Chapter 6
Cutting and Packing Problems with Placement Constraints

Andreas Fischer and Guntram Scheithauer

Abstract In real-life problems of cutting and packing very often placement constraints are present. For instance, defective regions of the raw material (wooden boards, steel plates, etc.) shall not become part of the desired products. More generally, due to different quality demands, some products may contain parts of lower quality which are not allowed for other goods. Within this work we consider one- and two-dimensional rectangular cutting and packing problems where items of given types have to be cut from (or packed on) raw material such that an objective function attains its maximum. In the one-dimensional (1D) case, we assume for each item type that allocation intervals (regions of the raw material) are given so that any item of the same type must be completely contained in one of the corresponding allocation intervals. In addition, we deal with problems where the lengths of the 1D items of a given type may vary within known tolerances. In the two-dimensional (2D) case, where rectangular items of different types have to be cut from a large rectangle, we investigate guillotine cutting under the condition that defective rectangular regions are not allowed to be part of the manufactured products (even not partially). For these scenarios we present solution strategies which rely on the branch and bound principle or on dynamic programming. Based on properties of the corresponding objective functions we discuss possibilities to reduce the computational complexity. This includes the definition of appropriate sets of potential allocation (cut) points which have to be inspected to obtain an optimal solution. By dominance considerations the set of allocation points is kept small. In particular, the computational complexity becomes independent of the unit of measure of the input data. Possible generalizations will be discussed as well.

Keywords Cutting and packing • Placement constraints • Quality demands • Defective regions • Set of allocation points

A. Fischer (✉) • G. Scheithauer
Institute of Numerical Mathematics, Technische Universität Dresden, 01062 Dresden, Germany
e-mail: Andreas.Fischer@tu-dresden.de; Guntram.Scheithauer@tu-dresden.de

G. Fasano, J.D. Pintér (eds.), *Optimized Packings with Applications*, Springer
Optimization and Its Applications 105, DOI 10.1007/978-3-319-18899-7_6

6.1 Introduction

In real-life problems of cutting and packing placement constraints are present very often. For instance, defective regions of the raw material (wooden boards, steel plates, etc.) shall not become part of desired products. More generally, due to different quality demands, some products may contain parts of lower quality which are not allowed for other goods. For packing problems forbidden regions may exist where no objects must be placed. Due to the strong relationship between cutting and packing problems this paper mostly concentrates on cutting problems.

6.1.1 Aims and Scope

We consider one- and two-dimensional rectangular cutting problems where items of given types have to be cut from raw material such that an objective function attains its maximum. Such problems are also called 1D or 2D rectangular knapsack problems. Two scenarios will be discussed in detail. In the first, some rectangular parts of the raw material are not allowed to be used at all. In the second more general scenario, different quality demands are considered.

In the 1D case, we assume for each item type that allocation intervals (regions of the raw material) are given so that any item of the same type must be completely contained in one of the corresponding allocation intervals. In the 2D case, where rectangular items of different types have to be cut from a large rectangle, we investigate guillotine cutting under the condition that defective rectangular regions are not allowed to be part of the manufactured products (even not partially). Different qualities of the raw material are described by allocation areas.

Furthermore, we also deal with problems where the lengths (or width) of the (one- or two-dimensional) items of a given type may vary within known tolerances.

For these scenarios we present solution strategies which rely on the branch and bound (B&B) principle or on dynamic programming (DP). Based on properties of the corresponding objective functions we discuss possibilities to reduce the computational complexity. This includes the definition of appropriate sets of potential allocation (cut) points which have to be inspected to obtain an optimal solution. By dominance considerations the set of allocation points is kept small. In particular, the computational complexity becomes independent of the unit of measure of the input data.

Some generalizations will be discussed as well. Finally, we hope that techniques from this area can be used and extended to new fields, for example for the placement of chips and other electronic parts on boards.

6.1.2 Related Work

Cutting and packing (C&P) problems with defective or forbidden regions were studied in the past. In the earlier survey paper by Sweeney and Paternoster [15] some work related to this topic is referenced, whereas in the recent typology of C&P [17] the topic is only briefly addressed. Therefore, we give a short overview on articles that are relevant for our work.

Hahn [6] presented a recursive DP-based procedure to solve a 2D cutting problem with defects. She suggested a three-stage guillotine cutting scenario with vertical cuts in the first stage.

Herz [7] presented a recursive B&B-based procedure for the 2D rectangular knapsack problem (without defects) to obtain *canonical* patterns by introducing *discretization points*, see the definition of allocation points in Sect. 6.2. Dowsland [4] used certain discretization points to analyze the structure of optimal (and nearly optimal) solutions and the objective function for the manufacturer's pallet loading problem, a special 2D knapsack problem where only one type of pieces (rotatable by 90°) has to be packed.

Beasley [2] presented a 0/1 model and a tree-search procedure for 2D non-guillotine rectangle packing including the occurrence of forbidden regions. Upper bounds are computed from a Lagrangian relaxation problem which are improved by the help of a subgradient ascent method. We will not use the approach in [2] since it requires a very large number of 0/1-variables.

In Terno et al. [16] a principle used by Nicholson [8] was applied to 2D rectangle cutting and packing problems leading to the concept of reduced sets of allocation (or cut) points (cf. Sect. 6.2). The book (Scheithauer [12]) presents a renewed description of this concept.

For the three-stage guillotine cutting of defective boards a recursive procedure was developed by Scheithauer and Terno [13]. In particular, appropriately reduced sets of allocation points were applied.

Algorithmic approaches for 1D cutting problems with different quality demands were addressed by Sweeney and Haessler [14]. Such problems which are modeled by allocation intervals and pieces of variable length were also investigated in Scheithauer [11]. The latter paper generalizes real-world problems in hardwood cutting. Similar problems are considered in Rönnqvist [9] and Rönnqvist and Åstrand [10], where a discretization of the board is used.

6.1.3 General Notation and Assumptions

We assume throughout the paper that all input data are positive integers. The set of positive integers is denoted by $\mathbb{Z}_>$. In the 1D case, the length of the raw material is given by L. The pieces $i \in I := \{1, \ldots, m\}$ which shall be cut have the lengths ℓ_i.

Moreover, profit coefficients γ_i for $i \in I$ are known. Additionally, in the 2D case, the raw material has width W, whereas the rectangular pieces are of widths w_i. It is always assumed that

$$\max\{\ell_i \mid i \in I\} \leq L \quad \text{and} \quad \max\{w_i \mid i \in I\} \leq W$$

holds. For later use we define

$$\ell_{min} := \min\{\ell_i \mid i \in I\}, \quad w_{min} := \min\{w_i \mid i \in I\}.$$

and

$$\ell := (\ell_1, \ldots, \ell_m)^\top, \quad w := (w_1, \ldots, w_m)^\top, \quad \gamma := (\gamma_1, \ldots, \gamma_m)^\top.$$

Moreover, in case that a maximum of indexed numbers is taken over an empty index set the maximum is set to 0. In order to describe (parts of) objects we use

$$[a, b] \times [c, d] := \{(x, y) \in \mathbb{R}^2 \mid a \leq x \leq b,\ c \leq y \leq d\}$$

and, for short, $b \times d := [0, b] \times [0, d]$, where $a, b, c, d \in \mathbb{Z}_+$ with $a \leq b$ and $c \leq d$. By $\mathbb{Z}_+$ the set of all non-negative integers is denoted. If not stated otherwise, we allow that several copies of a piece can be obtained from the raw material. For the sake of simplicity, we do not consider a (positive) kerf nor least distances between two allocation points within a pattern. Furthermore, in the 2D case we do not allow rotation of pieces for the same reason.

6.2 Reduced Set of Potential Allocation Points

The (standard) *Knapsack Problem* (KP) is a basic problem also within the field of cutting and packing. This problem consists of finding a vector $x^* \in \mathbb{Z}_+^m$ such that x^* satisfies the capacity constraint $a^\top x \leq b$ and the objective $c^\top x$ attains its maximum for $x = x^*$. For short, we write

$$\text{KP}(c, a, b): \quad \sum_{i \in I} c_i x_i \to \max \quad \text{subject to} \quad \sum_{i \in I} a_i x_i \leq b,\ x_i \in \mathbb{Z}_+ \text{ for } i \in I, \tag{6.1}$$

where $a = (a_1, \ldots, a_m)^\top \in \mathbb{Z}_>^m$, $c = (c_1, \ldots, c_m)^\top \in \mathbb{Z}_>^m$, and $b \in \mathbb{Z}_>$ are given.

If we consider $a^\top x \leq y$ with a parameter $y \in \mathbb{R}$, then we obtain the following optimal value function $f : \mathbb{R} \to \mathbb{Z}_+ \cup \{-\infty\}$ related to KP(c, a, b):

$$f(y) := \max\left\{\sum_{i \in I} c_i x_i \mid \sum_{i \in I} a_i x_i \leq y,\ x_i \in \mathbb{Z}_+ \text{ for } i \in I\right\} \quad \text{for all } y \in \mathbb{R}, \tag{6.2}$$

where $f(y) := -\infty$ for any $y < 0$. It is well known that f is piecewise constant and non-decreasing. Its jump discontinuities are non-negative integers. The set of all these jump discontinuities of f depends on c and a and is a subset of

$$S(a) := \left\{ r = \sum_{i \in I} a_i x_i \mid x_i \in \mathbb{Z}_+,\ i \in I \right\}. \tag{6.3}$$

The fact that the optimal value function of KP(c, a, b) changes (increases) only at discrete points can be used in B&B or DP approaches to reduce the computational complexity of solving the knapsack problem.

If the knapsack problem (6.1) is used to model a cutting problem we call $S(a)$ *set of potential allocation points*. Replacing c by γ, a by ℓ, and b by L we see that KP(γ, ℓ, L) models a 1D cutting problem. In this case, x_i denotes the number how often piece i is cut. The set $S(\ell)$ contains infinitely many elements whereas the points (coordinates) for cutting the raw material are bounded by L. Therefore, we introduce the finite set

$$S(\ell, L) := \{ r \in S(\ell) \mid r \le L \}, \tag{6.4}$$

Of course, $S(\ell, L)$ contains all those jump discontinuities of the optimal value function arising from KP(γ, ℓ, L) which are not larger than L. Depending on γ, additional points may belong to $S(\ell, L)$ as well. Thus, the question arises whether one can describe the set of jump discontinuities exactly. This is possible in the important case when $\gamma = \ell$. Then, the knapsack problem KP(ℓ, ℓ, L) has the optimal value function f given by

$$f(y) = \max \left\{ \sum_{i \in I} \ell_i x_i \mid \sum_{i \in I} \ell_i x_i \le y,\ x_i \in \mathbb{Z}_+,\ i \in I \right\} \quad \text{for all } y \in \mathbb{R}$$

and $S(\ell, L)$ is exactly the set of those jump discontinuities of f which are not larger than L.

Let x^* denote a solution of KP(γ, ℓ, L) with $\ell^\top x^* < L$, i.e., there is some waste of raw material. Then, to cut the items according to x^*, infinitely many possibilities exist to choose a pattern, i.e. the coordinates of the items of the solution. If the items are placed as left as possible on the raw material, the number of such patterns is finite, all the waste lies right of the items, and all coordinates of the cut positions belong to $S(\ell)$. Such patterns are often called *normalized* or *left-justified*. Herz [7] used the terms *discretization point* for $r \in S(\ell)$ and *canonical* for left-justified patterns.

For a set $T \subset \mathbb{Z}_+$ and $y \in \mathbb{R}_+$ let $p_T(y)$ and $s_T(y)$ denote the *predecessor of y with respect to T* and the *successor of y with respect to T*, respectively, i.e.,

$$p_T(y) := \max\{r \in T \mid r \le y\} \quad \text{and} \quad s_T(y) := \min\{r \in T \mid r \ge y\}$$

for all $y \in [\min\{r \in T\}, \max\{r \in T\}]$. In terms of the 1D cutting problem, $p_{S(\ell)}(y)$ denotes the largest allocation point less than or equal to y, and $s_{S(\ell)}(y)$ denotes the least length of raw material needed to obtain a length of y. With other words, $p_{S(\ell)}(y)$ is the maximum usable length when the raw material has length y. Obviously, we have

$$y - \ell_{min} < p_{S(\ell)}(y) \leq y \quad \text{for all } y \geq \ell_{min}$$

and

$$p_{S(\ell)}(y) + p_{S(\ell)}(L - y) \leq p_{S(\ell)}(L) \quad \text{for all } y \in [0, L].$$

The knapsack problem KP(c, a, b) can be solved by means of the following backward dynamic programming (BDP) algorithm. If set T used in this algorithm contains at least all jump discontinuities of the optimal value function of KP(c, a, b), then Algorithm BDP provides a function $g : T \to \mathbb{Z}_+$ by which a solution of KP(c, a, b) can be easily determined. For example, $T := S(a, b)$ would do the job. Later on, it will turn out that Algorithm BDP can even successfully be used for solving knapsack problems if T contains only a certain subset of jump discontinuities.

Algorithm BDP
Input: c, a, b, T; Output: g
(1) Set $g(0) := 0$, $y := 0$.
(2) **While** $y < p_T(b)$ **do**
(3) $\quad y := s_T(y + 1)$,
(4) $\quad g(y) := \max\limits_{i \in I}\{c_i + g(p_T(y - a_i)) \mid y \geq a_i\}$.

Theorem 1. *Let T contain at least all jump discontinuities of the optimal value function f of KP(c, a, b). Then, if Algorithm BDP is used for determining $g : T \to \mathbb{Z}_+$, it holds*

$$f(y) = g(p_T(y)) \quad \textit{for all } y \in [0, b].$$

This well-known result can be also obtained for the following forward dynamic programming (FDP) algorithm.

Algorithm FDP
Input: c, a, b, T; Output: g
(1) Set $g(0) := 0$, $y := 0$.
(2) **While** $y \leq p_T(b - \min\{a_i \mid i \in I\})$ **do**
(3) $\quad$ **For all** $i \in I$ with $y + a_i \leq p_T(b)$ **do**
(4) $\quad\quad g(s_T(y + a_i)) := \max\{g(s_T(y + a_i)), c_i + g(y)\}$,
(5) $\quad \overline{y} := y$,
(6) $\quad$ **Repeat** $y := s_T(y + 1)$ **until** $g(\overline{y}) < g(y)$.

Note that $T = S(a,b)$ implies $s_T(y + a_i) = y + a_i \in T$ for all $i \in I$ and all $y \in T$ with $y + a_i \le b$. Thus, $s_T(y + a_i)$ can be replaced by $y + a_i$ in Step (4) of Algorithm FDP. Moreover, because of the repeat-loop in Step (6), some updates in Steps (3) and (4) can probably be saved compared to Step (4) of Algorithm BDP.

The worst-case complexity of both algorithms, BDP and FDP, is $O(b + m|T|)$ since y is increased at most b times, and at most m comparisons are done in the max-terms for each element of T. Thus, both are pseudo-polynomial algorithms.

In order to determine a reduced set of allocation points that is sufficient to obtain an optimal solution of KP(c,a,b) by Algorithm BDP or FDP we will apply some dominance condition (cf. [12, 16]). For that purpose we let $b > \max_{i\in I} a_i$ be satisfied. In view of the *separability* of the optimal value function f we have

$$f(b) = \max_{0<y\le b/2} \{f(y) + f(b-y)\}.$$

Since f is piecewise constant with jump discontinuities in $S(a)$ it further follows that

$$\begin{aligned} f(b) &= \max_{0<y\le b/2} \{f(p_{S(a)}(y)) + f(p_{S(a)}(b-y))\} \\ &= \max_{r\in S(a),\, 0<r\le b/2} \{f(r) + f(p_{S(a)}(b-r))\}. \end{aligned} \tag{6.5}$$

By $p_{S(a)}(r) \le p_{S(a)}(b - p_{S(a)}(b-r))$, we obtain

$$f(b) = \max_{r\in S(a),\, 0<r\le b/2} \{f(p_{S(a)}(b - p_{S(a)}(b-r))) + f(p_{S(a)}(b-r))\}.$$

This formula motivates the definition of the *reduced set of potential allocation points* by

$$S^{red}(a,b) := \{p_{S(a)}(b-r) \mid r \in S(a,b)\}.$$

Consequently, we have

$$f(b) = \max_{r\in T,\, 0<r\le b/2} \{f(r) + f(p_T(b-r))\} \quad \text{with} \quad T := S^{red}(a,b). \tag{6.6}$$

Theorem 2. *Let f denote the optimal value function of KP(c,a,b). If Algorithm BDP (or Algorithm FDP) with $T = S^{red}(a,b)$ is used to determine $g : T \to \mathbb{Z}_+$, then*

$$f(r) = g(r) \quad \textit{for all } r \in T \qquad \textit{and} \qquad f(y) \ge g(p_T(y)) \quad \textit{for all } y \in [0,b]$$

holds.

Due to $S^{red}(a,b) \subseteq S(a,b)$, the recursion (6.6) might be less expensive than the one in (6.5). Moreover, dependent on the instance, significant savings are possible if Algorithms BDP or FDP are applied for the solution of KP(c,a,b) with

Table 6.1 Potential allocation points for Example 1

y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(a,b)$	$\star$				$\star$			$\star$	$\star$	$\star$		$\star$	$\star$	$\star$	$\star$	$\star$
$S^{red}(a,b)$	$\star$				$\star$			$\star$	$\star$			$\star$				$\star$
$f(y)$	0				5			10		12		15		17	20	

$T = S^{red}(a,b)$. Note that using $S^{red}(a,b)$ instead of $S(a,b)$ is an application of the Nicholson-principle [8]. Since $p_{S(a)}(y) \geq p_{S^{red}(a,b)}(y)$ and $s_{S(a)}(y) \leq s_{S^{red}(a,b)}(y)$ for all $y \in [0,b]$, the application of Algorithm FDP with $T = S^{red}(a,b)$ does, in general, not any longer provide left-justified patterns.

Example 1. Let us consider the instance of the knapsack problem KP(c,a,b) with $c := (12,10,5)^\top$, $a := (9,7,4)^\top$, and $b := 15$. In Table 6.1, the elements of the sets $S(a,b)$ and $S^{red}(a,b)$ are marked by $\star$. Additionally, the optimal value function f is tabulated at their jump discontinuities. □

In Example 1, we have $|S^{red}(a,b)| < |S(a,b)| < b$. Moreover, we see that in this example the set of jump discontinuities is not a subset of $S^{red}(a,b)$. In general, the cardinality of $S(a,b)$ and $S^{red}(a,b)$ strongly depends on the input data. Nevertheless, there is a high potential to save memory and computation time by using $S^{red}(a,b)$ instead of $S(a,b)$. Moreover, the cardinality does not change if the unit of measure is changed, for instance from cm to mm.

Investigations how to compute $S(a,b)$ efficiently can be found in [3]. The computational amount for determining $S(a,b)$ and $S^{red}(a,b)$ is bounded from above by $O(mb)$. More precisely, it is bounded by $O(b + m|S(a,b)|)$ due to the application of Algorithm FDP for $KP(a,a,b)$.

6.3 The 1D Cutting Problem with Fix-Lengths

In the 1D case the presence of defective parts which are not allowed for any piece leads to independent smaller problem instances. Therefore, we consider only the scenario with different quality demands.

6.3.1 Problem Description

The following 1D cutting problem is considered: Pieces of various lengths ℓ_i, $i \in I$, and different quality demands $q \in Q$ have to be cut from a non-homogeneous raw material of length L in such a way that all allocation conditions (i.e., quality demands) are met and the total value of obtained pieces is maximal.

Such problems arise, for instance, in timber cutting. In that case, the length of the pieces is, in general, assumed to be variable within a given range but in this section we restrict the pieces to have fix-lengths, which is also of high interest. The more general case of pieces with variable lengths will be considered in the next section.

In order to formulate the cutting problem precisely, we consider several quality types $q \in Q$. To each quality type $q \in Q$ there is at least one piece $i \in I$ which is of this type. Conversely, each piece $i \in I$ is assigned to a quality type $q(i) \in Q$. Let $I_q \subset I$ denote the set of all pieces of quality type q, i.e., $I_q = \{i \in I \mid q(i) = q\}$. Hence, we have $\cup_{q \in Q} I_q = I$ and $I_q \cap I_p = \emptyset$ for $p, q \in Q$ with $q \neq p$.

Intervals of the raw material where exactly one quality demand is fulfilled will be called *allocation intervals* $A_k \subset [0, L]$ with $k \in K := \{1, \ldots, |K|\}$. These intervals are considered as given. The quality demand satisfied in A_k is denoted by $\widetilde{q}(k) \in Q$. Any allocation interval A_k can be described by

$$A_k := [b_k, e_k] \subseteq [0, L] \quad \text{with} \quad e_k - b_k \geq \min\{\ell_i \mid i \in I_{\widetilde{q}(k)}\}.$$

It is possible that different allocation intervals are of the same quality type. Two intervals A_j and A_k ($j \neq k$) may overlap, even if they fulfill the same quality demand. In the latter case, we assume $A_j \not\subseteq A_k$. Without loss of generality it can be assumed that

$$b_1 \leq b_2 \leq \cdots \leq b_{|K|} \quad \text{and} \quad e_k \leq e_{k+1} \text{ if } b_k = b_{k+1}.$$

For any piece $i \in I$ we require without loss of generality that there is a

$$k \in K_{q(i)} := \{k \in K \mid \widetilde{q}(k) = q(i)\}$$

so that $\ell_i \leq e_k - b_k$. This means, any piece $i \in I$ with *allocation point* $y_i \in [b_k, e_k - \ell_i]$ will be completely contained in an allocation interval with quality type $q(i)$.

For example, in hardwood cutting a quality demand could be that no more than one sound knot per reference length is allowed. Then, the occurrence of two sound knots within the reference length causes two partially overlapping allocation intervals.

The value of a piece $i \in I$ is again denoted by γ_i. In general, pieces of the same quality type may be obtained several times from the raw material, either from the same allocation interval (if it is sufficiently large) or from different allocation intervals of this quality type.

The cutting problems we consider consist of determining a (cutting) pattern that has a maximal total value of the obtained pieces.

A pattern π can be described by a (finite) sequence of triples (i_t, y_t, k_t), $t = 1, \ldots, t_\pi$ with $y_t + \ell_{i_t} \leq y_{t+1}$ for $t = 1, \ldots, t_\pi - 1$ where i_t denotes the index of the t-th placed piece, y_t is the allocation point of piece i_t, and k_t gives the corresponding allocation interval. Hence,

$$0 \le y_1,\ y_1 + \ell_{i_1} \le y_2, \cdots, y_{t_\pi} + \ell_{i_{t_\pi}} \le L,$$
$$b_{k_t} \le y_t, \quad y_t + \ell_{i_t} \le e_{k_t}, \quad q(i_t) = \widetilde{q}(k_t) \qquad \text{for } t = 1, \ldots, t_\pi.$$

Obviously, all allocation points can be considered to be integral.

6.3.2 Modeling

In order to formulate an integer optimization model it is assumed in this subsection that each piece is packed at most once. This can be done without loss of generality defining piece i several times with different indexes. We describe the allocation of a piece $i \in I$ by means of a 0/1-variable z_i as follows:

$$z_i := \begin{cases} 1 & \text{if piece } i \in I \text{ is allocated (should be cut),} \\ 0 & \text{otherwise.} \end{cases}$$

If the allocation point of piece i is at y_i, the piece covers the interval $T_i(y_i) := [y_i, y_i + \ell_i]$ but only if it has been packed, i.e., if $z_i = 1$. Hence, the allocation problem can be modeled as follows where $\operatorname{int} A$ denotes the interior of set A:

$$\sum_{i \in I} \gamma_i \cdot z_i \to \max \tag{6.7}$$

subject to

$$z_i \in \{0, 1\},\ y_i \in \mathbb{Z}_+, \quad \text{for all } i \in I, \tag{6.8}$$

$$\operatorname{int} T_i(y_i) \cap \operatorname{int} T_j(y_j) = \emptyset \qquad \text{for all } i, j \in I \text{ with } i < j \text{ and } z_i + z_j = 2, \tag{6.9}$$

$$\begin{gathered} \text{for each } i \in I \text{ with } z_i = 1 \text{ there are } q \in Q \text{ and } k \in K_q \\ \text{with } i \in I_q \text{ and } T_i(y_i) \subseteq A_k. \end{gathered} \tag{6.10}$$

Condition (6.9) ensures that the packed pieces do not overlap each other. Condition (6.10) guarantees that the allocation is done within an appropriate allocation interval.

To transform the previous model into an integer linear program (ILP) we define 0/1-variables u_{ij}, $i, j \in I$, $i < j$, and v_{ik}, $i \in I$, $k \in K_{q(i)}$ as follows:

$$u_{ij} := \begin{cases} 0 & \text{if piece } i \text{ is packed left to piece } j, \text{ i.e., } y_i + \ell_i \le y_j, \\ 1 & \text{if piece } i \text{ is packed right to piece } j, \text{ i.e., } y_j + \ell_j \le y_i, \end{cases}$$

and

$$v_{ik} := \begin{cases} 1 & \text{if piece } i \in I \text{ is packed within } A_k, \\ 0 & \text{otherwise.} \end{cases}$$

Conditions (6.9) and (6.10) can now be replaced by

$$\begin{aligned} y_i + \ell_i &\le y_j + L(2 - z_i - z_j + u_{ij}) \quad \text{for all } i,j \in I \text{ with } i < j, \\ y_j + \ell_j &\le y_i + L(3 - z_i - z_j - u_{ij}) \quad \text{for all } i,j \in I \text{ with } i < j, \end{aligned} \tag{6.11}$$

$$\begin{aligned} y_i &\ge b_k + L(z_i + v_{ik} - 2) \quad \text{for all } i \in I,\ k \in K_{q(i)}, \\ y_i + \ell_i &\le e_k + L(2 - z_i - v_{ik}) \quad \text{for all } i \in I,\ k \in K_{q(i)}, \end{aligned} \tag{6.12}$$

$$y_i \le L z_i \quad \text{for all } i \in I, \tag{6.13}$$

$$\sum_{k \in K_{q(i)}} v_{ik} = z_i \quad \text{for all } i \in I, \tag{6.14}$$

$$\begin{aligned} u_{ij} &\in \{0,1\} \quad \text{for all } i,j \in I \text{ with } i < j, \\ v_{ik} &\in \{0,1\} \quad \text{for all } i \in I, k \in K_{q(i)}. \end{aligned} \tag{6.15}$$

Conditions (6.11) are redundant if piece i or j is not allocated. Otherwise, if $z_i + z_j = 2$, because of $u_{ij} \in \{0,1\}$ one of the two conditions in (6.11) is non-trivial. If item i is not packed, or if i is not packed within A_k, then restrictions (6.12) are redundant. If an item is not used, then condition (6.13) ensures that the allocation point of this item is set 0. By (6.14) it is required that a corresponding allocation interval exists if item i is packed.

The number of binary and integer variables can become very large in general. To solve the ILP (6.7), (6.8), (6.11)–(6.15) within a real time application scenario, we will describe B&B or DP approaches. B&B with depth first search (LIFO) has the advantage that good feasible solutions are found quickly so that time termination criteria can be applied. For a DP algorithm the computational amount (run time needed to solve an instance) can be well estimated because of its pseudo-polynomiality.

6.3.3 Sets of Potential Allocation Points

For the allocation problem (6.7)–(6.10) the optimal value function $v : [0, L] \to \mathbb{Z}_+$ is defined by

$$v(y) := \max_{y,z} \left\{ \sum_{i \in I} \gamma_i z_i \mid \text{(6.8)–(6.10) hold and } y_i + \ell_i \le y \text{ for all } i \in I \text{ with } z_i = 1 \right\}$$

for all $y \in [0, L]$. The function v is non-decreasing since the feasible region enlarges if y increases. The optimal value function is piecewise constant since only a finite number of different sequences of packed pieces exists. Moreover, v is continuous from the right.

In case that

$$(0, L) \setminus \bigcup_{k \in K} \operatorname{int} A_k \neq \emptyset$$

the packing problem can be separated into some smaller problems which can be solved independently from each other. The optimal value of the original problem is the sum of the optimal values of the smaller problems. In the following it is always assumed that the packing problem is not separable, i.e.,

$$\bigcup_{k \in K} \operatorname{int} A_k = (0, L). \tag{6.16}$$

Hence, $b_1 = 0$ and $\max\{e_k : k \in K\} = L$.

Let $S^*(\gamma, \ell)$ denote the set of jump discontinuities of v for an instance with input data $L \in \mathbb{Z}_>$, $\ell \in \mathbb{Z}_>^m$, $\gamma \in \mathbb{Z}_>^m$, and $A_k = [b_k, e_k]$ for $k \in K$. Our aim is to find a superset of $S^*(\gamma, \ell)$ which is independent of γ and as small as possible.

Theorem 3. *For any $\ell \in \mathbb{Z}_>^m$ and any $\gamma \in \mathbb{Z}_>^m$, the inclusion*

$$S^*(\gamma, \ell) \subseteq S_{ap}(\ell) := \bigcup_{k \in K} (b_k \oplus S(\ell)) \cap [0, L].$$

is fulfilled where $b_k \oplus S(\ell) := \{y \mid y = b_k + r,\ r \in S(\ell)\}$.

Note that set $S_{ap}(\ell)$ does not only depend on ℓ but also on the given allocation intervals. For simplicity we do not show this dependence in the notation of $S_{ap}(\ell)$ and of other sets that will be defined later.

Proof. To each jump discontinuity of the optimal value function belongs a left-justified pattern. Any left-justified pattern has allocation points only at the beginning of an allocation interval, i.e. at b_k for some $k \in K$, or at points $b_k + r$ with $r \in S(\ell)$. All these points define set $S_{ap}(\ell)$. □

Corollary 1. *For all $r \in [0, L-1] \cap \mathbb{Z}$ and all $y \in (r, s_{S_{ap}(\ell)}(r+1))$ we have $v(y) = v(r) = v(p_{S_{ap}(\ell)}(y))$.*

In general, we can even obtain a set $\widehat{S}_{ap}(\ell)$ that is smaller than $S_{ap}(\ell)$ but still allows to obtain an optimal solution of problem (6.7)–(6.10). To this end, a more sophisticated procedure is used. Its basic principle is the construction of all possible combinations (patterns) in dependence of the quality demands and the corresponding items. For example, if we have an allocation interval A_k with $b_k = 0$ and an item $i \in I_{\widetilde{q}(k)}$ with $\ell_i > e_k$, then item i cannot be placed with allocation point 0. Therefore, it might happen that $\ell_i \in S(\ell)$ is not a jump discontinuity of v. A similar situation arises if, for item i, no allocation interval A_k with $q(i) = \widetilde{q}(k)$ and $b_k = 0$ exists. Then, item i cannot be placed with allocation point 0. Therefore, in general, the use of $S(\ell)$ in the definition of $S_{ap}(\ell)$ leads to a proper superset of the jump discontinuities.

The description of the procedure to obtain a reduced set $\widehat{S}_{ap}(\ell)$ which still contains all jump discontinuities of v requires some notation. For $y \in \{b_k : k \in K\}$, let

$$K_b(y) := \{k \in K \mid b_k = y\}, \quad Q_b(y) := \{q \in Q \mid q = \widetilde{q}(k),\ k \in K_b(y)\},$$

and, for $y \in [0, L)$, let

$$\begin{aligned} Q(y) &:= \{q \in Q \mid \exists k \in K_q \text{ with } b_k \le y,\ y + \min\{\ell_i \mid i \in I_q\} \le e_k\}, \\ k(y,q) &:= \max\{k \in K_q \mid b_k \le y\} \quad \text{for all } q \in Q(y), \\ K(y) &:= \{k \in K \mid k = k(y,q),\ q \in Q(y)\} \end{aligned}$$

be defined. The set $Q(y)$ represents all quality types $q \in Q$ for which a sufficiently large allocation interval A_k with $\widetilde{q}(k) = q$ exists such that a piece $i \in I_q$ with allocation point y can be obtained. If for $q \in Q(y)$ several allocation intervals contain points y and $y + \min\{\ell_i \mid i \in I_q\}$, then we take that with largest b_k and collect them in $K(y)$.

Then the procedure to construct the set $\widehat{S}_{ap}(\ell)$ starts at $\widehat{y} := 0$. Then, we begin to construct $\widehat{y} \oplus S(\ell)$ by successively adding the lengths of those items which can be placed because of an existing allocation interval. At each point b_k, $k \in K$, where an allocation interval begins, the construction of $b_k \oplus S(\ell)$ restricted to feasible placements has to be started.

Initialization The first jump discontinuities can arise when a leftmost piece is allocated at point $\widehat{y} := 0$:

$$\widehat{S} := \{\ell_i \mid i \in I_q,\ \ell_i \le e_{k(0,q)},\ q \in Q_b(0)\} \cup \{0\}.$$

Since $Q_b(0)$ represents all qualities having an allocation interval beginning at 0, all pieces of I_q, $q \in Q_b(0)$, can be placed at 0 which fit within the corresponding allocation interval, i.e., which are not longer than $e_{k(0,q)}$.

General Step Let

$$\widehat{y}_b := \min\{L;\ b_k \mid b_k > \widehat{y},\ k \in K\}, \qquad \widehat{y}_s := \min\{y \in \widehat{S} \mid y > \widehat{y}\}.$$

Here, $\widehat{y}_b$ denotes the coordinate of the next allocation interval which allows the placement of further items, whereas for $\widehat{y}_s$ there is already a feasible pattern which can possibly be extended.

If $\widehat{y}_b < \widehat{y}_s$, then $\widehat{y} := \widehat{y}_b$ and, because of the new allocation interval, all corresponding pieces are placed:

$$\widehat{S} := \widehat{S} \cup \{\widehat{y}\} \cup \{\widehat{y} + \ell_i \mid i \in I_q,\ \widehat{y} + \ell_i \le e_{k(\widehat{y},q)},\ q \in Q_b(\widehat{y})\}.$$

Otherwise, if $\widehat{y}_b \geq \widehat{y}_s$, then $\widehat{y} := \widehat{y}_s$ and all pieces belonging to $Q(\widehat{y})$ and of suitable length are placed:

$$\widehat{S} := \widehat{S} \cup \{\widehat{y} + \ell_i \mid i \in I_q,\ \widehat{y} + \ell_i \leq e_{k(\widehat{y},q)},\ q \in Q(\widehat{y})\}.$$

The algorithm terminates if no further piece can be placed, i.e., if

$$\widehat{y} > L - \min\{\ell_i \mid i \in I_L\}, \quad \text{where } I_L := \bigcup_{k \in K: e_k = L} I_{q(k)}.$$

Then, the set $\widehat{S}_{ap}(\ell)$ is given by the lastly obtained $\widehat{S}$.

The time for determining $\widehat{S}_{ap}(\ell)$ is bounded by $O((|K| + m)|\widehat{S}_{ap}(\ell)|)$ since $\widehat{y}$ is increased at most $|\widehat{S}_{ap}(\ell)|$ times and, for each such $\widehat{y}$, the identification of $Q_b(\widehat{y})$ or $Q(\widehat{y})$ costs at most $O(|K|)$ and not more than m pieces are considered.

According to the previous procedure the next result follows.

Theorem 4. *For any $\ell \in \mathbb{Z}^m_>$ and any $\gamma \in \mathbb{Z}^m_>$, the inclusions*

$$S^*(\gamma, \ell) \subseteq \widehat{S}_{ap}(\ell) \subseteq S_{ap}(\ell).$$

hold.

Thus, in analogy to Theorem 1, it is sufficient to use $\widehat{S}_{ap}(\ell)$ for a recursion based on DP for solving the 1D cutting problem with fix-lengths. To this end, let $T := \widehat{S}_{ap}(\ell)$ be defined.

Algorithm FDP-FL
Input: γ, ℓ, L, T; Output: g
(1) Set $g(0) := 0$, $y := 0$.
(2) **While** $y \leq p_T(L - \min\{\ell_i : i \in I_L\})$ **do**
(3) **For all** $k \in K(y)$ and all $i \in I_{q(k)}$ with $y + \ell_i \leq e_k$ **do**
(4) $g(s_T(y + \ell_i)) := \max\{g(s_T(y + \ell_i)), \gamma_i + g(y)\}$,
(5) $\overline{y} := y$,
(6) **Repeat** $y := s_T(y + 1)$ **until** $g(\overline{y}) < g(y)$ or $y \in \{b_k \mid k \in K\}$.

The worst-case complexity of Algorithm FDP-FL is $O((|K| + m)|T|)$ since y is increased at most $|T|$ times and, for each such y, the identification of $K(y)$ needs $O(|K|)$ time and at most m pieces are considered.

Theorem 5. *If Algorithm FDP-FL is used with $T := \widehat{S}_{ap}(\ell)$ to determine $g : T \to \mathbb{Z}_+$, then it holds*

$$v(y) = g(p_T(y)) \quad \textit{for all } y \in [0, L],$$

where v is the optimal value function of problem (6.7)–(6.10).

Moreover, the set $\widehat{S}_{ap}(\ell)$ can be advantageously applied within B&B approaches for solving the 1D cutting problem with fix-lengths (6.7)–(6.10).

6.3.4 Applying the Nicholson Principle

In the following we apply the Nicholson principle [8, 16] to obtain a further reduction of the sets $S_{ap}(\ell)$ and $\widehat{S}_{ap}(\ell)$ of potential allocation points. Let

$$S^{\leftarrow}_{ap}(\ell) := \bigcup_{k\in K}(e_k \ominus S(\ell)) \cap [0, L] \quad \text{where } e_k \ominus S(\ell) := \{y \mid y = e_k - r,\ r \in S(\ell)\}.$$

denote the set of potential allocation points maximal in the following sense: for any $y \in S^{\leftarrow}_{ap}(\ell)$ there is a combination of piece lengths whose first (leftmost) piece, say i, has allocation point y and which is not feasible for allocation points for i larger than y. Then, a first reduced set of allocation points is obtained by the Nicholson principle as follows:

$$S^{red}_{ap}(\ell) := \{p_T(y) \mid y \in S^{\leftarrow}_{ap}(\ell)\} \quad \text{with} \quad T := S_{ap}(\ell).$$

Theorem 6. *If Algorithm FDP-FL is used with* $T := S^{red}_{ap}(\ell)$ *to determine* $g : T \to \mathbb{Z}_+$*, then it holds*

$$v(y) = g(y) \quad \textit{for all } y \in T,$$

where v *is the optimal value function of problem* (6.7)–(6.10).

Proof. Similar to the optimal value function v defined in Sect. 6.3.3 for the allocation problem (6.7)–(6.10), but now looking from L to 0, we can define another optimal value function $\tilde{v} : [0, L] \to \mathbb{Z}_+$ by

$$\tilde{v}(y) := \max_{y,z} \left\{ \sum_{i\in I} \gamma_i z_i \mid \text{(6.8)–(6.10) hold and } y_i \geq y \text{ for all } i \in I \text{ with } z_i = 1 \right\}.$$

The function $\tilde{v}$ is non-increasing since the feasible region shrinks if l increases. Since $v(L)$ and $\tilde{v}(0)$ are the optimal values of the same problem, obviously we have $v(L) = \tilde{v}(0)$. Let the sequence of triples (i_t, y_t, k_t), $t = 1, \ldots, t_\pi$ with $y_t + \ell_{i_t} \leq y_{t+1}$ for all t represent any normalized optimal pattern π of problem (6.7)–(6.10). If π does not consist of a single piece with length L, then there exists $y^* \in (0, L) \cap S_{ap}(\ell)$ with

$$v(L) = \tilde{v}(0) = v(y^*) + \tilde{v}(y^*) = \max\{v(y) + \tilde{v}(y) \mid y \in [0, L]\}.$$

As y^* any element in $\{y_t, y_t + \ell_{i_t} \mid t = 1, \ldots, t_\pi\} \cap (0, L)$ can be taken. Therefore, we have

$$\max\{v(y) + \tilde{v}(y) \mid y \in [0, L]\} = \max\{v(y) + \tilde{v}(y) \mid y \in S_{ap}(\ell)\}$$

but we have to prove

$$\max\{v(y) + \tilde{v}(y) \mid y \in S_{ap}(\ell)\} = \max\{v(y) + \tilde{v}(y) \mid y \in S_{ap}^{red}(\ell)\}.$$

To see this, we assume there exist $r \in S_{ap}^{red}(\ell)$ and $y \in S_{ap}(\ell) \setminus S_{ap}^{red}(\ell)$ with $r < y < s_T(r+1) =: r'$ and $v(y) + \tilde{v}(y) > \max\{v(y) + \tilde{v}(y) \mid y \in S_{ap}^{red}(\ell)\}$.

Assuming $v(y) = v(r)$ then $v(r) + \tilde{v}(r) \geq v(y) + \tilde{v}(y)$ since $\tilde{v}$ is non-increasing. Hence, we have $v(r) < v(y)$.

Assuming $\tilde{v}(y) = \tilde{v}(r')$ then $v(r') + \tilde{v}(r') \geq v(y) + \tilde{v}(y)$ since v is non-decreasing.

It remains the case that $\tilde{v}(y) > \tilde{v}(r')$. Then there is $y' \in S_{ap}^{\leftarrow}(\ell)$ with $y' \geq y$ and $\tilde{v}(y) = \tilde{v}(y')$. Since $y \in S_{ap}(\ell)$ and $y' \in S_{ap}^{\leftarrow}(\ell)$ with $y' \geq y$ we have a contradiction to $y \notin S_{ap}^{red}(\ell)$. □

Corollary 2. *Among all optimal solutions of problem* (6.7)–(6.10) *there is a cutting pattern whose allocation points are all in* $S_{ap}^{red}(\ell)$.

In order to further reduce the set $S_{ap}^{red}(\ell)$ we will apply the Nicholson principle again by using $\widehat{S}_{ap}(\ell)$ instead of $S_{ap}(\ell)$. In analogy to the construction of $\widehat{S}_{ap}(\ell)$ in the previous subsection, a set $\widehat{S}_{ap}^{\leftarrow}(\ell)$ of rightmost allocation points can be constructed. Only those items are regarded which can be placed because of the existence of a corresponding allocation interval. For any $y \in \widehat{S}_{ap}^{\leftarrow}(\ell)$, there is a feasible pattern π, i.e., a sequence of triples (i_t, y_t, k_t), $k = 1, \ldots, t_\pi$ with $y_t + \ell_{i_t} \leq y_{t+1}$ for all t, whose first (leftmost) piece i_1 has allocation point $y_1 = y$. This pattern becomes infeasible for all y' with $y' > y$ if $y_1 := y'$ and any choice of the allocation points $y_2, \ldots, y_{t_\pi}$ with $y_t + \ell_{i_t} \leq y_{t+1}$ for all t. The construction requires some notation. For $y \in \{e_k \mid k \in K\}$, let

$$K_e(y) := \{k \in K \mid e_k = y\}, \quad Q_e(y) := \{q \in Q \mid q = \widetilde{q}(k),\ k \in K_e(y)\},$$

and, for $y \in (0, L]$, let

$$\begin{aligned}
\overline{Q}(y) &:= \{q \in Q : \exists k \in K_q \text{ with } e_k \geq y,\ y - \min\{\ell_i \mid i \in I_q\} \geq b_k\},\\
\overline{k}(y, q) &:= \min\{k \in K_q \mid e_k \geq y\} \quad \text{for all } q \in \overline{Q}(y),\\
\overline{K}(y) &:= \{k \in K \mid k = \overline{k}(y, q),\ q \in \overline{Q}(y)\}.
\end{aligned}$$

be defined.

Initialization Rightmost allocation points are obtained if a piece is allocated at a point in

$$\widehat{S}^{\leftarrow} := \{L - \ell_i \mid i \in I_q,\ \ell_i \leq L - b_{\overline{k}(L,q)},\ q \in Q_e(L)\} \cup \{L\}.$$

Since $Q_e(L)$ represents all qualities having an allocation interval ending at L, all pieces of I_q, $q \in Q_e(L)$, can be placed at $L - \ell_i$ which fit within the corresponding allocation interval, i.e. which are not longer than $L - b_{\overline{k}(L,q)}$. Let $\widehat{y} := L$.

General Step Let

$$\widehat{y}_e := \max\{0,\ e_k \mid e_k < \widehat{y},\ k \in K\}, \quad \widehat{y}_s := \max\{y \in \widehat{S}^{\leftarrow} \mid y < \widehat{y}\}.$$

Here, $\widehat{y}_e$ denotes the coordinate of the next allocation interval which allows the placement of further items, whereas for $\widehat{y}_s$ there is already a feasible pattern in the interval $[\widehat{y}_s, L]$ which can possibly be extended. If $\widehat{y}_e > \widehat{y}_s$, then $\widehat{y} := \widehat{y}_e$ and, because of the new allocation interval, all corresponding pieces are placed:

$$\widehat{S}^{\leftarrow} := \widehat{S}^{\leftarrow} \cup \{\widehat{y} - \ell_i \mid i \in I_q,\ \widehat{y} - \ell_i \geq b_{\overline{k}(\widehat{y},q)},\ q \in Q_e(\widehat{y})\}.$$

Otherwise, if $\widehat{y}_e \leq \widehat{y}_s$, then $\widehat{y} := \widehat{y}_s$ and all pieces belonging to $\overline{Q}(\widehat{y}_s)$ and suitable length are placed:

$$\widehat{S}^{\leftarrow} := \widehat{S}^{\leftarrow} \cup \{\widehat{y} - \ell_i \mid i \in I_q,\ \widehat{y} - \ell_i \geq b_{\overline{k}(\widehat{y},q)},\ q \in \overline{Q}(\widehat{y})\}.$$

The algorithm terminates if no further piece can be placed, i.e., if

$$\widehat{y} < \min\{\ell_i \mid i \in I_0\}, \quad \text{where } I_0 := \bigcup_{k \in K: b_k = 0} I_{q(k)}.$$

Then, the set $\widehat{S}^{\leftarrow}_{ap}(\ell)$ is given by the lastly obtained $\widehat{S}^{\leftarrow}$.

The time to determine $\widehat{S}^{\leftarrow}_{ap}(\ell)$ is similar to that needed for $\widehat{S}_{ap}(\ell)$ and is bounded from above by $O(mL)$.

Now, we are able to define the announced reduced set of allocation points by

$$\widehat{S}^{red}_{ap}(\ell) := \{p_T(y) \mid y \in \widehat{S}^{\leftarrow}_{ap}(\ell)\} \quad \text{with} \quad T := \widehat{S}_{ap}(\ell) \quad \text{(cf. Sect. 6.3.3)}.$$

Now, because of construction, we have $\widehat{S}^{red}_{ap}(\ell) \subseteq \widehat{S}_{ap}(\ell)$ and moreover

Theorem 7. *If Algorithm FDP-FL with* $T := \widehat{S}^{red}_{ap}(\ell)$ *is used to determine* $g : T \to \mathbb{Z}_+$, *then it holds*

$$v(y) = g(y) \quad \textit{for all} \quad y \in T,$$

where v *is the optimal value function of problem* (6.7)–(6.10).

The theorem can be proved in analogy to Theorem 6. The time needed for computing all g-values according to Theorem 7 is bounded by $O(m|\widehat{S}^{red}_{ap}(\ell)|)$.

Corollary 3. *Among all optimal solutions of problem* (6.7)–(6.10) *there is a cutting pattern whose allocation points are all in* $\widehat{S}^{red}_{ap}(\ell)$.

Similar to Theorem 2, the use of $S_{ap}^{red}(\ell)$ or $\widehat{S}_{ap}^{red}(\ell)$ does not guarantee to obtain all values $v(y)$ of the optimal value function v for $y \in (0, L)$, nevertheless $v(L)$ and a corresponding optimal pattern can be determined.

Example 2. Let the unit of measurement be millimeter. An arbitrarily long wooden board of width $W = 300$ has to be cut into strips with widths of 40, 50, or 60. The cutting kerf is 2.5. Due to different quality demands, strips of width 50 can only be obtained within the interval $[50, 200]$, and strips of width 60 only within $[150, 225]$. Multiplying all data by 2, adding 5 to the item widths and to the overall width to regard the kerf, and dividing all widths by 5 leads to a 1D cutting problem with the following input data: $L = 121$, item lengths $\ell_1 = 17$, $\ell_2 = 21$, $\ell_3 = 25$, and allocation intervals $A_1 = [0, 121]$, $A_2 = [20, 80]$, $A_3 = [60, 90]$. Then, we obtain $|S(\ell, L)| = 41$, $|S^{red}(\ell, L)| = 23$,

$$|S_{ap}(\ell)| = 67, \quad |S_{ap}^{\leftarrow}(\ell)| = 73, \quad |S_{ap}^{red}(\ell)| = 35,$$
$$|\widehat{S}_{ap}(\ell)| = 29, \quad |\widehat{S}_{ap}^{\leftarrow}(\ell)| = 36, \quad |\widehat{S}_{ap}^{red}(\ell)| = 12,$$

and

$$\widehat{S}_{ap}^{red}(\ell) = \{0, 17, 20, 34, 41, 51, 58, 62, 68, 87, 104, 121\}.$$

□

The computation of any of the introduced sets of potential allocation points takes a pseudo-polynomial amount of time. Due to its smaller cardinality, the application of $\widehat{S}_{ap}^{red}(\ell)$ can save computational effort in DP and B&B approaches if compared to the use of other sets of allocation points. If instances have to be solved which only differ in the profit coefficients γ, the construction of $\widehat{S}_{ap}^{red}(\ell)$ has to be done only once.

6.4 The 1D Cutting Problem with Variable Lengths

In some cutting tasks the lengths of desired items should not be fixed in advance. Instead, they can vary within known tolerances. For example, this is useful for producing finger joined lumber. There, items of various lengths (but with the same profile) are put together to obtain stripes of desired lengths.

6.4.1 Problem Formulation

Now, in contrast to the previous section, the lengths of the items to be cut are not fixed. Rather, it can take any value within a given range. More precisely, the length of piece i ($i \in I$) is again denoted by ℓ_i. However, ℓ_i is now a variable with

$$\ell_i \in [\underline{l}_i, \bar{l}_i] \quad (i \in I),$$

where $\underline{l}_i$ and $\bar{l}_i$ are given positive integers with $0 < \underline{l}_i \leq \bar{l}_i$, $i \in I$. Items with fix-lengths can also be considered by simply setting $\underline{l}_i = \bar{l}_i$.

The value of item i with length l is denoted by $\tilde{\gamma}_i(l)$. The function $\tilde{\gamma}$ is required to be affine, non-decreasing, and non-negative. For the sake of simplicity, we assume $\tilde{\gamma}_i(l) = \gamma_i \cdot l$ with some given $\gamma_i > 0$ for all $i \in I$.

Problems of this kind occur, for instance, related to hard wood cutting. There, pieces of various lengths (but with the same cross section) are put together using the finger-joining technology to get profiles of arbitrary length (see, e.g., [1]).

6.4.2 Modeling

In order to formulate a mixed-integer optimization model with 0/1-variables it is assumed in this subsection that each piece is allocated at most once (as in Sect. 6.3.2). The allocation of piece i is described by a 0/1-variable z_i defined as follows:

$$z_i = \begin{cases} 1 & \text{if piece } i \in I \text{ is allocated (should be cut),} \\ 0 & \text{otherwise.} \end{cases}$$

The allocation point of piece i is again denoted by y_i. Then piece i with length ℓ_i covers the interval $T_i(y_i, \ell_i) := [y_i, y_i + \ell_i]$ if it has been placed, i.e., if $z_i = 1$. Hence, the cutting (allocation) problem can be modeled as follows:

$$\sum_{i \in I} \gamma_i \cdot \ell_i \cdot z_i \to \max \tag{6.17}$$

subject to

$$z_i \in \{0, 1\},\ \ell_i, y_i \in \mathbb{R}_+ \quad i \in I, \tag{6.18}$$

$$\underline{l}_i z_i \leq \ell_i \leq \bar{l}_i z_i \quad i \in I, \tag{6.19}$$

$$\operatorname{int} T_i(y_i, \ell_i) \cap \operatorname{int} T_j(y_j, \ell_j) = \emptyset \quad \text{for all } i, j \in I \text{ with } i \neq j, \tag{6.20}$$

$$\begin{gathered} \text{for each } i \in I \text{ with } z_i = 1 \text{ there are } q \in Q \text{ and } k \in K_q \\ \text{with } i \in I_q \text{ and } T_i(y_i, \ell_i) \subseteq A_k. \end{gathered} \tag{6.21}$$

Condition (6.20) ensures that the packed pieces do not overlap each other and condition (6.21) guarantees that the packing of a piece is done within an allocation interval of related quality.

Note that the optimization model (6.17)–(6.21) has a nonlinear objective function. Similar to Sect. 6.3.2, the restrictions (6.20) and (6.21) can be linearized using the same 0/1-variables u_{ij}, $i, j \in I$, $i < j$, and v_{ik}, $i \in I$, $k \in K_{q(i)}$.

6.4.3 Optimal Value Function

For the optimization problem (6.17)–(6.21) the optimal value function $v : [0, L] \to \mathbb{R}_+$ is defined by

$$v(y) := \max_{z,y,\ell} \left\{ \sum_{i \in I} \gamma_i \ell_i z_i \mid \text{(6.18)–(6.21) hold and } y_i + \ell_i \le y \text{ for all } i \text{ with } z_i = 1 \right\}.$$

The function v is continuous from the right and non-decreasing since the feasible region enlarges if y increases. Moreover, v is piecewise affine since only a finite number of different sequences of allocated pieces exists and the functions $\tilde{\gamma}_i$ providing the profit of pieces $i \in I$ were assumed to be linear. By the same reason, the domain of v can be partitioned into intervals where v is either constant or linearly increasing with slope in $\{\gamma_1, \ldots, \gamma_m\}$.

Example 3. Let the following instance of a cutting problem be given:

$$\begin{array}{ll} I := \{1,2,3\}, & Q := \{1,2\}, \\ \underline{l}_1 := \underline{l}_2 := 30, \ \bar{l}_1 := \bar{l}_2 := 50, & \underline{l}_3 := 20, \ \bar{l}_3 := 100, \\ \gamma_1 := \gamma_2 := 8, & \gamma_3 := 5, \\ A_1 := [0,60], & A_2 := [70,100], \ A_3 := [0,100], \\ I_1 := \{1,2\}, & I_2 := \{3\}, \\ K_1 := \{1,2\}, & K_2 := \{3\}. \end{array}$$

Figure 6.1 shows the optimal value function v for Example 3. The leftmost gap of v

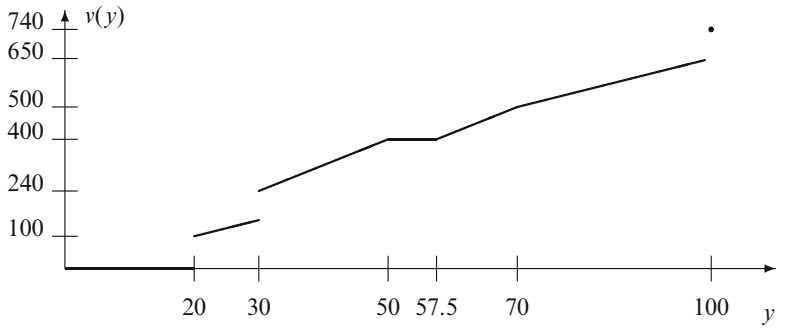

Fig. 6.1 Optimal value function v for Example 3

at $y = 20$ arises since item 3 with length $\ell_3 = \underline{l}_3 = 20$ and allocation point $y_3 = 0$ is placed. This leads to $v(20) = \gamma_3\ell_3 = 100$. For $y \in [20, 30)$, placing item 3 with length $\ell_3 = y$ and $y_3 = 0$ provides $v(y) = \gamma_3\ell_3 = 5y$. If $y = 30$, then item 1 (instead of item 3) is placed with $\ell_1 = 30$ and $y_1 = 0$. For $y \in [30, 50)$, placing item 1 with length $\ell_1 = y$ and $y_1 = 0$ yields $v(y) = \gamma_1\ell_1 = 8y$. If $y \in [50, 57, 5)$ then placing item 1 with $y_1 \in [0, y - 50]$ is optimal so that v remains constant in this interval. For $y \in [57.5, 70)$, both item 1 and item 3 are placed with $y_1 = 0$, $\ell_1 = y-20, y_3 = y-20, \ell_3 = 20$. This yields $v(y) = \gamma_1\ell_1+\gamma_3\ell_3 = 8(y-20)+100$. For $y \in [70, 100)$, the length of item 1 becomes maximal, namely $\ell_1 = \bar{l}_1 = 50$ with $y_1 = 0$ and the length of item 3 is $\ell_3 = y - 50$ with $y_3 = 50$. Therefore, $v(y) = 400 + 5(y - 50)$. Finally, the rightmost gap occurs at $y = 100$ because of the optimal pattern with $y_1 = 0, y_2 = 70, y_3 = 50, \ell_1 = 50, \ell_2 = 30, \ell_3 = 20$ and $v(100) = 740$. □

Note that the allocation pattern with the optimal value $v(L)$ might be not unique. For example, the optimal lengths of two items of the same quality need not be unique but their sum is the same for all optimal patterns with the same sequence of items.

Since y_i and ℓ_i are non-negative **real numbers,** infinitely many points become potential allocation points. However, the subsequent theorem shows that a finite subset of allocation points suffices to define an optimal (allocation) pattern.

In case of variable lengths, a pattern π is a finite sequence of quadruples $(i_t, y_t, \ell_t, k_t)_{t=1}^{t_\pi}$, where i_t denotes the index of the t-th placed piece, y_t is the allocation point of piece i_t, ℓ_t is its length, and k_t gives the corresponding allocation interval.

Theorem 8. *Among all optimal patterns for problem* (6.17)–(6.21) *there is a pattern* π *with*

$$y_t \in S_{ap}(\underline{l}, \bar{l}) \quad \text{for all } t = 1, \dots, t_\pi$$

where $\mathscr{A} := \{b_k, e_k \mid k \in K\}$ *and*

$$S_{ap}(\underline{l}, \bar{l}) := \big((\mathscr{A} \oplus S(\underline{l}) \oplus S(\bar{l})\big) \cup \big(\mathscr{A} \ominus S(\underline{l}) \ominus S(\bar{l})\big)\big) \cap [0, L].$$

Note that the time required for determining $S_{ap}(\underline{l}, \bar{l})$ is bounded by $O(mL)$.

Proof. We consider the allocation of two items of not necessarily different qualities, say items 1 and 2, with corresponding allocation intervals $A_1 = [b_1, e_1]$ and $A_2 = [b_2, e_2]$, respectively. We show that optimal patterns exist having allocation points y_1 and y_2 belonging to $S_{ap}(\underline{l}, \bar{l})$. More general cases can be proved inductively. Without loss of generality we can assume that the allocation intervals overlap as in Fig. 6.2. Otherwise, the allocation problem can be separated into two smaller problems which can be dealt with independently. Moreover, to keep the case by case analysis short we only consider cases where additionally

$$\underline{l}_1 + \underline{l}_2 \le e_2 - b_1 \le \bar{l}_1 + \bar{l}_2 \quad \text{and} \quad 2\underline{l}_i > \bar{l}_i,\ e_i - b_i < \bar{l}_i + \underline{l}_i \ \text{ for } i = 1, 2$$

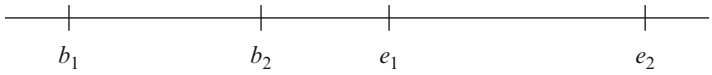

Fig. 6.2 Overlapping allocation intervals

holds. These conditions ensure that both items can be placed but not more than 2. Cases which do no fulfill these conditions can be handled analogously. Since an optimal pattern depends on the profit coefficients γ_1 and γ_2 several cases have to be considered:

Case 1: Let $\gamma_1 = \gamma_2$. As much as possible of the available length should be used, i.e., one of the two solutions

$$y_1 = b_1, \ell_1 = \min\{\bar{l}_1, e_1 - b_1\}, y_2 = b_1 + \ell_1, \ell_2 = \min\{\bar{l}_2, e_2 - y_2\},$$

or

$$\ell_2 = \min\{\bar{l}_2, e_2 - b_2\}, y_2 = e_2 - \ell_2, y_1 = b_1, \ell_1 = \min\{\bar{l}_1, y_2 - b_1\}$$

is optimal.

Case 2: Let $\gamma_1 > \gamma_2$. Then ℓ_1 should be as large as possible.

Subcase 2a: Let $e_1 - b_1 \geq \bar{l}_1$. If $e_2 \geq b_1 + \bar{l}_1 + \underline{l}_2$, then

$$y_1 = b_1, \quad \ell_1 = \bar{l}_1, \quad y_2 = \max\{b_2, b_1 + \bar{l}_1\}, \quad \ell_2 = e_2 - y_2.$$

describes an optimal pattern.
If $e_2 < b_1 + \bar{l}_1 + \underline{l}_2$, an optimal pattern is given by

$$\begin{array}{lll} y_1 = b_1,\ \ell_1 = e_2 - \underline{l}_2 - b_1,\ y_2 = e_2 - \underline{l}_2,\ \ell_2 = \underline{l}_2 & & \text{if } \underline{l}_2\gamma_2 \geq (\bar{l}_1 - \ell_1)\gamma_1, \\ y_1 = b_1,\ \ell_1 = \bar{l}_1, & \text{item 2 is not allocated} & \text{if } \underline{l}_2\gamma_2 < (\bar{l}_1 - \ell_1)\gamma_1. \end{array}$$

In any case we have $y_1, y_2 \in S_{ap}(\underline{l}, \bar{l})$.

Subcase 2b: Let $e_1 - b_1 < \bar{l}_1$. If $e_2 - e_1 \geq \underline{l}_2$ then

$$y_1 = b_1, \quad \ell_1 = e_1 - b_1, \quad y_2 = e_1, \quad \ell_2 = \min\{\bar{l}_2, e_2 - e_1\}$$

provides an optimal pattern.
If $e_2 - e_1 < \underline{l}_2$, the pattern given by

$$\begin{array}{lll} y_1 = b_1,\ \ell_1 = e_2 - \underline{l}_2 - b_1,\ y_2 = e_2 - \underline{l}_2,\ \ell_2 = \underline{l}_2 & & \text{if } \underline{l}_2\gamma_2 \geq (e_1 - b_1 - \ell_1)\gamma_1, \\ y_1 = b_1,\ \ell_1 = e_1 - b_1, & \text{item 2 is not allocated} & \text{if } \underline{l}_2\gamma_2 < (e_1 - b_1 - \ell_1)\gamma_1, \end{array}$$

is optimal as well. Again, we always have $y_1, y_2 \in S_{ap}(\underline{l}, \bar{l})$.

Case 3: Let $\gamma_1 < \gamma_2$. Then ℓ_2 should be as large as possible. The corresponding subcases can be dealt with like in Case 2. □

Similar to the case of only fix-lengths, the packing problem can be separated into some smaller problems if condition (6.16) is violated which can be solved independently from each other. So we assume again that (6.16) is fulfilled.

6.4.4 Packing a Single Piece

In order to compute the optimal value $v(L)$, the allocation of a single piece is considered in the following and will become a basic element in the DP and B&B solution approaches presented below.

Any feasible pattern within an interval $[0, l]$ yields a lower bound for $v(l)$. The current best lower bound for $v(l)$, obtained in a solution process, is denoted as $h(l)$.

The function h has analogous properties as v. Hence, there is a description of h by a finite sequence $(\underline{y}_j)$ of coordinates that at least contains all jump discontinuities and all kinks of h. Any two neighboring points $\underline{y}_j$, $\underline{y}_{j+1}$ define a so-called *basic* (or *reference*) *interval* $B_j := [\underline{y}_j, \underline{y}_{j+1})$. The end point $\underline{y}_{j+1}$ belongs to the basic interval B_{j+1} since h is continuous from the right.

For any interval B_j, the function h can be described by

$$h(y) = \alpha_j + \beta_j(y - \underline{y}_j) \quad \text{for all } y \in B_j.$$

To solve the cutting problem (6.17)–(6.21), a procedure is used which consists of successively placing single pieces. In principle, if piece i with length $\ell_i \in [\underline{l}_i, \bar{l}_i]$ and allocation point $y \in B_j$ is added to the current pattern (which determines h in B_j) then, depending on γ_i and β_j, an improved pattern might be obtained for the interval $[\underline{y}_j + \underline{l}_i, \underline{y}_{j+1} + \bar{l}_i)$. Note that in general, due to the variability of $\ell_i \in [\underline{l}_i, \bar{l}_i]$ and $y \in B_j$, infinitely many patterns exist. In contrast to this, the number of different sequences of items is finite and can be reduced by means of upper bounds and dominance tests. Details can be found in [11].

Because of the variability just mentioned it is not possible to consider single allocation points, rather it is necessary to handle intervals of allocation points. To this end, the basic intervals B_j, defined by the current h-function, can be used.

In the following we provide a construction procedure in which a current solution (pattern) for $[0, \underline{y}_{j+1})$ is extended by placing a single piece i with allocation point in B_j, if possible. The placing of piece i is considered simultaneously for all allocation points $y \in B_j$ and all suitable lengths $\ell_i \in [\underline{l}_i, \bar{l}_i]$. For that, two cases have to be distinguished (see Sects. 6.4.4.1 and 6.4.4.2).

6.4.4.1 Packing a Piece i with $\gamma_i \geq \beta_j$

Let $i \in I_q$ for some $q \in Q$, and let $\gamma_i \geq \beta_j$. In order to find a pattern with value as large as possible, piece i has to be placed with an allocation point in $B_j = [\underline{y}_j, \underline{y}_{j+1})$ as small as possible because of $\gamma_i \geq \beta_j$. Hence, allocation points are both $\underline{y}_j$ (if there is $k \in K_q$ with $\underline{y}_j \in A_k$ and $e_k - \underline{y}_j \geq \underline{l}_i$) and the points b_k for all $k \in K_q$ with $\underline{y}_j < b_k < \underline{y}_{j+1}$. Without loss of generality, let $\xi_1, \ldots, \xi_\kappa$ denote those allocation points with $\underline{y}_j \leq \xi_1 < \cdots < \xi_\kappa < \underline{y}_{j+1} =: \xi_{\kappa+1}$. Let $k(\xi_p) \in K_q$ denote the corresponding allocation interval, i.e., $b_{k(\xi_p)} = \xi_p$, for all $p = 1, \ldots, \kappa$. The length ℓ_i of piece i which should be placed with allocation point ξ_p, is bounded by the remaining length of the corresponding allocation interval and its maximal length, i.e., by

$$\min\{e_{k(\xi_p)} - \xi_p, \bar{l}_i\}, \quad \text{for } p = 1, \ldots, \kappa.$$

For $p = 1, \ldots, \kappa$, the following formula has to be applied to update the function h to possibly get an improved pattern with rightmost piece i:

$$h(y) := \begin{cases} \max\{h(y), h(\xi_p) + \gamma_i(y - \xi_p)\}, & \text{if } y \in [\xi_p + \underline{l}_i, \min\{e_{k(\xi_p)}, \xi_p + \bar{l}_i\}], \\ h(y) & \text{otherwise.} \end{cases} \tag{6.22}$$

Moreover, the placing of piece i with maximum length and variable allocation point has to be considered as well and leads to a further update of h for $p = 1, \ldots, \kappa$:

$$h(y) := \begin{cases} \max\{h(y), h(y - \bar{l}_i) + \gamma_i \bar{l}_i\}, & \text{if } y \in [\xi_p + \bar{l}_i, \min\{e_k, \xi_{p+1} + \bar{l}_i\}], \\ h(y) & \text{otherwise.} \end{cases} \tag{6.23}$$

Formulas (6.22) and (6.23) are based on the following proposition.

Proposition 1. *Let $i \in I_q$ with $\gamma_i \geq \beta_j$. For any pattern $T_i(y, l) := [y, y + l] \subseteq A_k$ with $q = \widetilde{q}(k)$, which is caused by the allocation of piece i with allocation point $y \in B_j$ and length $l \in [\underline{l}_i, \bar{l}_i]$, there is a pattern $T_i(y^*, l^*)$ with $y^* = \xi_p$ for some $p \in \{1, \ldots, \kappa\}$ or $l^* = \bar{l}_i$ so that $T_i(y^*, l^*)$ dominates $T_i(y, l)$ in respect to $\gamma_i l$.*

The proposition is a consequence of Theorem 8.

6.4.4.2 Packing a Piece i with $\gamma_i < \beta_j$

Let $i \in I_q$ with $\gamma_i < \beta_j$. In difference to above, the placement of piece i has to be done with an allocation point as large as possible and length as short as possible. For $p = 1, \ldots, \kappa$, the following formula for updating $h(y)$ has to be applied.

$$h(y) := \begin{cases} \max\{h(y), h(y - \underline{l}_i) + \gamma_i \underline{l}\}, & \text{if } y \in [\xi_p + \underline{l}, \min\{e_{k(\xi_p)}, \xi_{p+1} + \underline{l}_i\}], \\ h(y) & \text{otherwise.} \end{cases} \tag{6.24}$$

Here, $y - \underline{l}_i$ is the related (varying) allocation point. Because of $\gamma_i < \beta_j$, the packing of piece i with a length $\ell_i > \underline{l}_i$ is only necessary for the allocation point $\xi_{\kappa+1} := \underline{y}_{j+1}$ if there exists $k \in K_q$ with $e_k - \underline{y}_{j+1} > \underline{l}_i$. But $\underline{y}_{j+1}$ belongs to the next basic interval, B_{j+1}, and will be considered there since $h(\underline{y}_{j+1}) \geq \lim_{y \uparrow \underline{y}_{j+1}} h(y)$.

Proposition 2. *Let $i \in I_q$ with $\gamma_i < \beta_j$. For any pattern $T_i(y, l) \subseteq A_k$ with $q = \widetilde{q}(k)$, which is caused by the allocation of piece i with allocation point $y \in B_j$ and length $l \in [\underline{l}_i, \bar{l}_i]$, there is a pattern $T_i(y^*, l^*)$ with $y^* = \underline{y}_{j+1}$ or $l^* = \underline{l}_i$ which dominates $T_i(y, l)$ in respect to $\gamma_i l$.*

Formulas (6.22)–(6.24) cause, in general, a change of the basic intervals $B_{j'}$ for $j' \geq j$. If the update of h by these formulas led to a new function h with (partially) increased function values, then the new h need not be monotonously increasing. Therefore, we have to further update this new h by

$$h(y) := \max\{h(y') \mid \underline{y}_j \leq y' \leq y\} \quad \text{for all } y \in [\underline{y}_j, L] \tag{6.25}$$

so that it becomes monotone again.

6.4.5 Solution Approaches

In the subsection we provide two solution approaches for the problem with pieces of variable length where we apply the update rules discussed in Sects. 6.4.4.1 and 6.4.4.2.

6.4.5.1 Branch and Bound Algorithm

The B&B algorithm presented below is based on the LIFO strategy. Appropriate upper bounds, denoted as $\tilde{u}(\cdot)$, are given in [11] where $\tilde{u}(y) \geq v(L) - v(y)$ for all $y \in [0, L]$. Without loss of generality, we assume $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_m$. Branching will be made with respect to

- the basic intervals and
- the pieces which can be allocated next according to a basic interval.

The index μ denotes the branching depth in the algorithm. For any basic interval B_j, the label ν_j denotes whether the basic interval B_j is already investigated for further branching (then $\nu_j = 0$), or if it has still to be considered (then $\nu_j = 1$). Furthermore, $\tilde{B}_\mu = [\tilde{y}_\mu, \hat{y}_\mu)$ denotes the current basic interval. Note, the initial basic interval $\tilde{B}_0 = [0, L]$ is reduced in Step (4) when the first pieces are placed, and the same can happen for other basic intervals during the algorithm. The branching strategy presented here uses the LIFO principle (depth first search), but modifications are obviously possible.

B&B algorithm for 1D cutting problems with variable lengths

(1) *Initialization*
Set $B_0 := [0, L]$, $\alpha_0 := 0$, $\beta_0 := 0$, $\nu_0 = 1$, $\mu := 0$.

(2) *Selection of the next basic interval for branching*
Select the basic interval B_j with largest $\underline{y}_j$-coordinate with $\nu_j = 1$. If such an interval B_j does not exist then go to (5). Otherwise, set $\widetilde{B}_\mu := B_j$ and $\nu_j := 0$. If $\max_{y \in \widetilde{B}_\mu} \{h(y) + \widetilde{u}(y)\} \leq h(L)$, then go to (2). If $L \in \widetilde{B}_\mu$ then go to (4).

(3) *Selection of the next piece for branching*
Select the next piece $i_\mu \in I$ (not already considered for branching) which can be allocated with an allocation point in B_μ. If such a piece does not exist then set $\mu := \mu - 1$ and go to (2).
If $\max_{y \in \widetilde{B}_{\mu-1}} \{h(y) + \gamma_i \underline{l}_i + \widetilde{u}(y + \underline{l}_i)\} \leq h(L)$ then go to (3).

(4) *Allocating a piece*
If $L \in \widetilde{B}_\mu$ then allocate (one after another) all feasible pieces with allocation point $\widetilde{y}_\mu$ with all feasible lengths: a new partition of $\widetilde{B}_\mu = [\widetilde{y}_\mu, L]$ results; set $\nu_j := 1$ for all B_j with $\underline{y}_j \geq \widetilde{y}_\mu$. Otherwise, if $L \notin \widetilde{B}_\mu$ then allocate piece i_μ for all feasible lengths and all allocation points within $\widetilde{B}_\mu$ and update h in accordance with (6.22) – (6.25). If h is increased in some interval $B_{j'}$ during the update process then set $\nu_{j'} := 1$. If h has not been increased anywhere then go to (3), otherwise set $\mu := \mu + 1$ and go to (2).

(5) *Identify an optimal pattern.*

6.4.5.2 FDP Algorithm

For the solution of the considered cutting problem also a FDP method can be used. The general principle of the FDP is similar to the FDP algorithm for the knapsack problem in Sect. 6.2. It computes optimal values $v(y)$ and corresponding patterns for each $y \leq L$, where y is successively increased. Thereby, the values $v(y)$ are obtained by updating the function h in such a way that, starting from a known optimal solution, feasible pieces are placed with all suitable lengths l to possibly get a better solution.

As in the B&B algorithm, intervals of allocation points are considered. In the B&B algorithm, after investigating the basic interval B_j other intervals $B_{j'}$ with $\underline{y}_{j'+1} \leq \underline{y}_j$ have to be considered in general (due to backtracking), and therefore it can happen that B_j (or a subset of it) has to be considered anew if h_j has been (partially) increased. Using the FDP approach, the basic interval B_j is considered exactly once, namely if $h(y) = v(y)$ for all $y \in B_j$ holds. This can also be guaranteed in a B&B algorithm when an appropriate branching strategy (based on breadth first search) is used.

Note that during the allocation of a piece with allocation points in B_j further tests with upper bounds can be used as in the B&B algorithm. If piece i is feasible and $\underline{y}_j + \underline{l}_i < \underline{y}_{j+1}$ holds, then B_j can be split (i.e., $\underline{y}_{j+1}$ can be reduced) in a suitable way. Moreover, during the update process only partitions of $[0, L - \min\{\underline{l}_i \mid i \in I_L\}]$

FDP algorithm for 1D cutting problems with variable lengths

(1) *Initialization*
Set $B_0 := [0, L]$, $\alpha_0 := 0$, $\beta_0 := 0$. Allocate (one after another) all feasible pieces at $\xi = 0$ with all feasible lengths. A first partition of $[0, L]$, i.e., a first finite sequence $(\underline{y}_j)$, is obtained.
If $\underline{y}_1 > \min\{\min\{\underline{l}_i \mid i \in I\}, \min\{b_k \mid b_k > 0, k \in K\}\} =: \widetilde{y}$ then split B_0 into $[0, \widetilde{y})$ and $[\widetilde{y}, \underline{y}_1)$, and renumber. Set $j := 0$.

(2) *Selection of the next basic interval*
Set $j := j + 1$. If $\underline{y}_j + \min\{\underline{l}_i \mid i \in I_L\} > L$ then goto (4).
If $\max_{y \in B_j}\{h(y) + \widetilde{u}(y)\} \leq h(L)$ then go to (2).

(3) *Allocating pieces with allocation points in B_j*:
Allocate (one after another) all feasible pieces i at every allocation point $\xi \in B_j$ and with all feasible lengths according to (6.22) – (6.25).
A new partition of $[0, L]$, i.e., a new finite sequence $(\underline{y}_j)$, is obtained but changes of the $\underline{y}_{j'}$ are only possible for $j' \geq j + 1$. Go to (2).

(4) *Identify an optimal pattern.*

have to be considered since there is no feasible allocation point within the interval $(L - \min\{\underline{l}_i \mid i \in I_L\}, L)$.

In the worst case, $O(m|S_{ap}(\underline{l}, \bar{l})|)$ updates according to (6.22)–(6.25) have to be done in Step (3) of the algorithm. A single update requires at most $O(\bar{l}_i)$ time.

An advantage of the FDP approach can be the relatively constant and well assessable expense to solve an instance (pseudo-polynomiality of the algorithm). In general, this is not the case for the B&B method, where some examples can require much more computation time as in average. However, in general, good (near optimal) solutions are found quickly by a B&B algorithm with LIFO strategy so that a termination after a predefined time span is reasonable for on-line scenarios.

6.5 The 2D Cutting Problem with Quality Demands

In this section we consider 2D cutting problems. Rectangular pieces have to be cut from a larger rectangle of non-homogeneous raw material such that the yield is maximal. Thereby some rectangular parts of the raw material are not allowed to be used for some pieces because of bad quality. We investigate two cases: firstly, so-called defective regions, or simply defects, cannot be used to obtain desired pieces, and secondly, different quality demands are considered. We analyze the case of fixed dimensions of the pieces and give appropriate sets of allocation points, usable in DP and B&B approaches. We also discuss the case when one of the size parameters can vary.

6.5.1 Forbidden Regions

In this subsection we consider the case that some parts of the raw material cannot be used at all to obtain a desired piece. The next subsection is devoted to the discussion of different quality demands.

6.5.1.1 Problem Formulation

Let a rectangle of raw material (wood, metal, glass, etc.) of length L and width W be given. Moreover, the pieces $i \in I := \{1, \dots, m\}$ to be cut are of length ℓ_i, width w_i and have profit coefficient γ_i. Only guillotine cuts are allowed to obtain desired pieces. The part of the raw material used for each piece has to be defect-free. The aim is to maximize to total yield of the cutting pattern. We denote by (x_i, y_i) the *allocation point* of piece i, i.e., if piece i is placed with allocation point (x_i, y_i) then it covers the rectangular region $[x_i, x_i + \ell_i] \times [y_i, y_i + w_i]$. Hence, a 2D pattern can be described by a set of triples (i_t, x_{i_t}, y_{i_t}), $t = 1, \dots, \kappa$, where $i_t \in I$ denotes the t-th placed piece and (x_{i_t}, y_{i_t}) the corresponding allocation point.

Since only guillotine cuts can be applied we can assume, without loss of generality, that all defective parts of the raw material are described by rectangles or a finite union of rectangles. Let D_k, $k \in \overline{K} := \{1, \dots, |\overline{K}|\}$, denote the defective parts with

$$D_k := D_k(a_k, b_k, c_k, d_k) := \{(x, y) \mid a_k \le x \le c_k, b_k \le y \le d_k\} \subset [0, L] \times [0, W],$$

and define

$$D := \bigcup_{k \in \overline{K}} D_k.$$

6.5.1.2 Sets of Allocation Points: No Defects

If $\overline{K} = \emptyset$, the well-known recurrence formula of Gilmore and Gomory [5] can be applied to obtain an optimal pattern with no restriction on the number of stages. Let $u(L', W')$, with $L' \in [0, L]$ and $W' \in [0, W]$, denote the optimal value for rectangle $L' \times W' = [0, L'] \times [0, W']$. Then, as a consequence of Theorem 1 we have

Theorem 9. *Let $T_\ell := S(\ell, L)$ and $T_w := S(w, W)$. Then,*

$$u(L', W') = \overline{u}(p_{S(\ell)}(L'), p_{S(w)}(W')) \quad \text{for all } L' \in [0, L],\ W' \in [0, W], \tag{6.26}$$

where $\overline{u}(L', W')$ is obtained by the following recursion:

$$\overline{u}(L', W') := \max\{\gamma(L', W'),\ g(L', W'),\ h(L', W')\} \quad \textit{for all } L' \in T_\ell,\ W' \in T_w \tag{6.27}$$

with

$$\begin{aligned}
\gamma(L', W') &:= \max\{\gamma_i \mid i \in I,\ \ell_i \le L',\ w_i \le W'\},\\
g(L', W') &:= \max\{\overline{u}(r, W') + \overline{u}(p_{T_\ell}(L' - r), W') \mid 0 < r \le L'/2,\ r \in T_\ell\},\\
h(L', W') &:= \max\{\overline{u}(L', s) + \overline{u}(L', p_{T_w}(W' - s)) \mid 0 < s \le W'/2,\ s \in T_w\}.
\end{aligned}$$

Note that $u(L, W)$ can also be computed using the reduced sets of allocation points $T_\ell := S^{red}(\ell, L)$ and $T_w := S^{red}(w, W)$. In this case, the "=" in (6.26) has to be replaced by "$\ge$" but "=" holds in particular for all $(L', W') \in S^{red}(\ell, L) \times S^{red}(w, W)$.

The time needed for computing $\overline{u}(L', W')$ for all $(L', W') \in T_\ell \times T_w$ according to formula (6.27) is bounded by $O(|T_\ell|\,|T_w|(m + |T_\ell| + |T_w|))$. A reduction to $O(|T_\ell|\,|T_w|(|T_\ell| + |T_w)|)$ can be achieved by an appropriate initialization which avoids the consideration of $\gamma(L', W')$ for each (L', W'), see [12].

6.5.1.3 Sets of Allocation Points: With Defects

If $\overline{K} \neq \emptyset$, the sets of potential allocation points increase since every defective part D_k causes new potential allocation points, e.g., the right end c_k of D_k probably allows the allocation of pieces with x-coordinate c_k, and the left border a_k of D_k can cause that a piece i has x-allocation coordinate $a_k - \ell_i$. This is in difference to Theorem 9 since now regions are not allowed for allocation.

To simplify the description we define an artificial defect with coordinates $a_0 = L$, $b_0 = W$, $c_0 = d_0 = 0$ and set $\overline{K}_0 := \overline{K} \cup \{0\}$. Let $S^*_L(\gamma, \ell)$ and $S^*_W(\gamma, w)$ denote the sets of jump discontinuities in L- and W-direction of the optimal value function $v : [0, L] \times [0, W] \to \mathbb{Z}_+$ for the problem with defects.

Theorem 10. *For any $\ell \in \mathbb{Z}^m_>$, $w \in \mathbb{Z}^m_>$ and $\gamma \in \mathbb{Z}^m_>$, we have*

$$S^*_L(\gamma, \ell) \subseteq S^{ap}_L(\ell) := \bigcup_{k \in \overline{K}_0} (c_k \oplus S(\ell)) \cap [0, L],$$

$$S^*_W(\gamma, w) \subseteq S^{ap}_W(w) := \bigcup_{k \in \overline{K}_0} (d_k \oplus S(w)) \cap [0, W].$$

Moreover, with $T_\ell := S^{ap}_L(\ell)$ and $T_w := S^{ap}_W(w)$,

$$v(L', W') = v(p_{T_\ell}(L'), p_{T_w}(W')) \quad \textit{for all } (L', W') \in [0, L] \times [0, W].$$

holds.

The proof is similar to that of Theorem 3 but now, two dimensions have to be considered. The result of Theorem 10 can be strengthened with respect to the computational complexity by applying the Nicholson principle similar to Sect. 6.3.4. Let

$$S_L^{\leftarrow}(\ell) := \bigcup_{k \in K_0} (a_k \ominus S(\ell)) \cap [0, L], \quad S_W^{\leftarrow}(w) := \bigcup_{k \in K_0} (b_k \ominus S(w)) \cap [0, W]$$

denote the sets of potential allocation points maximal in the sense that, e.g., for any $x \in S_L^{\leftarrow}(\ell)$ there is a combination of piece lengths whose first (left-most) piece, say i, has allocation point x and which is not feasible for allocation points for i larger than x (and similar in W-direction). We define the reduced sets of allocation points by

$$S_L^{red}(\ell) := \{p_{S_L^{ap}(\ell)}(x) \mid x \in S_L^{\leftarrow}(\ell)\} \subseteq S_L^{ap}(\ell),$$
$$S_W^{red}(w) := \{p_{S_W^{ap}(w)}(y) \mid y \in S_W^{\leftarrow}(w)\} \subseteq S_W^{ap}(w).$$

Similar to the 1D case, in general $S_L^{red}(\ell)$ and $S_W^{red}(w)$ are not supersets of $S_L^*(\gamma, \ell)$ and $S_W^*(\gamma, w)$, respectively, but contain sufficiently many points to compute the optimal value $v(L, W)$.

In order to obtain $v(L, W)$ some modifications in comparison with the recursion (6.27) have to be done. The essential difference to the case $\overline{K} = \emptyset$ is that now the yield of a rectangular region $R := [L', L''] \times [W', W'']$ of raw material depends on its position because of the varying quality. That means, the yield function used in a DP recursion is now defined by

$$\gamma(R) := \begin{cases} u(L'' - L', W'' - W'), & \text{if } R \cap \operatorname{int} D = \emptyset, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, a DP recursion to compute v which uses the sets of allocation points $S_L^{red}(\ell)$ and $S_W^{red}(w)$ is given by the following procedure.

For all $R := [L', L''] \times [W', W'']$ with $L', L'' \in S_L^{red}(\ell)$ and $W', W'' \in S_W^{red}(w)$ set

$$\overline{v}(R) := \begin{cases} 0, & \text{if } L'' - L' < \ell_{min} \text{ or } W'' - W' < w_{min}, \\ \max\{\gamma(R), g(R), h(R)\} & \text{otherwise} \end{cases} \tag{6.28}$$

with

$$g(R) := \max\{\overline{v}(L', r, W', W'') + \overline{v}(r, L'', W', W'') \mid r \in S_L^{red}(\ell),\ L' < r < L''\},$$
$$h(R) := \max\{\overline{v}(L', L'', W', s) + \overline{v}(L', L'', s, W'')) \mid s \in S_W^{red}(w),\ W' < s < W''\}.$$

Theorem 11. *Let* $T_\ell := S_L^{red}(\ell)$ *and* $T_w := S_W^{red}(w)$. *Then,*

$$v(L', W') = \overline{v}(0, p_{S_L^{ap}}(L'), 0, p_{S_W^{ap}}) \quad \textit{for all } (L', W') \in T_\ell \times T_w$$

holds, where $\overline{v}$ *is defined by the recursion* (6.28).

The proof is similar to that of Theorem 6 where the L- and W-directions have to be taken into account.

The computation of $\overline{v}(0, L', 0, W')$ for all $(L', W') \in T_\ell \times T_w$ according to (6.28) requires at most $O(|T_\ell|^2|T_w|^2(|K| + |T_\ell| + |T_w|))$ time, since, due to the dependence on the defective regions, $O(|T_\ell|^2|T_w|^2)$ optimal values $\overline{v}(R)$ have to be computed with a DP approach. Obviously, this estimation is rather rough. For instance, if R is defect-free and $u(R)$ is known, $v(R)$ can be determined in constant time.

Since, in general, a large number of $v(R)$-values is needed, the application of a B&B approach becomes more favorable. As upper bound for $v(R)$ we can simply use $u(L'' - L', W'' - W')$ as defined in (6.26) but tighter bounds which regard the existence of defects should be preferred.

The number of small rectangles $R = [L', L''] \times [W', W'']$ used to define subproblems in a B&B algorithm can be further reduced since, e.g., $[L', L''] \cap S_L^{red}(\ell)$ can contain allocation points which are not meaningful for dissecting $[L', L'']$. In principle, appropriate reduced sets of allocation points can be defined for each R similar to those for $L \times W$. Therefore, in order to keep the number of subproblems in a B&B approach small, for each R the reduced set of allocation points should be computed as follows. Let $K_a(R), \ldots, K_d(R)$ denote those defects which are relevant for allocating pieces into the rectangle R:

$$\begin{aligned}
K_a(R) &:= \{k \in K \mid L' + \ell_{min} \le a_k < L'',\ (W', W'') \cap [b_k, d_k] \neq \emptyset\},\\
K_b(R) &:= \{k \in K \mid W' + w_{min} \le b_k < W'',\ (L', L'') \cap [b_k, c_k] \neq \emptyset\},\\
K_c(R) &:= \{k \in K \mid L' < c_k \le L'' - \ell_{min},\ (W', W'') \cap [b_k, d_k] \neq \emptyset\},\\
K_d(R) &:= \{k \in K \mid W' < d_k \le W'' - w_{min},\ (L', L'') \cap [b_k, c_k] \neq \emptyset\}.
\end{aligned}$$

The corresponding sets of allocation points are

$$\begin{aligned}
\tilde{S}_L(R) &:= ((L' \oplus S(\ell)) \cup \{\bigcup (c_k \oplus S(\ell)) \mid k \in K_c(R)\}) \cap [L', L''],\\
\tilde{S}_W(R) &:= ((W' \oplus S(w)) \cup \{\bigcup (d_k \oplus S(w) \mid k \in K_d(R)\}) \cap [W', W''],\\
\tilde{S}_L^{\leftarrow}(R) &:= ((L'' \ominus S(\ell)) \cup \{\bigcup (a_k \ominus S(\ell)) \mid k \in K_a(R)\}) \cap [L', L''],\\
\tilde{S}_W^{\leftarrow}(R) &:= ((W'' \ominus S(w)) \cup \{\bigcup (b_k \ominus S(w)) \mid k \in K_b(R)\}) \cap [W', W''].
\end{aligned}$$

Applying the Nicholson principle, we define reduced sets of allocation points for a single rectangular region R by

$$\tilde{S}_L^{red}(R) := \{p_{\tilde{S}_L(R)}(x) \mid x \in S_L^{\leftarrow}(R)\}, \quad \tilde{S}_W^{red}(R) := \{p_{\tilde{S}_W(R)}(x) \mid x \in S_W^{\leftarrow}(R)\},$$

and we have

$$\tilde{S}_L^{red}(R) \subseteq S_L^{red}(\ell) \cap [L', L''], \quad \tilde{S}_W^{red}(R) \subseteq S_W^{red}(w) \cap [W', W''].$$

By the above construction we obtain

Theorem 12. *Let us consider the 2D cutting problem for the rectangle* $R = [L', L''] \times [W', W'']$ *with defective parts. Then, among all optimal patterns of this problem, there is an optimal pattern having only allocation points with coordinates in* $\tilde{S}_L^{red}(R)$ *and* $\tilde{S}_W^{red}(R)$.

6.5.2 Allocation Areas

Here we investigate the more general case that the raw material consists of areas of different qualities. Obviously, the case with forbidden regions, as discussed in the previous subsection, can be seen as a special case, in which for all items the same parts of the raw material can be used.

6.5.2.1 Problem Formulation

The following 2D cutting problem is considered. Rectangular pieces i of various dimensions $\ell_i \times w_i$, $i \in I$, and different quality demands $q(i) \in Q$ have to be cut from a non-homogeneous raw material of size $L \times W$ in such a way that all allocation conditions (i.e., quality demands) are met and the total value of obtained pieces is maximal. It is allowed that pieces can be cut several times.

As in Sect. 6.4, the set Q denotes the set of all different quality demands. Moreover, let $I_q \subset I$ denote the set of all pieces with quality demand q, i.e., $I_q := \{i \in I \mid q(i) = q\}$. We assume $\cup_{q \in Q} I_q = I$ and $I_q \cap I_p = \emptyset$ for $q \neq p$, $q, p \in Q$.

Parts of the raw material, where a quality demand is fulfilled, are represented by an *allocation area* A_k, $k \in K := \{1, \ldots, |K|\}$. We assume that exactly one quality $q = \widetilde{q}(k) \in Q$ is assigned to each $k \in K$, that the allocation areas are given in the form

$$A_k = \{(x, y) \mid a_k \leq x \leq c_k,\ b_k \leq y \leq d_k\} \subseteq [0, L] \times [0, W],$$

and that, for any $k \in K$, there is $i \in I_{\widetilde{q}(k)}$ with

$$\ell_i \leq c_k - a_k,\ w_i \leq d_k - b_k.$$

We allow that allocation areas can occur several times but then for different qualities, and that they can overlap although if they belong to the same quality demand. But we assume $A_k \not\subseteq A_j$ for all $k, j \in K_q$, $k \neq j$ and all $q \in Q$, where $K_q := \{k \in K \mid q = \widetilde{q}(k)\}$.

For example, if in hardwood cutting a quality demand requires that no black knot is allowed, then the occurrence of a black knot causes up to four partially overlapping allocation areas.

6.5.2.2 Sets of Potential Allocation Points

For the 2D allocation (cutting or packing) problem let $v : [0, L] \times [0, W] \to \mathbb{Z}_+$, denote the optimal value function. This function is non-decreasing. Moreover, it is piecewise constant since only a finite number of different sequences of allocated pieces exists.

Clearly, if $\cup_{k\in K}(a_k, c_k) \neq (0, L)$ or $\cup_{k\in K}(b_k, d_k) \neq (0, W)$, then the problem can be split into subproblems or the size of the raw material can be reduced.

Let $S_L^*(\gamma, \ell)$ and $S_W^*(\gamma, w)$ denote the coordinates of the jump discontinuities in L- and W-direction of the optimal value function v. Our aim is to find supersets of $S_L^*(\gamma, \ell)$ and $S_W^*(\gamma, w)$ which are independent on γ_i, $i \in I$.

Theorem 13. *For any $\ell \in \mathbb{Z}_>^m$, $w \in \mathbb{Z}_>^m$ and $\gamma \in \mathbb{Z}_>^m$, we have*

$$S_L^*(\gamma, \ell) \subseteq S_L^{ap}(\ell) := \bigcup_{k\in K}(a_k \oplus S(\ell)) \cap [0, L],$$

$$S_W^*(\gamma, w) \subseteq S_W^{ap}(w) := \bigcup_{k\in K}(b_k \oplus S(w)) \cap [0, W].$$

Moreover, with $T_\ell := S_L^{ap}(\ell)$ and $T_w := S_W^{ap}(w)$, for all $(L', W') \in [0, L] \times [0, W]$,

$$v(L', W') = v(p_{T_\ell}(L'), p_{T_w}(W'))$$

holds.

The proof is similar to that of Theorem 3. The result of Theorem 13 can be strengthened with respect to the computational complexity by applying the Nicholson principle similar to Sect. 6.3.4. Let

$$S_L^{\leftarrow}(\ell) := \bigcup_{k\in K_0}(c_k \ominus S(\ell)) \cap [0, L], \quad S_W^{\leftarrow}(w) := \bigcup_{k\in K_0}(d_k \ominus S(\ell)) \cap [0, L].$$

denote the set of potential allocation points maximal in the sense that, for any $x \in S_L^{\leftarrow}(\ell)$, there is a combination of piece lengths whose first (left-most) piece, say i, has allocation point x and which is not feasible for allocation points for i larger than x (and similar in W-direction). Applying the Nicholson principle, we define reduced sets of allocation points

$$S_L^{red}(\ell) := \{p_{S_L^{ap}(\ell)}(x) \mid x \in S_L^{\leftarrow}(\ell)\}, \quad S_W^{red}(w) := \{p_{S_W^{ap}(w)}(x) \mid x \in S_W^{\leftarrow}(w)\}.$$

Theorem 14. *Let us consider the 2D cutting problem for the rectangle* $R = [L', L''] \times [W', W'']$ *with quality demands. Then, among all optimal patterns of this problem, there is an optimal pattern having only allocation points in* $S_L^{red}(\ell)$ *and* $S_W^{red}(w)$.

The proof is similar to that of Theorem 6.

The essential difference to the case without quality demands is again the fact that the yield of a rectangular region $R := [L', L''] \times [W', W'']$ of raw material depends on its position because of the varying quality. That means, the yield function used in a DP recursion is defined as follows:

$$\gamma(R) := \begin{cases} 0, & \text{if } I(R) = \emptyset, \\ \max\{\gamma_i \mid i \in I(R)\} & \text{otherwise,} \end{cases}$$

where $I(R) := \{i \in I \mid \exists k \in K_{q(i)} : R \subseteq A_k,\ L'' - L' \geq \ell_i,\ W'' - W' \geq w_i\}$.

Thus, a DP recursion to compute v which uses the sets of allocation points $S_L^{red}(\ell)$ and $S_W^{red}(w)$ is then as follows:

For all $R := [L', L''] \times [W', W'']$ with $L', L'' \in S_L^{red}(\ell)$ and $W', W'' \in S_W^{red}(w)$ set

$$\overline{v}(R) := \begin{cases} 0, & \text{if } L'' - L' < \ell_{min} \text{ or } W'' - W' < w_{min}, \\ \max\{\gamma(R), g(R), h(R)\} & \text{otherwise} \end{cases}$$

with

$$g(R) := \max\{\overline{v}(L', r, W', W'') + \overline{v}(r, L'', W', W'') \mid r \in S_L^{red}(\ell),\ L' < r < L''\},$$
$$h(R) := \max\{\overline{v}(L', L'', W', s) + \overline{v}(L', L'', s, W'')) \mid s \in S_W^{red}(w),\ W' < s < W''\}.$$

The amount of time required to determine all $\overline{v}(R)$-values can be estimated as in Sect. 6.5.1.

Because of the dependence on the allocation areas, now $O(|S_L^{red}(\ell)|^2 \cdot |S_W^{red}(w)|^2)$ optimal values $\overline{v}(R)$ have to be computed with a DP approach. Due to this possibly large number, the usage of a B&B approach becomes more favorable. As upper bound for $v(R)$ one could simply use $u(L'' - L', W'' - W')$ but bounds which regard the allocation areas should be preferred.

Similar to the previous subsection, the number of small rectangles $R = [L', L''] \times [W', W'']$ used to define subproblems in a B&B algorithm can be further reduced. Therefore, for each R, the relevant allocation points should be computed as follows. Let $K_a(R), \ldots, K_d(R)$ denote those allocation areas which are relevant for allocating pieces into a rectangle R:

$$K_a(R) := \{k \in K \mid L' < a_k \leq L'' - \ell_{min},\ (W', W'') \cap [b_k, d_k] \neq \emptyset\},$$
$$K_b(R) := \{k \in K \mid W' < b_k \leq W'' - w_{min},\ (L', L'') \cap [b_k, c_k] \neq \emptyset\},$$
$$K_c(R) := \{k \in K \mid L' + \ell_{min} \leq c_k < L'',\ (W', W'') \cap [a_k, d_k] \neq \emptyset\},$$
$$K_d(R) := \{k \in K \mid W' + w_{min} \leq d_k < W'',\ (L', L'') \cap [a_k, c_k] \neq \emptyset\}.$$

The corresponding sets of allocation points are given by

$$\tilde{S}_L(R) := ((L' \oplus S(\ell)) \cup \{\bigcup(a_k \oplus S(\ell)) \mid k \in K_a(R)\}) \cap [L', L''],$$
$$\tilde{S}_W(R) := ((W' \oplus S(w)) \cup \{\bigcup(b_k \oplus S(w) \mid k \in K_b(R)\}) \cap [W', W''].$$
$$\tilde{S}_L^{\leftarrow}(R) := ((L'' \ominus S(\ell)) \cup \{\bigcup(c_k \ominus S(\ell)) \mid k \in K_c(R)\}) \cap [L', L''],$$
$$\tilde{S}_W^{\leftarrow}(R) := ((W'' \ominus S(w)) \cup \{\bigcup(d_k \ominus S(w)) \mid k \in K_d(R)\}) \cap [W', W''].$$

Applying the Nicholson principle, we obtain the reduced sets of allocation points

$$\tilde{S}_L^{red}(R) := \{p_{\tilde{S}_L(R)}(x) \mid x \in S_L^{\leftarrow}(R)\}, \quad \tilde{S}_W^{red}(R) := \{p_{\tilde{S}_W(R)}(y) \mid y \in S_W^{\leftarrow}(R)\},$$

and we have

$$\tilde{S}_L^{red}(R) \subseteq S_L^{red}(\ell), \quad \tilde{S}_W^{red}(R) \subseteq S_W^{red}(w).$$

Finally, we get

Theorem 15. *We consider the 2D cutting problem for the rectangle* $R = [L', L''] \times [W', W'']$ *with quality demands. Then, among all optimal patterns of this problem, there is an optimal pattern which has only allocation points with coordinates in* $\tilde{S}_L^{red}(R)$ *and* $\tilde{S}_W^{red}(R)$.

Another option to reduce the number of subproblems in a B&B approach consists in replacing $S(\ell)$ and $S(w)$ by those sets which contain only combinations of lengths of items which can be feasibly placed within R.

6.5.3 Generalizations

As a generalization of the problem considered in the previous subsection, one may allow that the pieces have a variable size in one dimension and a fixed size for the other. Here we consider pieces with fixed widths and allow that the length ℓ_i of piece i can take any value in $[\underline{l}_i, \bar{l}_i]$. A related application occurs in hardwood cutting where, in general, the variable lengths are much larger than the fixed widths.

Although, from the theoretical point of view, it would be favorable to use a non-staged guillotine cutting technology, in practical application two-stage guillotine

cutting (and probably, three-stage) is mostly used. We consider here an exact two-stage guillotine cutting (cf. [5]) with horizontal cuts, (i.e., in L-direction) in the first stage. No trimming is allowed. Different quality demands are represented again by allocation areas.

As a naive (basic) solution approach one can compute for each different width $\tilde{w}_j$ and each potential allocation point y the optimal value $v(y, \tilde{w}_j)$ for the part of raw material $[0, L] \times [y, y + \tilde{w}_j]$. In order to limit the computational amount the set of potential allocation points has to be defined appropriately, for instance as $S_W^{red}(w)$. The computation of $v(y, \tilde{w}_j)$ is in fact a 1D cutting problem whose input data are obtained as follows. For a piece $i \in I$ with $w_i = \tilde{w}_j$ and an allocation area $A_k = A_k(a_k, b_k, c_k, d_k)$ with $k \in K_{q(i)}$, the restriction of A_k to the strip $[0, L]x \times [y, y + \tilde{w}_j]$ determines the allocation interval $[a_k, c_k]$ for item i if $[y, y + \tilde{w}_i] \subseteq [b_k, d_k]$ and $\underline{l}_i \leq c_k - a_k$.

Having computed all values $v(y, \tilde{w}_j)$ an optimal combination of the strips can be obtained by solving a 1D cutting problem in W-direction.

Depending on the real cutting technology, the positions of cross (vertical) and rip (horizontal) cuts can be restricted by reduced sets of allocation points in a similar way.

Moreover, practical requirements, such as a least distance between two cut (allocation) positions or a positive kerf, can be regarded within the proposed solution approaches or in the definition of sets of allocation points. If there are restrictions on how often a piece shall be placed, then an appropriate definition of the set $S(\ell)$ should be used, namely

$$\widehat{S}(\ell, u) := \left\{ \sum_{i \in I} \ell_i x_i \mid x_i \leq u_i,\ x_i \in \mathbb{Z}_+,\ i \in I \right\} .$$

6.6 Conclusions

Within this paper we considered one- and two-dimensional cutting and packing problems with additional placement constraints. Such constraints can be caused by defective parts of the raw material or by parts which satisfy different quality demands. We identified appropriate (reduced) sets of potential allocation points that do not depend on the profit coefficients. These sets either cover the set of jump discontinuities of the optimal value function of the allocation problem, or at least contain appropriate allocation points which still allow to compute an optimal pattern.

The proposed sets of potential allocation points strongly depend on the real data. If there are very small-sized pieces or many defects or different quality regions, the cardinality of these sets can be large but not greater than the corresponding size

parameter of the raw material. In case of rather large pieces a high potential to save computational costs arises if the proposed sets of allocation points are used. Moreover, the cardinality of these sets does not change if the unit of measure is changed.

In the one-dimensional case, the explicit computation of a (reduced) set of potential allocation points may look as a meaningless expense, but the basic principle of its definition can be regarded directly within the solution approach as shown in the algorithms for 1D cutting problems with allocation intervals.

In the two-dimensional case, the construction of the (reduced) sets remains, in fact, a one-dimensional task since both dimensions can be handled independently. The use of the proposed sets of potential allocation points can lead to a significant reduction of the number of states, which have to be considered in a DP based approach, and, in a similar way, to a reduction of the number of subproblems which arise during a B&B based solution process. In particular, these reductions are of high importance in cases with complex quality demands.

An appropriate use of the profit coefficients of a problem might be helpful to further reduce the number of allocation points so that the computational effort for obtaining an optimal pattern can be reduced. This topic is left for future research. Moreover, we would like to mention that the definition of good allocation areas might become a non-trivial task if difficult quality demands have to be met, for example one may think of conditions on the number of knots within a certain area of a wooden board.

Acknowledgements This work is supported in a part by the German Research Foundation (DFG) in the Collaborative Research Center 912 "Highly Adaptive Energy-Efficient Computing."

References

1. Astrand, E., Rönnqvist, M.: Crosscut optimization of boards given complete defect information. For. Prod. J. **44**, 15–24 (1994)
2. Beasley, J.E.: An exact two-dimensional non-guillotine cutting tree search procedure. Oper. Res. **33**, 49–65 (1985)
3. Cintra, G.F., Miyazawa, F.K., Wakabayashi, Y., Xavier, E.C.: Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation. Eur. J. Oper. Res. **191**, 61–85 (2008)
4. Dowsland, K.A.: The three-dimensional pallet chart: An analysis of the factors affecting the set of feasible layouts for a class of two-dimensional packing problems. J. Oper. Res. Soc. **35**, 895–905 (1984)
5. Gilmore, P.C., Gomory, R.E.: Multistage cutting stock problems of two and more dimensions. Oper. Res. **13**, 94–120 (1965)
6. Hahn, S.G.: On the optimal cutting of defective sheets. Oper. Res. **16**, 1100–1114 (1968)
7. Herz, J.C.: Recursive computational procedure for two-dimensional stock cutting. IBM J. Res. Dev. **16**, 462–469 (1972)
8. Nicholson, T.A.J.: Finding the shortest route between two points in a network. Comp. J. **9**, 275–280 (1966)

9. Rönnqvist, M.: A method for the cutting stock problem with different qualities. Eur. J. Oper. Res. **83**, 57–68 (1995)
10. Rönnqvist, M., Astrand, E.: Integrated defect detection and optimization for cross cutting of wooden boards. Eur. J. Oper. Res. **108**, 490–508 (1998)
11. Scheithauer, G.: The solution of packing problems with pieces of variable length and additional allocation constraints. Optimization **34**, 81–96 (1995)
12. Scheithauer, G.: Zuschnitt- und Packungsoptimierung. Vieweg + Teubner, Wiesbaden (2008)
13. Scheithauer, G., Terno, J.: Guillotine cutting of defective boards. Optimization **19**, 111–121 (1988)
14. Sweeney, P.E., Haessler, R.W.: One-dimensional cutting stock decisions for rolls with multiple quality grades. Eur. J. Oper. Res. **44**, 224–231 (1990)
15. Sweeney, P.E., Paternoster, E.R.: Cutting and packing problems: A categorized application-oriented research bibliography. J. Oper. Res. Soc. **43**, 691–706 (1992)
16. Terno, J., Lindemann, R., Scheithauer, G.: Zuschnittprobleme und ihre praktische Lösung. Harri Deutsch, Thun and Frankfurt/Main (1987)
17. Wäscher, G., Haußner, H., Schumann, H.: An improved typology of cutting and packing problems. Eur. J. Oper. Res. **183**, 1109–1130 (2007)