used for all dimensional problems. To solve the three-dimensional problem, an upper bound for the three-dimensional volume ***P*** shown in Figure 2 is necessary. The upper bound is left for a future study.

References

[1] J.E. Beasley, Algorithms for u n c o n s t r a i n e d two-dimensionalguillotine cutting, Journal of the Operational Research Society 36 (1985) 297-306.

[2] Y.-G. G and M.-K. Kang, A new upper bound for unconstrained two-dimensional cutting and packing, Journal of the Operational Research Society 53 (2002) 587-591.

[3] Y.-G. G, Y.-J. Seong, and M.-K. Kang, A best-first branch and bound algorithm for unconstrained two-dimensional cutting problems, Operations Research Letters 31 (2003) 301-307.

[4] P.C. Gilmore and R.E. Gomory, Multistage cutting stock problems of two and more dimensions,Operations Research 13 (1965) 94-119.

[5] P.C. Gilmore and R.E. Gomory, The theory and computation of knapsack functions, Operations Research 14 (1966) 1045-1074.

[6] J.C. Herz, A recursive computing procedure for two-dimensional stock cutting, IBM Journal of Research and Development 16 (1972) 462-469.

[7] M. Hifi, An improvement of Viswanathan and Bagchi's exact algorithm for constrained two-dimensional cutting stock, Computers & Operations Research 24 (1997) 727-736.

[8] M. Hifi and V. Zissimopoulos, A recursive exact algorithm for weighted two-dimensional cutting, European Journal of Operational Research 91 (1996) 553-564.

[9] M. Hifi, Exact algorithms for unconstrained three-dimensional cutting problems: a comparative study, Computers & Operations Research 31 (2004) 657-674.

[10] K.V. Viswanathan and A. Bagchi, Best-first search methods for constrained two-dimensional cutting stock problems, Operations Research 41 (1993) 768-776.

[11] P.Y. Wang, Two algorithms for constrained two-dimensional cutting stock problems, Operations Research 31 (1983) 573-586.

성영조

1994년 한양대학교 졸업
1996년 한양대학교대학원 석사 졸업
한양대학교 대학원 박사 수료
1996년 3월 ~ 199년 10월 한양대학교 공학기술연구소 연구원
1999년 11월 ~ 2004년 이알하우스(주) 이알정보기술연구소