

# Decision Procedure for the Existence of Two-Channel Prefix-Free Codes

Hoover H. F. Yin, Ka Hei Ng, Yu Ting Shing, Russell W. F. Lai, and Xishi Wang

## Abstract

The Kraft inequality gives a necessary and sufficient condition for the existence of a single channel prefix-free code. However, the multichannel Kraft inequality does not imply the existence of a multichannel prefix-free code in general. It is natural to ask whether there exists an efficient decision procedure for the existence of multichannel prefix-free codes. In this paper, we tackle the two-channel case of the above problem by relating it to a constrained rectangle packing problem. Although a general rectangle packing problem is NP-complete, the extra imposed constraints allow us to propose an algorithm which can solve the problem efficiently.

## I. INTRODUCTION

*Prefix-free codes*, also known as *prefix codes*, are a class of uniquely decodable codes which has the property that a codeword can be decoded without referring to the symbols of any future codewords. In other words, a codeword of a prefix code can be recognized instantaneously once all the symbols of that codeword are received. There exist prefix codes which are optimal codes, e.g., the Huffman codes [1].

There are different constructions of single channel prefix codes for data compression, e.g., Shannon coding [2], Shannon-Fano coding [3], and Huffman coding [1]. Among these examples, Huffman coding always produce an optimal code [1]. We can construct a Huffman code in linear time when the statistical information of the data to be compressed is sorted [4]. Other than data compression, prefix codes are also used in applications like country calling codes [5] and UTF-8 encoding [6].

It is well-known that the Kraft inequality [7] gives a necessary condition for a single channel source code to be uniquely decodable [8]. This result was generalized into a multichannel case where the channels use a *homogeneous alphabet size*, i.e., all channels use the same alphabet size, in [9]. The case for *heterogeneous alphabet sizes*, i.e., the channels can use different alphabet sizes, was not investigated in [9]. On the other hand, the Kraft inequality gives a sufficient condition for the existence of a single channel prefix code [10]. For the multichannel case, however, the sufficient condition does not hold in general even for homogeneous alphabet size [9]. A natural problem thus arises: Does there exist an efficient decision procedure for the existence of multichannel prefix codes when the channels use heterogeneous alphabet sizes?

The decision procedure aims to close the gap where the multichannel Kraft inequality fails. Precisely, the decision procedure takes a finite multiset of finite codeword lengths as an input, and decides the existence of a prefix code where the multiset of codeword lengths of the prefix code equals to the input multiset exactly.

In this paper, we generalize the multichannel Kraft inequality for channels using heterogeneous alphabet sizes and illustrate the failure of the sufficient condition geometrically. We also present the relation between the existence of a prefix code and the existence of solutions of a constrained rectangle packing problem. We then tackle the two-channel case of the problem of deciding the existence of prefix codes via a reduction to a constrained rectangle packing problem. Although a general rectangle packing problem is NP-complete [11], the constrained version we are interested in can be solved efficiently by our proposed algorithm.

H. Yin and X. Wang are with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong. Email: {yhf015, wx116}@ie.cuhk.edu.hk

K. Ng is with the Department of Physics, The Chinese University of Hong Kong, Hong Kong. Email: kaheicanaan@gmail.com

Y. Shing is with the Mathematics Panel, St. Francis Xavier's College, Hong Kong. Email: js@sfx.edu.hk

R. Lai is with the Chair of Applied Cryptography, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany. Email: russell.lai@cs.fau.de

Part of the work of H. Yin was done when he was with the Institute of Network Coding, The Chinese University of Hong Kong, Hong Kong, which was supported by a grant from the University Grants Committee of the Hong Kong Special Administrative Region (Project No. AoE/E-02/08).

The work of R. Lai was supported by the German research foundation (DFG) through the collaborative research center 1223, and by the state of Bavaria at the Nuremberg Campus of Technology (NCT). NCT is a research cooperation between the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and the Technische Hochschule Nürnberg Georg Simon Ohm (THN).

## II. MULTICHANNEL SOURCE CODES AND MULTICHANNEL PREFIX-FREE CODES

We describe briefly the multichannel source codes and multichannel prefix-free codes proposed in [9] and give some minor generalizations for heterogeneous alphabet sizes.

Let there be  $n$  channels. Denote the alphabet of the information source  $Z$  by  $\mathcal{Z}$ . The alphabet used in the  $i$ -th channel is denoted by  $\mathcal{Z}_i$  where  $i = 1, 2, \dots, n$ . Let  $q_i = |\mathcal{Z}_i|$  be the size of the alphabet.<sup>1</sup> Define  $\mathcal{Z}_i^0 = \{\epsilon\}$  and  $\mathcal{Z}_i^j = \{wv : w \in \mathcal{Z}_i^{j-1}, v \in \mathcal{Z}_i\}$  for  $j \geq 1$ , where  $\epsilon$  is the empty string. In other words,  $\mathcal{Z}_i^j$  is the set of all possible strings of length  $j$  formed by the alphabets in  $\mathcal{Z}_i$ . The collection of all concatenations of alphabets from  $\mathcal{Z}_i$  is the set  $\mathcal{Z}_i^* = \bigcup_{j=0}^{\infty} \mathcal{Z}_i^j$ , i.e.,  $\mathcal{Z}_i^*$  is the set containing all possible finite word sequences formed by the alphabets in  $\mathcal{Z}_i$ , including the empty word.

**Definition 1** (Multichannel Source Codes). An  $n$ -channel *source code*  $\mathcal{Q}$  for the source random variable  $Z$  is a mapping from  $\mathcal{Z}$  to  $\prod_{i=1}^n \mathcal{Z}_i^*$ . Every element in  $\prod_{i=1}^n \mathcal{Z}_i^*$  is called a *word*. For any *source symbol*  $z \in \mathcal{Z}$ ,  $\mathcal{Q}(z)$  is the *codeword* for  $z$ . The image  $\text{Im}(\mathcal{Q}) \subseteq \prod_{i=1}^n \mathcal{Z}_i^*$  of  $\mathcal{Q}$  is called the *codebook*.

**Definition 2** (Multichannel Prefix-Free Codes). Two codewords are *prefix-free* to each other if and only if there exists at least one channel  $i$  such that the  $i$ -th component of the two codewords are prefix-free to each other. An  $n$ -channel *prefix-free code*  $\mathcal{Q}$  is an  $n$ -channel source code where  $\text{Im}(\mathcal{Q}) \subseteq \prod_{i=1}^n \mathcal{Z}_i^*$  such that every pair of codewords in  $\text{Im}(\mathcal{Q})$  are prefix free to each other.

When a codeword is transmitted, the  $i$ -th component of the codeword is transmitted through the  $i$ -th channel. When more than one codewords are transmitted, the codewords are concatenated channel-wise. The boundaries of the codewords are thus not explicit anymore. In order to distinguish the boundaries, we are interested in a class of source codes called the *uniquely decodable code*.

**Definition 3** (Uniquely Decodable Codes). For any two distinct finite sequences of source symbols, if their finite sequences of codewords are different, then the source code is a *uniquely decodable code*.

Similar to the single channel Kraft inequality, the multichannel Kraft inequality gives a necessary condition for a source code to be uniquely decodable. Suppose there are  $m$  codewords in a  $n$ -channel source code  $\mathcal{Q}$ . Let  $\ell_i^j$  be the length of the  $j$ -th codeword in the  $i$ -th channel, where  $j = 1, 2, \dots, m$ . The *codeword length* of the  $j$ -th codeword is defined as a tuple  $(\ell_1^j, \dots, \ell_n^j)$ .

**Theorem 1** (Kraft Inequality). *If  $\mathcal{Q}$  is uniquely decodable, then the lengths of its codewords satisfy*

$$\sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} \leq 1. \quad (1)$$

*Proof:* We can use a similar technique proposed in [12] to prove the generalized Kraft inequality. See Appendix A. ■

When we use a homogeneous alphabet sizes, the multichannel Kraft inequality becomes the version

$$\sum_{j=1}^m q_1^{\sum_{i=1}^n -\ell_i^j} \leq 1$$

proposed in [9]. We can also generalize the entropy bound.

**Theorem 2** (Entropy Bound). *Fix a positive real number  $D$ . Let  $\mathcal{Q}$  be a uniquely decodable code for a source random variable  $Z$  with probability  $\{p_1, p_2, \dots, p_m\}$  and  $D$ -ary entropy  $H_D(Z)$ . Then,*

$$\sum_{j=1}^m p_j \sum_{i=1}^n \ell_i^j \log_D q_i \geq H_D(Z), \quad (2)$$

where the equality holds if and only if  $\sum_{i=1}^n \ell_i^j \log_D q_i = -\log_D p_j$ .

*Proof:* See Appendix A. ■

<sup>1</sup>The work in [9] only considered  $q_1 = q_2 = \dots = q_n$ .

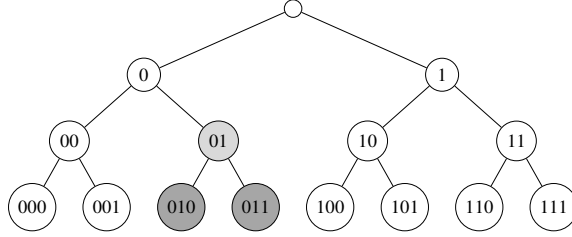


Fig. 1: An example of a binary prefix tree where the maximum codeword length is 3. If 01 is chosen as a codeword, then all its descendants, i.e., 010 and 011 in the figure, cannot be chosen as a codeword anymore.

The L.H.S. of (2) looks a bit different from the traditional entropy bound or the multichannel version in [9]. Here we give an explanation on the meaning of this entropy bound.

We first consider the simple case of single channel source codes, i.e.,  $n = 1$ . Suppose we choose  $D = q_1$ , the left hand side of (2) is the average codeword length. Let  $s_j$  be the sum of the codeword lengths over all the channels of the  $j$ -th codeword, i.e.,  $s_j = \sum_{i=1}^n \ell_i^j$ . The left hand side of (2) is the average of  $s_j$ . That is, we concatenate all the channels of a codeword together and treat it as a codeword in a single channel.

However, when we have heterogeneous alphabet sizes, the codeword length of each channel is in a different unit. The length  $(\ell_1^j, \dots, \ell_n^j)$  means that the codeword in the  $i$ -th channel takes  $\ell_i^j$  symbols from a  $q_i$ -ary alphabet. To unify the measurements, we express  $\ell_i^j$  symbols from a  $q_i$ -ary alphabet by using  $\ell_i^j \log_D q_i$   $D$ -ary symbols. Then, the length of the  $j$ -th codeword after concatenating all the channels is  $\sum_{i=1}^n \ell_i^j \log_D q_i$   $D$ -ary symbols. The average of this length among all the codewords is the L.H.S. of (2).

In practice, we want a codeword of a uniquely decodable code to be decoded without referring to the symbols of any future codewords. A multichannel source code having this property is known as a *strongly self-punctuating code* [9].

**Definition 4** (Strongly Self-Punctuating Codes). A source code  $\mathcal{Q}$  is a *strongly self-punctuating code* if, for any  $n$  sequences  $\mathcal{S}_i \subseteq \mathcal{Z}_i^*$ ,  $i = 1, 2, \dots, n$ , there exists no more than one codeword in  $\mathcal{Q}$  such that the  $i$ -th component of the codeword is a prefix of  $\mathcal{S}_i$  for all  $i = 1, 2, \dots, n$ .

By convention, the multichannel prefix-free codes are also called the multichannel prefix codes. By [9, Thm. 1],<sup>2</sup> we know that a multichannel source code is a strongly self-punctuating code if and only if it is a prefix-free code.

### III. RECTANGLE PACKING AND PREFIX-FREE CODES

A *rectangle packing problem* is a decision procedure for the packability of a given set of two-dimensional parallel<sup>3</sup> rectangular blocks with fixed orientations in a given enclosing two-dimensional container without any overlapping.

In this section, we show the relation between the existence of prefix codes with a given multiset of lengths and the existence of solutions of a special case of the rectangle packing problem.

Let  $(\ell_1^j, \dots, \ell_n^j)$  be the length of the  $j$ -th codeword in a codebook of size  $m$ . Define  $\ell_i^{\max} := \max_{j=1}^m \ell_i^j$  as the maximum length of all codewords in the  $i$ -th channel.

#### A. Single Channel Case

We first consider the single channel case, i.e.,  $n = 1$ . For brevity, we omit the subscripts denoting the 1-st channel, which is the only channel we have in this subsection. A prefix tree can be used to show the prefix relationship between the words in  $\mathcal{Z}^*$ . Suppose the maximum codeword length  $\ell^{\max}$  is given, then the prefix tree is a complete  $q$ -ary tree of height  $\ell^{\max}$ , i.e., the longest root-to-leaf path contains  $\ell^{\max}$  edges. If a node in the tree is chosen as a codeword of a prefix code, then all its descendants cannot be chosen as a codeword anymore. That is, a node is not prefix free to all its descendants. Fig. 1 illustrates an example of a binary prefix tree.

<sup>2</sup>Although [9] only considered homogeneous alphabet size, the proof of Theorem 1 in [9] is independent of alphabet sizes. That is, the same proof is still valid for heterogeneous alphabet sizes.

<sup>3</sup>A set of rectangles are *parallel* if all their edges are either parallel or orthogonal to each other.

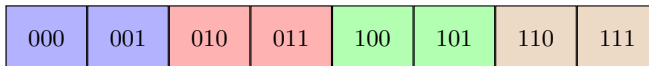


Fig. 2: The container containing all the leaf nodes of the prefix tree shown in Fig. 1. Each colored region corresponds to the leaf nodes of the subtree rooted at either 00, 01, 10 or 11. If a word of length 2 is chosen as a codeword of a prefix-free code, then one of the colors will be selected and that colored region will be occupied. For example, if 01 is chosen, then all the words with the prefix 01 will be eliminated, i.e., red region is occupied.

As selecting a node makes all its descendants unavailable, it is convenient to view the selection as eliminating the selected node and all its descendants. A word of length  $\ell$  will eliminate  $q^{\ell^{\max} - \ell}$  consecutive leaves. If we use a rectangle, or a *block*, to represent the set of consecutive leaves to be eliminated after selecting a node, then the width of the block, i.e., the number of leaves being represented, must be a power of  $q$ .

On the other hand, we can use an enclosing rectangle, or a *container*, to represent the set of all leaves of the prefix tree. As shown in Fig. 1, each possibility of selecting a node corresponding to a word of length  $\ell$  is equivalent to packing the corresponding block of width  $q^{\ell^{\max} - \ell}$  into the container. Fig. 2 illustrates the container which corresponds to the prefix tree shown in Fig. 1, where each colored region corresponds to the leaves of the subtree rooted at either 00, 01, 10 or 11. If a word of length 2 is chosen as a codeword of the prefix code, then a block of width  $2^{3-2} = 2$  will be packed into one of the colored region. For example, if 01 is chosen, then all words with the prefix 01 will be eliminated, i.e., the block is packed in the red region. Observe that we cannot pack the block across two colors at the same time as their prefixes are different, i.e., we have an alignment constraint on the elimination.

The assignment of a codeword depends on the position the block is packed in the container. If two blocks overlap with each other, then the two corresponding codewords are not prefix-free to each other. So, there exists a single channel prefix code with the given codeword lengths if and only if the corresponding blocks can be packed into the container with no overlapping and satisfying the alignment constraint.

If the blocks can be packed into the container, then the sum of the widths of the blocks must be less than or equal to the width of the container, i.e.,

$$\sum_{i=1}^m q^{\ell^{\max} - \ell^i} \leq q^{\ell^{\max}}, \quad (3)$$

which gives the single channel Kraft inequality  $\sum_{i=1}^m q^{-\ell^i} \leq 1$ . On the other hand, suppose we know that the given codeword lengths satisfy the Kraft inequality, then the inequality (3) also holds. Without loss of generality, let  $\ell^1 \leq \ell^2 \leq \dots \leq \ell^m = \ell^{\max}$ . Their corresponding blocks have width  $q^{\ell^{\max} - \ell^1} \geq q^{\ell^{\max} - \ell^2} \geq \dots \geq q^{\ell^{\max} - \ell^m} = 1$ . First, we can put a block of width  $q^{\ell^{\max} - \ell^1}$  to the leftmost position of the container. Then, we can put a block of width  $q^{\ell^{\max} - \ell^2}$  just on the right of the first. We can repeat the procedure to pack in all the remaining blocks. Note that this packing method satisfies the alignment constraint. As there is no gap between the packed blocks, we can conclude that the blocks can be packed into the container. This procedure also gives the codewords of the prefix code if it exists. A similar way to relate a single channel prefix code to the Kraft inequality can be found in [13].

### B. Multichannel Case

Using a similar model, an  $n$ -channel codeword can be considered as an  $n$ -dimensional block. The container which contains all possible words is now  $n$ -dimensional, where the  $i$ -th dimension corresponds to the  $i$ -th channel. To generalize the notion of width, a block or a container occupying  $a_i$  units of space in the  $i$ -th dimension is said to have size  $[a_1, \dots, a_n]$ . We use parentheses and brackets to distinguish the length and size. We sometimes call a block (resp. container) with size  $[a_1, \dots, a_n]$  an “ $a_1 \times a_2 \times \dots \times a_n$  block (resp. container)”. When  $n = 2$ , we call  $a_1$  and  $a_2$  the *width* and *height* respectively.

Given the lengths of the codewords, the sizes of the corresponding blocks and the container can be uniquely determined. A codeword having length  $(\ell_1^j, \dots, \ell_n^j)$  corresponds to a block having size

$$\left[ q_1^{\ell_1^{\max} - \ell_1^j}, \dots, q_n^{\ell_n^{\max} - \ell_n^j} \right].$$

000 000	001 000	010 000	011 000	100 000	101 000	110 000	111 000
000 001	001 001	010 001	011 001	100 001	101 001	110 001	111 001
000 010	001 010	010 010	011 010	100 010	101 010	110 010	111 010
000 011	001 011	010 011	011 011	100 011	101 011	110 011	111 011
000 100	001 100	010 100	011 100	100 100	101 100	110 100	111 100
000 101	001 101	010 101	011 101	100 101	101 101	110 101	111 101
000 110	001 110	010 110	011 110	100 110	101 110	110 110	111 110
000 111	001 111	010 111	011 111	100 111	101 111	110 111	111 111

Fig. 3: An example of a two-dimensional container for a two-channel prefix code. The width and height correspond to the first and second channels respectively, where the maximum codeword lengths of both channels are 3.

These blocks are to be packed into a container of size

$$\left[ q_1^{\ell_1^{\max}}, \dots, q_n^{\ell_n^{\max}} \right].$$

If the  $i$ -th channel is unused by all codewords, then we have  $\ell_i^{\max} = 0$ , i.e., the  $i$ -th dimension of the container has length 1. It is equivalent to consider a lower dimensional container by removing the  $i$ -th channel.

Fig. 3 illustrates an example of a two-dimensional container. If a projection is applied to the width (or the height) of the container in the figure, it becomes the container we have shown in Fig. 2 (or its transpose). That is, the alignment constraint is satisfied dimension-wise.

The following theorem implies that the following are equivalent: 1) the existence of a multichannel prefix code with a given multiset of codeword lengths, and 2) the existence of a solution of a rectangle packing problem with dimension-wise alignment constraints with the corresponding multiset of block sizes.

**Theorem 3.** *Two codewords are prefix free to each other if and only if their corresponding blocks do not overlap.*

*Proof:* By Definition 2, two codewords are prefix free to each others if there exists at least one channel in which the corresponding two components are prefix free to each others. Now, consider a projection of the container to one of the dimension. If the blocks overlap in that dimension, it means that they are not prefix free to each others in that corresponding channel. It is not possible for the blocks to overlap in all dimensions, or otherwise it contradicts that there exists at least one channel they are prefix free to each other. So, the projections of the two blocks do not overlap in at least one of the dimensions, which implies that the two blocks do not overlap in the container.

Conversely, suppose the two blocks do not overlap in the container. Then, we have at least one dimension in which their projections do not overlap. That is, they are prefix free to each other in that corresponding channel. ■

It is trivial to prove that the codeword lengths of any prefix code must satisfy Kraft inequality in the perspective of rectangle packing. To see why, first note that by Theorem 3 it is possible to pack all the blocks of the corresponding sizes inside an appropriate container. It then follows that the sum of volumes of all blocks must be at most the total volume of the container, where the volume of a block (resp. container) is defined as the product of the components in the size of the block (resp. container). That is,

$$\sum_{j=1}^m \prod_{i=1}^n q_i^{\ell_i^{\max} - \ell_i^j} \leq \prod_{i=1}^n q_i^{\ell_i^{\max}},$$

or equivalently

$$\sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} \leq 1,$$

which is exactly the Kraft inequality (1).

The converse is unfortunately not true. That is, the Kraft inequality, which accounts only for volumes but not geometry, is not sufficient to show the existence of a prefix code. We show by giving a counterexample below.

Consider a two-channel binary code. Suppose there are two codewords in the code and the codeword lengths are  $(1, 0)$  and  $(0, 1)$  respectively. The size of the container is  $2 \times 2$ , and the corresponding block sizes are  $1 \times 2$  and  $2 \times 1$ . The area of the container and the sum of the areas of the blocks are both 4, so the multichannel Kraft inequality is satisfied. No matter which of the two possible choice to put a  $1 \times 2$  block, what remains is a  $1 \times 2$  region which is impossible to contain the remaining  $2 \times 1$  block. We thus conclude that prefix codes with the given codeword lengths do not exist.

#### IV. EFFICIENT DECISION PROCEDURE FOR TWO-CHANNEL PREFIX-FREE CODES

In this section, we present an efficient decision procedure for the existence of two-channel prefix codes via a constrained two-dimensional rectangle packing problem.

##### A. Problem Formulation

Our proposed algorithm in Section IV-B will split the empty spaces of a container into multiple smaller containers, so we will formulate a model which consider more than one containers. We first give some definitions which will be used in the remaining text.

**Definition 5** ([Informal Overview] Regions, Blocks, and Containers). A region  $\text{Reg}(x, y, w, h)$  is a rectangular subset of points on the Cartesian plane with width  $w$  and height  $h$ , where the left and bottom borders are closed, and the right and top borders are open. The location  $(x, y)$  of a region is defined as the coordinate of the lower-left corner of the region. The size of the region is denoted by  $[w, h]$ .

We pay special attention to “regular” and “aligned” regions, which are concepts defined with respect to the arities  $q_1$  and  $q_2$ . A region is *regular* if  $x$  is a power of  $q_1$  and  $y$  is a power of  $q_2$ . It is *aligned* if  $x$  is a multiple of  $w$ , and  $y$  is a multiple of  $h$ .

A containers  $C$  is simply a region with a semantic meaning: It is meant to “contain” block(s). A block  $B$  can be considered as a region with a variable location. We can “pack” a block by assigning it a location, denoted by  $B(x, y)$ .

**Definition 6** (Regions). Fix integers  $q_1, q_2 > 1$ . For  $x, y \in \mathbb{Z}$  and  $w, h \in \mathbb{N}$ , the *region*  $R = \text{Reg}(x, y, w, h)$  is defined as  $\text{Reg}(x, y, w, h) := [x, x + w) \times [y, y + h) \subseteq \mathbb{R}^2$ . The tuples  $\text{Loc}(R) := (x, y)$  and  $\text{Size}(R) = [w, h]$  are called the *location* and the *size* of the region respectively.<sup>4</sup> Two regions  $\text{Reg}(x, y, w, h)$  and  $\text{Reg}(x', y', w', h')$  are said to *overlap* with each other if  $\text{Reg}(x, y, w, h) \cap \text{Reg}(x', y', w', h') \neq \emptyset$ . The area of  $R$  is defined as  $\text{Area}(R) := wh$ . The notation of size  $\text{Size}(\cdot)$  is extended naturally to sets, i.e., if  $S$  is a set of regions, then  $\text{Size}(S) := \{\text{Size}(R) : R \in S\}$ .

**Definition 7** (Regularity and Alignment). A size  $[w, h]$  is said to be *regular* (with respect to  $q_1$  and  $q_2$ ) if  $w$  is a power of  $q_1$  and  $h$  is a power of  $q_2$ . A region is said to be regular if its size is regular. A region  $\text{Reg}(x, y, w, h)$  is said to be *aligned* if  $x$  is a multiple of  $w$  and  $y$  is a multiple of  $h$ .

To compare and hence sort two-dimensional sizes in the algorithm to be discussed in Section IV-B, we define a partial ordering which compares both the width and the height equally, and a total ordering which first compares the maximums of the width and the height, then the width, and finally the height. Formally, we give the following definitions.

**Definition 8** (Partial and Total Ordering of Sizes). Let  $s_1 = [w_1, h_1]$  and  $s_2 = [w_2, h_2]$  be two distinct sizes. We define a partial ordering  $\succ$  and a total ordering  $>$  of sizes. We write  $s_1 \succ s_2$  if  $w_1 \geq w_2$  and  $h_1 \geq h_2$ . We write  $s_1 > s_2$  if one of the following is satisfied:

<sup>4</sup>We only need to consider nonnegative integers  $x, y$  and positive integers  $w, h$  in this paper.

- $\max\{w_1, h_1\} > \max\{w_2, h_2\}$
- $\max\{w_1, h_1\} = \max\{w_2, h_2\}$  and  $w_1 > w_2$
- $\max\{w_1, h_1\} = \max\{w_2, h_2\}$  and  $w_1 = w_2$  and  $h_1 > h_2$

Note that if  $s_1 \succ s_2$  then  $s_1 > s_2$ .

*Example 1.* The sizes  $[8, 8]$ ,  $[8, 4]$ ,  $[8, 2]$ ,  $[4, 8]$ ,  $[2, 8]$ ,  $[4, 2]$  are sorted in descending order.

In the following, we state a basic geometric fact about regular aligned regions.

**Lemma 1.** *If  $R_1$  and  $R_2$  are two regular aligned regions with  $R_2 \succeq R_1$ , then either  $R_2$  completely covers  $R_1$ , or they do not overlap. More precisely, if  $R_1 = \text{Reg}(x_1, y_1, w_1, h_1)$  and  $R_2 = \text{Reg}(x_2, y_2, w_2, h_2)$  are regular and aligned,  $w_2 \geq w_1$ , and  $h_2 \geq h_1$ , then one of the following must be true:*

- $R_1 \cap R_2 = R_1$
- $R_1 \cap R_2 = \emptyset$

*Proof:* The lemma is quite obvious. For the sake of completeness, we provide a proof for the case where the lower-left corner of  $R_2$  overlaps with the upper-right corner of  $R_1$ .

Since  $R_1$  and  $R_2$  are regular and aligned, for  $i \in \{1, 2\}$ , let  $w_i = q_1^{a_i}$  and  $x_i = b_i w_i = b_i q_1^{a_i}$ , for some non-negative integers  $a_i$  and integers  $b_i$ . Since  $w_2 \geq w_1$ , we have  $a_2 \geq a_1$ .

Suppose  $R_1 \cap R_2 \neq \emptyset$ . Consider the case where the lower-left corner of  $R_2$  overlaps with the upper-right corner of  $R_1$ . In particular, we have:

$$\begin{aligned} x_1 &\leq x_2 < x_1 + w_1 \leq x_2 + w_2 \\ b_1 q_1^{a_1} &\leq b_2 q_1^{a_2} < (b_1 + 1) q_1^{a_1} \\ b_1 &\leq b_2 q_1^{a_2 - a_1} < b_1 + 1 \end{aligned}$$

Since  $b_2 q_1^{a_2 - a_1}$  is an integer, the third inequality forces  $b_2 q_1^{a_2 - a_1} = b_1$ , which implies that  $x_1 = b_1 q_1^{a_1} = b_2 q_1^{a_2} = x_2$ . Using a similar argument, we can show that  $y_1 = y_2$ . Thus  $R_1 \cap R_2 = R_1$ .

The other three cases can be proven analogously. ■

The following lemma is a single dimension version of Lemma 1.

**Lemma 2.** *Let  $X_1 = [x_1, x_1 + w_1)$ ,  $X_2 = [x_2, x_2 + w_2)$ ,  $Y_1 = [y_1, y_1 + h_1)$  and  $Y_2 = [y_2, y_2 + h_2)$  be intervals where  $w_1 \mid x_1$ ,  $w_2 \mid x_2$ ,  $h_1 \mid y_1$ ,  $h_2 \mid y_2$ ,  $w_1$  and  $w_2$  are powers of  $q_1$ ,  $h_1$  and  $h_2$  are powers of  $q_2$ , and  $w_1 \leq w_2$ ,  $h_1 \leq h_2$ . Then, we have*

- $X_1 \cap X_2 = X_1$  or  $X_1 \cap X_2 = \emptyset$ ; and
- $Y_1 \cap Y_2 = Y_1$  or  $Y_1 \cap Y_2 = \emptyset$ .

*Proof:* We can treat the intervals  $X_1, X_2$  be a projection of regions. Let  $R_1 = \text{Reg}(x_1, 0, w_1, 1)$  and  $R_2 = \text{Reg}(x_2, 0, w_2, q_1)$ . As we have  $q_1 > 1$ , thus  $R_1 \preceq R_2$ . Then, we can apply Lemma 1 to show that either  $R_1 \cap R_2 = R_1$  or  $R_1 \cap R_2 = \emptyset$ , which corresponds to either  $X_1 \cap X_2 = X_1$  or  $X_1 \cap X_2 = \emptyset$ . A similar argument can be used to prove the case for  $Y_1$  and  $Y_2$ . ■

**Definition 9** (Blocks and Containers). Let  $[w, h]$  be a regular size. A *block*  $B_{w,h}$  is a function mapping locations to regions, i.e.,  $B_{w,h} : (u, v) \in \mathbb{Z}^2 \mapsto \text{Reg}(u, v, w, h)$ . The tuple  $\text{Size}(B_{w,h}) := [w, h]$  is called the size of the block  $B_{w,h}$ . We omit the subscript in  $B_{w,h}$  when the size is clear from the context. The value  $\text{Area}(B) := wh$  is called the area of  $B$ .

A *container*  $C$  is simply a region  $\text{Reg}(x, y, w, h)$ , for some  $x, y \in \mathbb{Z}$  and  $w, h \in \mathbb{N}$ , with semantic meaning. The definitions of the location, the size, and the area of a container is inherited from those of a region.

Note that the sizes of the blocks transformed from the corresponding codeword lengths are always regular.

Consider a multiset<sup>5</sup>  $\mathcal{B} = \{B_1, \dots\}$  of regular blocks, and a set  $\mathcal{C} = \{C_1, \dots\}$  of non-overlapping containers. Let  $C_i = \text{Reg}(x_C^i, y_C^i, w_C^i, h_C^i)$  be the  $i$ -th container, and  $[w_B^i, h_B^i] = \text{Size}(B_i)$  be the size of the  $i$ -th block.

<sup>5</sup>A multiset [14] is a set with possibly repeated elements.

When we say some blocks or containers are sorted, it means that it is sorted by their sizes in descending order according to Definition 8. Without loss of generality, we assume  $\mathcal{B}$  is sorted in **descending order**.

**Definition 10** (Constrained Rectangle Packing Problem). Let  $\mathcal{B} = \{B_i\}_{i=1}^{|\mathcal{B}|}$  be a multiset of blocks and  $\mathcal{C}$  be a set of non-overlapping containers. The two-dimensional rectangle packing problem specified by  $(\mathcal{B}, \mathcal{C})$ , is to find a simultaneous assignment  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  to all blocks in  $\mathcal{B}$ , such that the resulting regions  $B(x_i, y_i)$  are aligned and non-overlapping, and each of them is contained in a container in  $\mathcal{C}$ , i.e.,

$$\begin{cases} \forall B_i \in \mathcal{B}, B_i(x_i, y_i) \text{ is aligned, i.e., } w_B^i | x_i \text{ and } h_B^i | y_i \\ \forall B_i, B_j \in \mathcal{B} \text{ s.t. } i \neq j, B_i(x_i, y_i) \cap B_j(x_j, y_j) = \emptyset \\ \forall B_i \in \mathcal{B}, \exists C \in \mathcal{C} \text{ s.t. } B_i(x_i, y_i) \subseteq C \end{cases} .$$

A simultaneous assignment  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  satisfying the above is called a solution. If no such assignment exists, then we say that (the problem specified by)  $(\mathcal{B}, \mathcal{C})$  has no solution. Note that if a solution exists, then each block  $B_i$  must be contained in one and only one container, since the containers are non-overlapping.

At the beginning when there is only one initial container, we have the problem  $(\mathcal{B}, \mathcal{C}_0)$  where  $\mathcal{C}_0 = \{C_0\}$ ,  $C_0 = \text{Reg}(0, 0, q_1^{\max}, q_2^{\max})$ .

### B. The Algorithm

Before stating our algorithm for the problem  $(\mathcal{B}, \mathcal{C})$ , we introduce a sub-algorithm which aims to cut containers into smaller containers satisfying some constraints. The cutting algorithm for the function defined as follows.

**Definition 11** (Container Cutting Function  $\sigma$ ). Let  $C$  be a container and  $s$  be a regular size. We model the cutting of containers by defining a function  $\sigma(C, s)$  which maps the container  $C$  and the size  $s$  into the smallest set of non-overlapped regular aligned containers each with size  $\preceq s$ . Also, the resulting containers must satisfy  $\bigcup_{C' \in \sigma(C, s)} C' = C$ . If no such set exists, then  $\sigma(C, s) := \emptyset$ , the empty set. The notation extends naturally to sets of non-overlapping containers. Let  $\mathcal{C}$  be a set of containers. We define  $\sigma(\mathcal{C}, s) := \bigcup_{C \in \mathcal{C}} \sigma(C, s)$ .

Note that the cutting function  $\sigma$  might not be well defined since there may exist more than one smallest sets of such regions. To show that the cutting function is well defined (Lemma 4), we introduce the following notation.

**Definition 12** (Quotient Containers and Remainder Regions). Let  $C = \text{Reg}(x_C, y_C, w_C, h_C)$  be a container, and  $[w, h]$  be a regular size. The set of *quotient containers* with respect to  $[w, h]$  contained in  $C$  is the set of all regular aligned containers of size  $[w, h]$  overlapping with  $C$ . Formally, it is defined as

$$\mathcal{Q} = \text{QuoCon}(C, [w, h]) = \{\text{Reg}(x, y, w, h) \subseteq C : w | x, h | y\}.$$

In the typical case where  $\mathcal{Q} \neq \emptyset$ , let  $x_{\mathcal{Q}}, y_{\mathcal{Q}}, w_{\mathcal{Q}}, h_{\mathcal{Q}}$  be such that  $\text{Reg}(x_{\mathcal{Q}}, y_{\mathcal{Q}}, w_{\mathcal{Q}}, h_{\mathcal{Q}}) = \bigcup_{C' \in \mathcal{Q}} C'$ . Consider the space obtained by removing the region  $\text{Reg}(x_{\mathcal{Q}}, y_{\mathcal{Q}}, w_{\mathcal{Q}}, h_{\mathcal{Q}})$  from  $C$ . Such a space can be divided into a set of (at most) 8 *remainder regions*, which is defined as

$$\begin{aligned} & \text{RemReg}(C, [w, h]) \\ &= \left\{ \begin{array}{lll} \text{Reg}(x_{\text{left}}, y_{\text{high}}, w_{\text{left}}, h_{\text{high}}), & \text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}}), & \text{Reg}(x_{\text{right}}, y_{\text{high}}, w_{\text{right}}, h_{\text{high}}), \\ \text{Reg}(x_{\text{left}}, y_{\text{mid}}, w_{\text{left}}, h_{\text{mid}}), & & \text{Reg}(x_{\text{right}}, y_{\text{mid}}, w_{\text{right}}, h_{\text{mid}}), \\ \text{Reg}(x_{\text{left}}, y_{\text{low}}, w_{\text{left}}, h_{\text{low}}), & \text{Reg}(x_{\text{mid}}, y_{\text{low}}, w_{\text{mid}}, h_{\text{low}}), & \text{Reg}(x_{\text{right}}, y_{\text{low}}, w_{\text{right}}, h_{\text{low}}) \end{array} \right\} \end{aligned}$$

where  $x_{\text{left}} = x_C$ ,  $x_{\text{mid}} = x_{\mathcal{Q}}$ ,  $x_{\text{right}} = x_{\mathcal{Q}} + w_{\mathcal{Q}}$ ,  $y_{\text{low}} = y_C$ ,  $y_{\text{mid}} = y_{\mathcal{Q}}$ ,  $y_{\text{high}} = y_{\mathcal{Q}} + h_{\mathcal{Q}}$ ,  $w_{\text{left}} = x_{\mathcal{Q}} - x_C$ ,  $w_{\text{mid}} = w_{\mathcal{Q}}$ ,  $w_{\text{right}} = (x_C + w_C) - (x_{\mathcal{Q}} + w_{\mathcal{Q}})$ ,  $h_{\text{low}} = y_{\mathcal{Q}} - y_C$ ,  $h_{\text{mid}} = h_{\mathcal{Q}}$ , and  $h_{\text{high}} = (y_C + h_C) - (y_{\mathcal{Q}} + h_{\mathcal{Q}})$ .

In the event that  $\mathcal{Q} = \emptyset$ , then we define  $\text{RemReg}(C, [w, h]) = C$ . Note that for any size  $[w, h]$  it holds that  $C = \text{QuoCon}(C, [w, h]) \cup \text{RemReg}(C, [w, h])$ .

Note that there are other ways to divide the space obtained by removing the quotient containers from  $C$  into regions. We define the remainder regions with the intuition that no regular aligned regions of size at most  $[w, h]$  can be found across the boundaries of two remainder regions. Therefore it does not really matter how the remainder regions are defined. Formally, we prove the following lemma.



**Lemma 3.** Let  $C$  be a container,  $[w, h]$  and  $[w', h']$  be regular sizes with  $[w, h] \succeq [w', h']$ . Let  $\mathcal{R} = \text{RemReg}(C, [w, h])$ . Let  $\mathcal{R}' \subseteq \mathcal{R}$  be any subset of remainder containers covering a rectangular space, i.e., one can define  $x_R, y_R, w_R, h_R$  such that  $R = \text{Reg}(x_R, y_R, w_R, h_R) = \bigcup_{R' \in \mathcal{R}'} R'$ . Then

$$\text{QuoCon}(R, [w', h']) = \bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h']).$$

*Proof:* Note that  $|\mathcal{R}'| \leq 3$  and the proof is trivial for  $|\mathcal{R}'| < 2$ . Also, it is obvious that  $\text{QuoCon}(R, [w', h'])$  and  $\bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h'])$  are empty sets when  $[w, h] = [w', h']$ . If we have  $|\mathcal{R}'| \geq 2$ , then one of the corner in  $\mathcal{R}$  must be in  $\mathcal{R}'$ .

We first consider  $|\mathcal{R}'| = 2$ . Without loss of generality, let  $\mathcal{R}'$  be the set containing the top-left and top-middle remainder regions, i.e.,

$$\mathcal{R}' = \{\text{Reg}(x_{\text{left}}, y_{\text{high}}, w_{\text{left}}, h_{\text{high}}), \text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}})\}.$$

The other configurations can be proved by similar arguments.

It is clear that we have  $\text{QuoCon}(R, [w', h']) \supseteq \bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h'])$ .

Case I:  $\text{QuoCon}(R, [w', h']) = \emptyset$ . The proof is done immediately.

Case II:  $\text{QuoCon}(R, [w', h']) \neq \emptyset = \bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h'])$ . Let

$$Q = \bigcup_{Q' \in \text{QuoCon}(R, [w', h'])} Q' = \text{Reg}(x_Q, y_Q, w_Q, h_Q),$$

where  $w' \mid x_Q$ . We must have  $w_Q < 2w'$ , or otherwise  $\bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h']) \neq \emptyset$ . That is, we have  $w_Q = w'$ . Note that  $Q \cap \text{Reg}(x_{\text{left}}, y_{\text{high}}, w_{\text{left}}, h_{\text{high}}) \neq \emptyset$  and  $Q \cap \text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}}) \neq \emptyset$ , or otherwise it leads to the same contradiction that  $\bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h']) \neq \emptyset$ . Let  $Q \cap \text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}}) = \text{Reg}(x_B, y_B, w_B, h_B)$ , where  $w \mid x_B > x_Q$ . As  $[w, h] \succeq [w', h']$  are both regular, we also have  $w' \mid x_B$ . On the other hand, we have  $x_Q + w' > x_B > x_Q$  but  $w' \mid x_Q, x_B$ , which is a contradiction. That is, case II will not occur.

Case III:  $\text{QuoCon}(R', [w', h']) \neq \emptyset$  for all  $R' \in \mathcal{R}'$ . Let the remainder regions of  $\text{Reg}(x_{\text{left}}, y_{\text{high}}, w_{\text{left}}, h_{\text{high}})$  and  $\text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}})$  be the sets

$$\left\{ \begin{array}{ccc} \text{Reg}_{1A}, & \text{Reg}_{2A}, & \text{Reg}_{3A}, \\ \text{Reg}_{4A}, & & \text{Reg}_{6A}, \\ \text{Reg}_{7A}, & \text{Reg}_{8A}, & \text{Reg}_{9A} \end{array} \right\} \quad \text{and} \quad \left\{ \begin{array}{ccc} \text{Reg}_{1B}, & \text{Reg}_{2B}, & \text{Reg}_{3B}, \\ \text{Reg}_{4B}, & & \text{Reg}_{6B}, \\ \text{Reg}_{7B}, & \text{Reg}_{8B}, & \text{Reg}_{9B} \end{array} \right\}$$

respectively in a geometric illustration. By the same means of illustration, we have

$$\bigcup_{R' \in \mathcal{R}'} \text{RemReg}(R', [w', h']) = \left\{ \begin{array}{cccccc} \text{Reg}_{1A}, & \text{Reg}_{2A}, & \text{Reg}_{3A}, & \text{Reg}_{1B}, & \text{Reg}_{2B}, & \text{Reg}_{3B}, \\ \text{Reg}_{4A}, & & \text{Reg}_{6A}, & \text{Reg}_{4B}, & & \text{Reg}_{6B}, \\ \text{Reg}_{7A}, & \text{Reg}_{8A}, & \text{Reg}_{9A}, & \text{Reg}_{7B}, & \text{Reg}_{8B}, & \text{Reg}_{9B} \end{array} \right\}$$

and

$$\text{RemReg}(R, [w', h']) = \left\{ \begin{array}{ccc} \text{Reg}_{1A}, & \text{Reg}_{2A} \cup \text{Reg}_{3A} \cup \text{Reg}_{1B} \cup \text{Reg}_{2B}, & \text{Reg}_{3B}, \\ \text{Reg}_{4A}, & & \text{Reg}_{6B}, \\ \text{Reg}_{7A}, & \text{Reg}_{8A} \cup \text{Reg}_{9A} \cup \text{Reg}_{7B} \cup \text{Reg}_{8B}, & \text{Reg}_{9B} \end{array} \right\}.$$

Due to the fact that the height of  $\text{Reg}_i$  is less than  $h'$  for  $i = 2A, 3A, 1B, 2B, 8A, 9A, 7B, 8B$ , we know that there is no subset of  $\text{Reg}_{2A} \cup \text{Reg}_{3A} \cup \text{Reg}_{1B} \cup \text{Reg}_{2B}$  or  $\text{Reg}_{8A} \cup \text{Reg}_{9A} \cup \text{Reg}_{7B} \cup \text{Reg}_{8B}$  is in  $\text{QuoCon}(R, [w', h'])$ . Similarly, we do not need to consider  $\text{Reg}_i$  where  $i = 1A, 4A, 7A, 3B, 6B, 9B$  as their width is less than  $w'$ . That is, we only need to show that  $\text{Reg}_{6A} \cup \text{Reg}_{4B} = \emptyset$ .

For  $i = 6A, 4B$ , we write  $\text{Reg}_i = \text{Reg}(x_i, y_i, w_i, h_i)$ . Note that we have  $w' \mid x_{6A}, w \mid x_{4B}, w' \mid x_{4B} + w_{4B}$ , and  $x_{6A} + w_{6A} = x_{4B}$ ,  $0 \leq w_{6A}, w_{4B} < w'$ . So, we also have  $0 \leq w_{6A} + w_{4B} < 2w'$  and  $w' \mid w_{6A} + w_{4B}$ . This implies that  $w_{6A} + w_{4B}$  is either 0 or  $w'$ . Suppose  $w_{6A} + w_{4B} = w'$ , then we have  $0 < w_{6A}, w_{4B} < w'$ . As  $[w, h] \succeq [w', h']$  are regular, we have  $w' \mid x_{4B}$ . That is, we have  $x_{6A} + w' > x_{4B} > x_{6A}$  and  $w' \mid x_{6A}, x_{4B}$ , which is a contradiction. So, we must have  $w_{6A} + w_{4B} = 0$ . In other words, we have  $\text{Reg}_{6A} \cup \text{Reg}_{4B} = \emptyset$ .

Case IV: there is one and only one  $R' \in \mathcal{R}'$  such that  $\text{QuoCon}(R', [w', h']) = \emptyset$ . Without loss of generality, let  $\text{QuoCon}(\text{Reg}(x_{\text{mid}}, y_{\text{high}}, w_{\text{mid}}, h_{\text{high}}), [w', h']) \neq \emptyset$ . By using a similar argument in case III, we do not need to consider the upper, lower and the rightmost remainder regions. For the leftmost remainder region, first, we have

$w \mid x_{\text{mid}}$ . As  $[w, h] \succeq [w', h']$  are regular, we also have  $w' \mid x_{\text{mid}}$ . So, we must have  $w_{\text{left}} < w'$  or otherwise  $\text{QuoCon}(R', [w', h']) \neq \emptyset$  for both  $R' \in \mathcal{R}'$ . Then, we have  $x_{\text{left}} = x_{\text{mid}} - w_{\text{left}}$  thus  $w' \nmid x_{\text{left}}$ , which means that  $\text{QuoCon}(R, [w', h']) = \bigcup_{R' \in \mathcal{R}'} \text{QuoCon}(R', [w', h'])$ .

The above four cases finish the proof for  $|\mathcal{R}'| = 2$ . For  $|\mathcal{R}'| = 3$ , it can be done by simply applying twice the arguments for  $|\mathcal{R}'| = 2$ .  $\blacksquare$

With the above notions, we can now prove the uniqueness of the output of the cutting function.

**Lemma 4.** *Let  $\mathcal{C}$  be a set of non-overlapping containers and  $[w, h]$  be a regular size. The set given by  $\sigma(\mathcal{C}, [w, h])$  is unique, i.e.,  $\sigma$  is a function.*

*Proof:* It suffices to show that  $\sigma(C, [w, h])$  is unique for each  $C \in \mathcal{C}$ . Fix  $C \in \mathcal{C}$  and let  $C = \text{Reg}(x_C, y_C, w_C, h_C)$ . Let  $\mathcal{S}$  be a smallest set of non-overlapping regular aligned containers each with width and height at most  $w$  and  $h$  respectively, and  $C = \bigcup_{S \in \mathcal{S}} S$ . We will show that such a set  $\mathcal{S}$  is unique, and thus is the value of  $\sigma(C, [w, h])$ . Our argument is constructive. In particular, we will recursively determine disjoint subsets of  $\mathcal{S}$ , until the entire  $\mathcal{S}$  is determined. This can be turned into an algorithm which computes  $\sigma(C, [w, h])$ .

Let  $[w', h']$  be a regular size with the largest width and height respectively such that  $[w', h'] \preceq [w, h]$  and  $\mathcal{Q} = \text{QuoCon}(C, [w', h']) \neq \emptyset$ . That is,  $[w', h']$  is the largest size (in total order) of regular aligned regions we can cut from  $C$ . We claim that  $\mathcal{Q} \subseteq \mathcal{S}$ . Suppose not, then there exists a regular aligned container  $C^* \in \mathcal{Q}$  such that  $C^* \notin \mathcal{S}$ . By the definition of  $[w', h']$ , we know that every container in  $\mathcal{S}$  has size  $\preceq [w', h']$ . By the definition of quotient containers,  $C^* \subseteq C$ . Since  $C = \bigcup_{S \in \mathcal{S}} S$ , we have  $C^* \subseteq \bigcup_{S \in \mathcal{S}} S$ . So, there is a smallest subset  $\mathcal{S}' \subseteq \mathcal{S}$  such that  $C^* \subseteq \bigcup_{S \in \mathcal{S}'} S$ . Note that  $C^*$  and all  $S \in \mathcal{S}'$  are regular aligned containers. For any  $S \in \mathcal{S}'$ , as we have  $\text{Size}(S) \preceq [w', h'] = \text{Size}(C^*)$ , we can apply Lemma 1 to show that either  $S \cap C^* = S$  or  $S \cap C^* = \emptyset$ . The latter cannot be true as it contradicts the minimality of  $\mathcal{S}'$ . That is, we have  $C^* = \bigcup_{S \in \mathcal{S}'} S \notin \mathcal{S}$ , which contradicts the minimality of  $\mathcal{S}$ . So, we must have  $C^* \in \mathcal{S}$ .

By the above, we have concluded that  $\mathcal{Q} \subseteq \mathcal{S}$ . Now, consider the remainder regions  $\mathcal{R} = \text{RemReg}(C, [w', h'])$ . Note that by Lemma 3,  $\text{QuoCon}$  acts locally on the remainder regions. We can thus apply the argument above recursively by substituting  $C$  with  $R$  for each  $R \in \mathcal{R}$ , and eventually uniquely determine  $\mathcal{S}$ .  $\blacksquare$

Building on the cutting algorithm, we propose a naïve greedy algorithm (Algorithm 1) which solves  $(\mathcal{B}, \mathcal{C})$ . The algorithm cuts the containers at hand minimally into smaller regular aligned containers, such that one of the resulting containers is just enough to contain the largest unpacked block at hand. The algorithm naturally packs the largest block at hand to such a container, releases the unused space back to the set of containers at hand, and continues until all blocks are packed.

The naïve algorithm (Algorithm 1) is very inefficient as it needs to compute at each step the container cutting function  $\sigma$  entirely although only a small part of its output is used. Later, we will show an optimized algorithm which has input-output behavior identical to that of Algorithm 1 but is much more efficient by performing “lazy” computation of  $\sigma$ .

**Theorem 4.** *Algorithm 1 outputs a solution to  $(\mathcal{B}, \mathcal{C})$  if and only if  $(\mathcal{B}, \mathcal{C})$  has a solution. Furthermore, Algorithm 1 is a decision procedure for  $(\mathcal{B}, \mathcal{C})$  if we modify it a little bit that*

- i) output 0 if Algorithm 1 outputs  $\perp$ ; and
- ii) output 1 if Algorithm 1 does not output  $\perp$ .

The problem of determining the existence of a solution to  $(\mathcal{B}, \mathcal{C})$  is a yes-no question. It is obvious that the modifications for the decision procedure will not affect the correctness of the algorithm.

We devote the next two subsections to prove the above theorem.

### C. Properties of the Solutions

In order to prove Theorem 4, we need to establish some properties regarding the solutions of  $(\mathcal{B}, \mathcal{C})$ .

**Lemma 5.** *Let  $\mathcal{B} = \{B_1, \dots, B_m\}$  be a multiset of blocks with sizes sorted in descending order, and  $\mathcal{C}$  be a set of non-overlapping containers. Let  $w_{\text{max}} = \max_{i \in [m]} w_B^i$  and  $h_{\text{max}} = \max_{i \in [m]} h_B^i$  be the width and height of the widest and tallest blocks respectively. It holds that  $(\mathcal{B}, \mathcal{C})$  has a solution if and only if  $(\mathcal{B}, \sigma(\mathcal{C}, [w_{\text{max}}, h_{\text{max}}]))$  has a solution. Furthermore, any solution to  $(\mathcal{B}, \mathcal{C})$  is also a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, s^*))$ , vice versa.*

---

**Algorithm 1:** An algorithm solving  $(\mathcal{B}, \mathcal{C})$ 

---

**Data:** Integers  $q_1, q_2 > 1$  with respect to which regularity and alignment are defined; A multiset  $\mathcal{B} = \{B_i\}_{i=1}^{|\mathcal{B}|}$  of regular blocks with sizes sorted in descending order; A set  $\mathcal{C}$  of non-overlapping containers.

**Result:** A solution to  $(\mathcal{B}, \mathcal{C})$  if one exists, or  $\perp$  otherwise.

```
for  $i \in [|\mathcal{B}|]$  do
   $s^* := \min\{s : \forall B \in \mathcal{B}, s \succeq \text{Size}(B)\}$  ;
  /* Let  $s^*$  be the smallest size which is sufficient to cover each block in
      $\mathcal{B}$ . */
   $\mathcal{C} \leftarrow \sigma(\mathcal{C}, s^*)$  ;
  /* Cut the containers in  $\mathcal{C}$  into the smallest set of regular aligned
     containers each with size at most  $s^*$ . */
   $\mathcal{S} := \{C \in \mathcal{C} : \text{Size}(C) \succeq \text{Size}(B_i)\}$  ;
  /* Consider the subset of containers which are large enough to contain  $B_i$ 
     which is the largest block at hand. */
  if  $\mathcal{S} = \emptyset$  then
    return  $\perp$  ;
  /* If no such subset exists, the algorithm reports a failure. */
   $\bar{\mathcal{S}} := \arg \min_{C \in \mathcal{S}} \{\text{Size}(C)\}$  ;
  /* Among those containers, consider a subset which have the smallest
     size. */
   $C^* \leftarrow_{\mathcal{S}} \bar{\mathcal{S}}$  ;
  /* Choose an arbitrary container  $C^*$  from this subset. */
   $\text{Loc}_i := \text{Loc}(C^*)$  ;
   $\mathcal{B} := \mathcal{B} \setminus \{B_i\}$  ;
   $\mathcal{C} := \mathcal{C} \setminus \{C^*\} \cup \{C^* \setminus B_i(\text{Loc}_i)\}$  ;
  /* Assign  $B_i$  to the location of  $C^*$  so that their lower-left corners
     overlap. Remove  $B_i$  from the set of blocks at hand. Remove the space
     occupied by  $B_i$  from the container  $C^*$ . */
return  $\{\text{Loc}_i\}_{i=1}^{|\mathcal{B}|}$ 
```

---

*Proof:* We first show that  $(\mathcal{B}, \mathcal{C})$  has a solution if  $(\mathcal{B}, \sigma(\mathcal{C}, [w_{max}, h_{max}]))$  has a solution. Suppose the latter holds. By Definition 10, there exists a simultaneous assignment  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  which satisfies the following

$$\begin{cases} \forall B_i, B_j \in \mathcal{B} \text{ s.t. } i \neq j, B_i(x_i, y_i) \cap B_j(x_j, y_j) = \emptyset, \\ \forall B_i \in \mathcal{B}, \exists C_i \in \sigma(\mathcal{C}, [w_{max}, h_{max}]) \text{ s.t. } B_i(x_i, y_i) \subseteq C_i, \\ \forall B_i \in \mathcal{B}, B_i(x_i, y_i) \text{ is aligned, i.e., } w_B^i | x_i \text{ and } h_B^i | y_i. \end{cases}$$

Fix  $i \in [|\mathcal{B}|]$ . By the definition of  $\sigma$  (Definition 11),  $\sigma(\mathcal{C}, [w_{max}, h_{max}]) = \bigcup_{C \in \mathcal{C}} \sigma(C, [w_{max}, h_{max}])$ . Therefore since  $C_i \in \sigma(\mathcal{C}, [w_{max}, h_{max}])$ , there must exist  $C_i^* \in \mathcal{C}$  such that  $C_i \in \sigma(C_i^*, [w_{max}, h_{max}])$ . Using Definition 11 again,  $C_i^* = \bigcup_{C' \in \sigma(C_i^*, [w_{max}, h_{max}])} C'$ . Therefore  $C_i \subseteq C_i^*$ . From these observations, the second constraint of a solution implies that

$$\forall B_i \in \mathcal{B}, \exists C_i^* \in \mathcal{C} \text{ s.t. } B_i(x_i, y_i) \subseteq C_i^*$$

which together with the first and the third constraints implies that  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  is a solution to  $(\mathcal{B}, \mathcal{C})$ .

Next, we show that  $(\mathcal{B}, \mathcal{C})$  has a solution only if  $(\mathcal{B}, \sigma(\mathcal{C}, [w_{max}, h_{max}]))$  has a solution. By Definition 10, since  $(\mathcal{B}, \mathcal{C})$  has a solution, there exists a simultaneous assignment  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  such that which satisfies the following

$$\begin{cases} \forall B_i, B_j \in \mathcal{B} \text{ s.t. } i \neq j, B_i(x_i, y_i) \cap B_j(x_j, y_j) = \emptyset, \\ \forall B_i \in \mathcal{B}, \exists C_i \in \mathcal{C} \text{ s.t. } B_i(x_i, y_i) \subseteq C_i, \\ \forall B_i \in \mathcal{B}, B_i(x_i, y_i) \text{ is aligned, i.e., } w_B^i | x_i \text{ and } h_B^i | y_i. \end{cases}$$

We have to show that for any  $B_i(x_i, y_i)$ , it must be contained in one of the containers in  $\sigma(\mathcal{C}, [w_{max}, h_{max}])$ . First, we know that  $B_i(x_i, y_i) \subseteq C_i = \bigcup_{C \in \sigma(\mathcal{C}_i, [w_{max}, h_{max}])} C$ . So, there exists at least one container  $\bar{C} \in \sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  such that  $\bar{C} \cap B_i(x_i, y_i) \neq \emptyset$ . Let  $\text{Reg}(x_C, y_C, w_C, h_C) := \bar{C}$  and  $\text{Reg}(x_i, y_i, w_i, h_i) := B_i(x_i, y_i)$ .

Case I:  $\text{Size}(\bar{C}) \succeq \text{Size}(B_i)$ . By Lemma 1, we have  $B_i(x_i, y_i) \subseteq \bar{C}$ .

Case II:  $w_C \leq w_B$  and  $h_C > h_B$ . Without loss of generality, let  $\bar{C}$  be the tallest container in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  which overlaps with  $B_i(x_i, y_i)$ . By Lemma 2, we have  $x_i \leq x_C < x_C + w_C \leq x_i + w_i$ , and  $y_C \leq y_i < y_i + h_i \leq y_C + h_C$ . Suppose there is a container  $R = \text{Reg}(x, y, w, h)$  overlaps the region  $\text{Reg}(x_i, y_C, x_C - x_i, h_C)$ . By Lemma 2, we have  $x \leq x_i < x_i + w_i \leq x + w$  if  $w \geq w_i$ , or  $x_i \leq x < x + w \leq x_i + w_i$  if  $w \leq w_i$ . The former case  $R$  overlaps with  $\bar{C}$ , thus we must have the latter case with a refinement that  $x_i \leq x < x + w \leq x_C$ . It is similar for the a container which overlaps the region  $\text{Reg}(x_C + w_C, y_C, (x_i + w_i) - (x_C + w_C), h_C)$ . That is, any container covering the region  $\text{Reg}(x_i, y_C, w_i, h_C)$  must stay in the interval  $[x_i, x_i + w_i)$  for the width dimension.

Now, we want to find a smallest set of containers in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  covering the region  $\text{Reg}(x_i, y_C, x_C - x_i, h_C)$ . As shown above, those containers in the set must stay in the interval  $[x_i, x_C)$  for the width dimension. We cannot have a container in the set taller than  $\bar{C}$  as this container must overlap with  $B_i(x_i, y_i)$ , which contradicts that  $\bar{C}$  is the tallest. That is, every container in the set has size  $\preceq [x_C - x_i, h_C]$ . By Lemma 1, every container in the set is a contained by  $\text{Reg}(x_i, y_C, x_C - x_i, h_C)$ . So, we have a set of containers in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  where the union of all containers in it is the regular aligned region  $\text{Reg}(x_i, y_C, x_C - x_i, h_C)$ , which contradicts the minimality of  $\sigma$ .

Case III:  $w_C > w_B$  and  $h_C \geq h_B$ . A similar argument to the one for case II can be applied to show that this case cannot happen.

Case IV:  $\text{Size}(\bar{C}) \preceq \text{Size}(B_i)$ . If there exists another container in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  which overlaps with  $B_i(x_i, y_i)$  such that it is wider or taller than  $B_i(x_i, y_i)$ , we would choose that container as  $\bar{C}$  and go to case I, II or III. Here, we consider every container in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$  which overlaps with  $B_i(x_i, y_i)$  has size  $\preceq \text{Size}(B_i)$ , and by Lemma 1,  $B_i(x_i, y_i)$  contains all these containers. That is, their union is exactly the regular aligned region  $B_i(x_i, y_i)$ , which contradicts the minimality of  $\sigma$ .

The above four cases conclude that after the cut by  $\sigma$ , any block  $B_i(x_i, y_i)$  must be contained in one of the containers in  $\sigma(\mathcal{C}_i, [w_{max}, h_{max}])$ , which implies that  $(\mathcal{B}, \sigma(\mathcal{C}, [w_{max}, h_{max}]))$  has a solution.

Note that we did not change the assignments of blocks during the whole proof, which means that any solution to  $(\mathcal{B}, \mathcal{C})$  is also a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, s^*))$ , vice versa.  $\blacksquare$

In Lemma 5,  $B_1$  is the largest unpacked block at hand, so the width or the height of  $\text{Size}(B_1)$  is the longest among the unpacked blocks. This means that the longer side of  $\text{Size}(B_1)$  equals to the corresponding side in  $s^*$ , i.e., the longer side of  $\text{Size}(B_1)$  must fit the corresponding side of every container in  $S$ . Then, the problem becomes a one-dimensional packing problem for the block.

**Lemma 6.** *Let  $\mathcal{B}$  be a multiset of regular blocks. Let  $\mathcal{C}$  be a set of non-overlapping regular aligned containers such that there exist  $C, C' \in \mathcal{C}$  with  $\text{Size}(C) = \text{Size}(C')$ . Suppose  $(\mathcal{B}, \mathcal{C})$  has a solution  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$ . Then  $\{(x'_i, y'_i)\}_{i=1}^{|\mathcal{B}|}$  defined below is also a solution to  $(\mathcal{B}, \mathcal{C})$ .*

$$(x'_i, y'_i) = \begin{cases} (x_i, y_i) - \text{Loc}(C) + \text{Loc}(C') & B_i(x_i, y_i) \subseteq C \\ (x_i, y_i) - \text{Loc}(C') + \text{Loc}(C) & B_i(x_i, y_i) \subseteq C' \\ (x_i, y_i) & \text{otherwise} \end{cases}$$

where operations on tuples are performed coordinate-wise.

*Proof:* For every block  $B_i$  where  $[w_i, h_i] := \text{Size}(B_i)$  and  $B_i(x_i, y_i) \subseteq C$ , we have  $w_i \mid x_i$  and  $h_i \mid y_i$  as  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  is a solution to  $(\mathcal{B}, \mathcal{C})$ . We must have  $\text{Size}(B_i) \preceq \text{Size}(C)$  or otherwise  $C$  is not large enough to contain  $B_i$ . Then, we have  $w_i \mid x_C$  and  $h_i \mid y_C$  where  $(x_C, y_C) := \text{Loc}(C)$ . As  $\text{Size}(C) = \text{Size}(C')$ , we also have  $w_i \mid x_{C'}$  and  $h_i \mid y_{C'}$  where  $(x_{C'}, y_{C'}) = \text{Loc}(C')$ . So, we have  $w_i \mid (x_i - x_C + x_{C'})$  and  $h_i \mid (y_i - y_C + y_{C'})$ . On the other hand, we have  $x_C \leq x_i + w_i \leq x_C + w_C$  and  $y_C \leq y_i + h_i \leq y_C + h_C$  where  $[w_C, h_C] := \text{Size}(C) = \text{Size}(C')$ . So, we have  $x_{C'} \leq x'_i + w_i \leq x_{C'} + w_C$  and  $y_{C'} \leq y'_i + h_i \leq y_{C'} + h_C$ . That is,  $B_i(x'_i, y'_i) \subseteq C'$  is a regular aligned region.

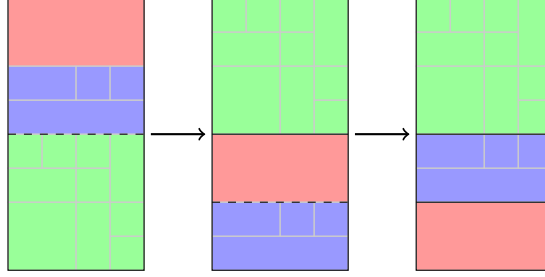


Fig. 4: An example on the reason why we can put the largest block at the bottom left corner. By Lemma 1, no block in the solution can cross the boundary shown by the dashed line. So, we can swap the upper half (red and blue) and the lower half (green) and obtain another solution. A similar argument can be used repeatedly until the red block is put at the bottom.

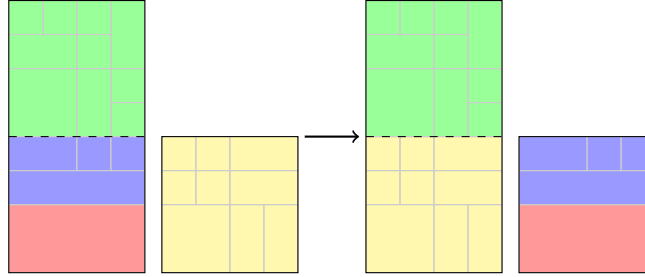


Fig. 5: An example on the reason why can select a smallest container in  $\mathcal{S}$ , which is the yellow container in the figure. Suppose there is a solution that the red block is put in a larger container. We can first swap the red block to the bottom as shown in Fig. 4. The size of the region below the dashed line is the same as the size of the yellow container. By Lemma 1, no block in the solution can cross the boundary shown by the dashed line. So, we can simply swap all the blocks below the dashed line with those in the yellow container.

All the blocks packed in a container is non-overlapping, thus after moving all the blocks in  $C$  to  $C'$ , they are still non-overlapping. This means that we can swap the blocks contained in  $C$  and  $C'$  and obtain another solution. ■

**Lemma 7.** Let  $\mathcal{B} = \{B_i\}_{i=1}^{|\mathcal{B}|}$  be a multiset of regular blocks with sizes sorted in descending order, and  $\mathcal{C}$  be a set of non-overlapping containers. Suppose  $(\mathcal{B}, \mathcal{C})$  has a solution. Let  $\mathcal{B}'$  and  $\mathcal{C}'$  be the value of  $\mathcal{B}$  and  $\mathcal{C}$  after the first ( $i = 1$ ) iteration. More precisely, let

$$\begin{aligned} \hat{s} &:= \min\{s : \forall B \in \mathcal{B}, s \succeq \text{Size}(B)\} \\ \mathcal{S} &:= \{C \in \sigma(\mathcal{C}, \hat{s}) : \text{Size}(C) \succeq \text{Size}(B_1)\} \\ C^* &\in \arg \min_{C \in \mathcal{S}} \{\text{Size}(C)\} \\ \mathcal{B}' &:= \mathcal{B} \setminus \{B_1\} \\ \mathcal{C}' &:= \mathcal{C} \setminus \{C^*\} \cup \{C^* \setminus B_1(\text{Loc}(C^*))\} \end{aligned}$$

Then  $(\mathcal{B}', \mathcal{C}')$  also has a solution.

*Proof:* By Lemma 5,  $(\mathcal{B}, \mathcal{C})$  has a solution if and only if  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  has a solution. We can thus assume that  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  has a solution. We first argue that there must exist a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$  (Claim 1).

With the above claim, it remains to prove that it is fine to pack  $B_1$  at the lower-left corner of  $C^*$ . Formally, we argue that if there exists a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$ , then there also exists a solution  $\{(x_i^\dagger, y_i^\dagger)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $(x_1^\dagger, y_1^\dagger) = \text{Loc}(C^*)$  (Claim 2). Once this is shown, the truthfulness of the lemma can then be verified by inspection. ■

**Claim 1.** If  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  has a solution, then there exists a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$ .

*Proof:* Let  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  be a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$ . There exists  $\bar{C} \in \mathcal{S} \subseteq \sigma(\mathcal{C}, \hat{s})$  such that  $B_1(x_1, y_1) \subseteq \bar{C}$ . Suppose  $\bar{C} \in \arg \min_{C \in \mathcal{S}} \{\text{Size}(C)\}$ , then since  $\text{Size}(\bar{C}) = \text{Size}(C^*)$ , we can use Lemma 6 and construct a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$ .

Suppose otherwise  $\bar{C} \in \mathcal{S} \setminus \arg \min_{C \in \mathcal{S}} \{\text{Size}(C)\}$ . Parse  $\hat{s}$  as  $\hat{s} = [\hat{w}, \hat{h}]$ . Since  $B_1$  is the largest block in  $\mathcal{B}$ , we have  $w_1 = \hat{w}$  or  $h_1 = \hat{h}$ . We consider the case  $w_1 = \hat{w}$ . Similar arguments can be applied to the case  $h_1 = \hat{h}$ . Let  $\bar{s} = [\bar{w}, \bar{h}] = \text{Size}(\bar{C})$ , and  $s^* = [w^*, h^*] = \text{Size}(C^*)$ . By the definition of  $\sigma$ , we have  $\bar{w} \leq \hat{w}$  and  $w^* \leq \hat{w}$ . On the other hand, since  $B_1(x_1, y_1) \subseteq \bar{C}$ , we have  $\bar{w} \geq w_1 = \hat{w}$ . By the construction of  $C^*$ , we have  $w^* \geq w_1 = \hat{w}$ . Summarizing the above, we have  $\bar{w} = w^* = w_1$ .

By the assumption that  $\bar{C} \in \mathcal{S} \setminus \arg \min_{C \in \mathcal{S}} \{\text{Size}(C)\}$ , we must have  $\bar{h} > h^*$ . By the construction of  $C^*$ , we have  $h^* \geq h_1$ . Since  $\bar{C}$ ,  $C^*$  and  $B_1$  are all regular,  $\bar{h}$ ,  $h^*$ , and  $h_1$  are all powers of  $q_2$ , hence  $h^* | \bar{h}$  and  $h_1 | h^*$ . Let  $(\bar{x}, \bar{y}) = \text{Loc}(\bar{C})$ . Consider the sets of regions  $\mathcal{R} = \{\text{Reg}(\bar{x}, \bar{y} + h^* \cdot j, w^*, h^*) : j = 0, 1, \dots, \frac{\bar{h}}{h^*} - 1\}$ . By construction, each  $R \in \mathcal{R}$  is regular, aligned, and non-overlapping, and  $\bigcup_{R \in \mathcal{R}} R = \bar{C}$ . Furthermore, for each  $R \in \mathcal{R}$ , we have  $\text{Size}(R) = \text{Size}(C^*) \succeq \text{Size}(B_1)$ . By Lemma 1, for each  $R \in \mathcal{R}$ , either  $R \cap B_1(x_1, y_1) = \emptyset$  or  $R \cap B_1(x_1, y_1) = B_1(x_1, y_1)$ . However,  $B_1(x_1, y_1) \subseteq \bigcup_{R \in \mathcal{R}} R$ . By the pigeonhole principle, there must exist  $R^* \in \mathcal{R}$  for which  $B_1(x_1, y_1) \subseteq R^*$ .

We show that a block is either contained in  $R^*$  or it does not overlap with  $R^*$  at all. More precisely, for all  $j = 1, \dots, m$ , either  $B_j(x_j, y_j) \cap R^* = \emptyset$  or  $B_j(x_j, y_j) \cap R^* = B_j(x_j, y_j)$ .

Note that  $w_j \leq \hat{w} = w^*$ . Suppose  $h_j \leq h^*$ , then we can use Lemma 1 and conclude that either  $B_j(x_j, y_j) \cap R^* = \emptyset$  or  $B_j(x_j, y_j) \cap R^* = B_j(x_j, y_j)$ .

We now tackle the difficult case where  $h_j > h^*$ . We first establish some notations. Recall that both  $B_j$  and  $R^*$  are regular, therefore both  $h_j$  and  $h^*$  are powers of  $q_2$ . Since  $h_j > h^*$ , we have  $h^* | h_j$ . Also recall that both  $B_j(x_j, y_j)$  and  $R^*$  are aligned. Let  $(x^*, y^*) = \text{Loc}(R^*)$ . We have  $h_j | y_j$  and  $h^* | y^*$ . Let  $\alpha_j, \kappa_j, \kappa^*$  be non-negative integers such that  $y^* = \kappa^* h^*$ ,  $y_j = \kappa_j h_j = \alpha_j \kappa_j h^*$ . Suppose towards contradiction that the lemma does not hold, then  $[y^*, y^* + h^*) \cap [y_j, y_j + h_j) \neq \emptyset$ . We argue that we must have  $y_j \leq y^* < y^* + h^* \leq y_j + h_j$ . Suppose this is the case, then  $B_j(x_j, y_j) \cap B_1(x_1, y_1) \neq \emptyset$ , contradicting the assumption that  $\{(x_i, y_i)\}_{i=1}^{|\mathcal{B}|}$  is a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$ , and hence the lemma is proven.

Suppose the above assumption does not hold, then there are two possible cases. For the first case, we have  $y^* < y_j < y^* + h^* \leq y_j + h_j$ . From the first two inequalities, we have  $\kappa^* h^* < \alpha_j \kappa_j h^* < (\kappa^* + 1) h^*$ , which is impossible as there is no space to put an integer  $\alpha_j \kappa_j$  between the consecutive integers  $\kappa^*$  and  $\kappa^* + 1$ . Similarly, for the second case, we have  $y_j \leq y^* < y_j + h_j < y^* + h^*$ . From the last two inequalities, we have  $\kappa^* h^* < \alpha_j (\kappa_j + 1) h^* < (\kappa^* + 1) h^*$ , which is again impossible.

To recap, we have shown that for all  $j$  we have either  $B_j(x_j, y_j)$  is completely contained in  $R^*$ , or it does not overlap with  $R^*$  at all. This means that cutting the container  $R^*$  from  $\bar{C}$  does not invalidate the solution. Formally, we have that  $(\mathcal{B}, (\sigma(\mathcal{C}, \hat{s}) \setminus \{\bar{C}\}) \cup \{\bar{C} \setminus R^*, R^*\})$  has a solution. Note that  $R^*, C^* \in (\sigma(\mathcal{C}, \hat{s}) \setminus \{\bar{C}\}) \cup \{\bar{C} \setminus R^*, R^*\}$  and  $\text{Size}(R^*) = \text{Size}(C^*)$ . Therefore by Lemma 6 there exists a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, (\sigma(\mathcal{C}, \hat{s}) \setminus \{\bar{C}\}) \cup \{\bar{C} \setminus R^*, R^*\})$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$ . However, a solution to  $(\mathcal{B}, (\sigma(\mathcal{C}, \hat{s}) \setminus \{\bar{C}\}) \cup \{\bar{C} \setminus R^*, R^*\})$  is also a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  and therefore our claim is proven.  $\blacksquare$

**Claim 2.** *If there exists a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $B_1(x_1^*, y_1^*) \subseteq C^*$ , then there also exists a solution  $\{(x_i^\dagger, y_i^\dagger)\}_{i=1}^{|\mathcal{B}|}$  to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$  with  $(x_1^\dagger, y_1^\dagger) = \text{Loc}(C^*)$ .*

*Proof:* We will continue to assume that  $w_1 = \hat{w}$  which implies that  $w_1 = w^*$ . As before, the case where  $h_1 = \hat{h}$  can be dealt with using similar arguments. Now since  $w_1 = w^*$ , it must hold that  $x_1^* = x_1^\dagger$ . Suppose  $y_1^* = y_1^\dagger$ , then the claim is proven. Otherwise, suppose  $y_1^* \neq y_1^\dagger$ . Note that  $y_1^\dagger < y_1^* < y_1^* + h_1 \leq y_1^\dagger + h^*$ . Let  $\delta := y_1^* - y_1^\dagger$ , and let  $(\delta_0, \dots, \delta_\ell)$  be the  $q_2$ -ary representation of  $\delta$  where  $\delta = \sum_{j=0}^{\ell} \delta_j q_2^j$  and  $\ell = \log_{q_2} h^* - 1$ . Let  $j^* \leq \ell$  be the largest integer for which  $\delta_{j^*} \neq 0$ . We define the regions  $R_{j^*} := \text{Reg}(x_1^\dagger, y_1^\dagger + \delta_{j^*}, w^*, q_2^{j^*})$  and  $R'_{j^*} := \text{Reg}(x_1^\dagger, y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*}, w^*, q_2^{j^*})$ . It is straightforward to check that both  $R_{j^*}$  and  $R'_{j^*}$  are regular aligned.

Recall that  $B_1$  and  $C^*$  are regular, therefore  $h_1$  and  $h^*$  are both powers of  $q_2$ . Since  $h_1 \leq h^*$ , we have  $h_1 | h^*$ . Since  $B_1(x_1^*, y_1^*)$  and  $C^*$  are aligned, both  $y_1^*$  and  $y_1^\dagger$  are multiples of  $h_1$ . Therefore  $\delta$  is also a multiple of  $h_1$ . Since we have assumed  $\delta > 0$ , it must be the case that  $h_1 \leq \delta$ . Since  $h_1$  is a power of  $q_2$ , we have  $h_1 \leq q_2^{j^*}$ . Therefore,  $B_1(x_1^*, y_1^*) \subseteq R'_{j^*}$ .

Next, we argue that each block  $B_j(x_j^*, y_j^*)$  is either contained in  $R_{j^*}$ , or contained in  $R'_{j^*}$ , or does not overlap with  $R_{j^*}$  and  $R'_{j^*}$  at all. We first note that  $\text{Size}(R_{j^*}) = \text{Size}(R'_{j^*}) = [w^*, q_2^{j^*}]$ , and  $w_j \leq w_1 = w^*$ . Therefore if  $h_j \leq q_2^{j^*}$  we can use Lemma 1 and prove the claim.

Now consider the case where  $h_j > q_2^{j^*}$ . Since  $B_j$  is regular, we have  $h_j \geq q_2^{j^*+1}$ . Note that the lower boundary of  $R_{j^*}$  is located at  $y_1^\dagger$  unit of the second dimension, while the lower boundary of  $R'_{j^*}$  is located at  $y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*}$  unit. The two boundaries therefore have a height difference of  $\delta_{j^*} \cdot q_2^{j^*} < q_2^{j^*+1}$  unit. Thus  $y_j$  cannot be in between the two boundaries or else  $B_j(x_j^*, y_j^*)$  would overlap with  $B_1(x_1^*, y_1^*)$ . In other words, we have either  $y_j^* < y_1^\dagger$  or  $y_j^* \geq y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*}$ .

Consider the first case where  $y_j^* < y_1^\dagger$ . If  $y_1^\dagger < y_j^* + h_j$ , then part of  $B_j$  is contained by  $C^*$  and part of it is contained by some other container(s), which violates the assumption that  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  is a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$ . We must therefore have  $y_j^* + h_j \leq y_1^\dagger$ . This implies that  $B_j(x_j^*, y_j^*)$  does not overlap with  $C^*$ . Hence it also does not overlap with  $R_{j^*}$  and  $R'_{j^*}$ .

Now consider the second case where  $y_j^* \geq y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*}$ . Suppose that  $y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*} \leq y_j^* < y_1^\dagger + (\delta_{j^*} + 1) \cdot q_2^{j^*}$ . Recall that  $R'_{j^*}$  and  $B_j$  are both regular, therefore  $y_1^\dagger = \mu q_2^{j^*}$  and  $y_j^* = \nu q_2^{j^*}$  (since  $h_j$  is a multiple of  $q_2^{j^*+1}$ ) for some non-negative integers  $\mu$  and  $\nu$ . Substituting the expressions, we have

$$\begin{aligned} y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*} &\leq y_j^* < y_1^\dagger + (\delta_{j^*} + 1) \cdot q_2^{j^*} \\ (\mu + \delta_{j^*}) q_2^{j^*} &\leq \nu q_2^{j^*} < (\mu + \delta_{j^*} + 1) q_2^{j^*} \end{aligned}$$

which forces  $\nu = \mu + \delta_{j^*}$  and hence  $y_1^\dagger + \delta_{j^*} \cdot q_2^{j^*} = y_j^*$ . This however contradicts to the assumption that  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  is a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$ , since  $B_j(x_j^*, y_j^*)$  overlaps with  $B_1(x_1^*, y_1^*)$ . Therefore it must be the case that  $y_j^* \geq y_1^\dagger + (\delta_{j^*} + 1) \cdot q_2^{j^*}$  and therefore  $B_j(x_j^*, y_j^*)$  does not overlap with  $R_{j^*}$  and  $R'_{j^*}$ .

To recap, we have constructed two regular aligned regions  $R_{j^*}$  and  $R'_{j^*}$  of the same size, such that they are contained in  $C^*$  and  $R'_{j^*}$  contains  $B_1(x_1^*, y_1^*)$ . Furthermore, each block  $B_j(x_j^*, y_j^*)$  is either contained in  $R_{j^*}$ , or in  $R'_{j^*}$ , or does not overlap with them at all. From these guarantees, we can conclude that  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  is also a solution to  $(\mathcal{B}, (\sigma(\mathcal{C}, \hat{s}) \setminus \{C^*\}) \cup \{R_{j^*}, R'_{j^*}, C^* \setminus (R_{j^*} \cup R'_{j^*})\})$ . We can then apply Lemma 6 and obtain a solution to  $(\mathcal{B}, (\sigma(\mathcal{C}, \hat{s}) \setminus \{C^*\}) \cup \{R_{j^*}, R'_{j^*}, C^* \setminus (R_{j^*} \cup R'_{j^*})\})$ , which is also a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, \hat{s}))$ , such that  $B_1$  is packed into  $R_{j^*}$  instead. Repeating this process until there does not exist  $j^*$  with  $\delta_{j^*} \neq 0$ , we can obtain a solution in which  $(x_1^\dagger, y_1^\dagger)$  is assigned to  $B_1$ . ■

Lemma 7 states a way to formulate the problem after we packed  $B_1$  by packing  $B_1$  at the bottom left corner of  $C^*$ , where  $C^*$  is a smallest container at hand which is large enough to contain  $B_1$ .<sup>6</sup> We present Fig. 4 and 5 to give a high level explanation on why Lemma 7 works. In these figures, we let  $q_1 = q_2 = 2$ , and the red block be the largest block at hand. After applying Lemma 5, the red block must fit to either the width or the height of the container. Here we illustrates the case that the width is fitted. Suppose  $(\mathcal{B}, \mathcal{C})$  has a solution, then  $(\mathcal{B}, \sigma(\mathcal{C}, s^*))$  also has a solution by Lemma 5. Let  $\bar{C}$  be the container  $B_1$  is packed in a solution to  $(\mathcal{B}, \sigma(\mathcal{C}, s^*))$ . If  $\bar{C} \in \mathcal{S}$ , then without loss of generality we can say  $C^* = \bar{C}$ .<sup>7</sup> We can swap the blocks in  $C^*$  such that the largest block can be packed at the bottom left corner while preserving the validity of the solution, which is shown in Fig. 4. If  $\bar{C} \notin \mathcal{S}$ , then we have  $\text{Size}(\bar{C}) \succeq \text{Size}(C^*)$ . Note that one of the side of  $\bar{C}$  has the same length as the one of  $C^*$ . Fig. 5 illustrates that we can swap the blocks in  $C^*$  and some of the blocks in  $\bar{C}$  while preserving the validity of the solution.

#### D. Correctness of Our Algorithm and Time Complexity

With the help of the established lemmas, we are now able to prove Theorem 4.

*Proof:* (Theorem 4) It is straightforward to check that if Algorithm alg:csp terminates without outputting  $\perp$ , then its output is a valid solution to the problem  $(\mathcal{B}, \mathcal{C})$ . The converse of this theorem is the main result of this

<sup>6</sup>To see why, we give a simple example. Suppose we have  $q_1 = q_2 = 2$ , two blocks of sizes  $2 \times 1$  and  $1 \times 2$ , and two aligned containers of sizes  $2 \times 2$  and  $2 \times 1$ . It is easy to see that the problem has a solution. If we choose the  $2 \times 2$  container, which is not the smallest, and pack the largest block, i.e., the  $2 \times 1$  block, into it, then what we left are two  $2 \times 1$  containers. There is no way to pack the remaining  $1 \times 2$  block.

<sup>7</sup>It is trivial that if  $C^*$  and  $\bar{C}$  are distinct containers in  $\mathcal{S}$ , we can swap the blocks contained by them without affecting the validity of the solution as they are both regular aligned containers of the same size.

paper, which states that if  $(\mathcal{B}, \mathcal{C})$  indeed has a solution, then Algorithm `alg:csp` can find one of the solutions. We prove by recursively applying Lemma 7.

By Lemma 7, if  $(\mathcal{B}, \mathcal{C})$  has a solution, then there must exist a solution in which the first (which is the largest) block is packed into certain containers. Specifically, using the notations in Lemma 7, for any container  $C^*$  which is among the smallest containers in  $\sigma(\mathcal{C}, \hat{s})$  which are large enough to contain  $B_1$ , there must exist a solution  $\{(x_i^*, y_i^*)\}_{i=1}^{|\mathcal{B}|}$  where  $(x_1^*, y_1^*) = \text{Loc}(C^*)$ . Note that the first iteration of Algorithm `alg:csp` picks exactly one such  $C^*$  and sets  $(x_1^*, y_1^*) = \text{Loc}(C^*)$ . Using again the notations in Lemma 7, we let  $\mathcal{B}'$  be the multiset of blocks obtained from  $\mathcal{B}$  by removing  $B_1$ , and  $\mathcal{C}'$  be the set of containers obtained from  $\mathcal{C}$  by removing the space occupied by  $B_1$ . Note that the second largest block in  $\mathcal{B}$  is now the largest block in  $\mathcal{B}'$ . We can then apply Lemma 7 again on  $(\mathcal{B}', \mathcal{C}')$  which guarantees that Algorithm `alg:csp` assigns a good location for  $B_2$ . Repeating this process, Algorithm `alg:csp` is able to assign a good location to each block in  $\mathcal{B}$ , and the lemma is thus proven.  $\blacksquare$

Given a set  $\mathcal{C}$  of containers and a set  $\mathcal{B} = \{B_1, \dots, B_m\}$  of blocks. Define  $[w_B^i, h_B^i] := \text{Size}(B_i)$  for  $i = 1, \dots, m$ ,  $w_{\max} := \max_{i=1}^m w_B^i$  and  $h_{\max} := \max_{i=1}^m h_B^i$ . Observe that we only need to keep track of the number of containers of different sizes, we create a 2D array  $\mathbf{A}$ , where  $\mathbf{A}_{i,j}$  stores the number of containers of size  $[q_1^i, q_2^j]$  for  $i \in \{0, \dots, \log_{q_1} w_{\max}\}$ ,  $j \in \{0, \dots, \log_{q_2} h_{\max}\}$ . This array can be initialized in  $\mathcal{O}((\log_{q_1} w_{\max})(\log_{q_2} h_{\max}))$  time. On the other hand, for each input container of size  $[a, b]$ , we can perform the initial cut by  $\sigma$  and register the cut containers to  $\mathbf{A}$  in  $\mathcal{O}((\log_{q_1} a)(\log_{q_2} b))$  time.

**Definition 13** (Layers). A block or a container of size  $[w, h]$  belongs to *layer*  $l$  if and only if  $l = \max\{w, h\}$ .

In Algorithm 1, the blocks are packed sequentially according to their sizes sorted in descending order. That is, the algorithm packs the blocks belonging to the same layer before it packs the blocks belonging to a lower layer. When the algorithm handles the layer  $l$ , every container  $C \in \mathcal{C}$  has  $\text{Size}(C) \preceq [l, l]$ . We can ensure that every container which can contain some block belonging to layer  $l$  has at least one of its sides fitting the corresponding side of the block. That is, when the algorithm finishes packing all the blocks belonging to layer  $l$  and moves to layer  $l'$ , we only need to apply  $\sigma(\mathcal{C}, [q_1^{\lceil \log_{q_1} l' \rceil}, q_2^{\lceil \log_{q_2} l' \rceil}])$  instead of the one shown in the algorithm. The actual procedure is to move the count from  $\mathbf{A}_{i,j}$  to either  $\mathbf{A}_{i-k,j}$ ,  $\mathbf{A}_{i,j-p}$  or  $\mathbf{A}_{i-k,j-p}$  for some  $k \in \{1, 2, \dots, i\}$ ,  $p \in \{1, 2, \dots, j\}$ , where the corresponding  $i, j$  representing the layer  $l = \max\{i, j\}$ . We can actually consider  $k = p = 1$ , as if no block belongs to the new layer, we can simply move again to a lower layer. For each case when  $k = p = 1$ , we only need to multiply  $\mathbf{A}_{i,j}$  by  $q_1$ ,  $q_2$  or  $q_1 q_2$  respectively and add it to the count stored in the corresponding new position, which takes  $\mathcal{O}(1)$  time. That is, the total time complexity of  $\sigma$  for moving from a layer to a lower one during the whole algorithm is  $\mathcal{O}((\log_{q_1} w_{\max})(\log_{q_2} h_{\max}))$ .

We now consider the block assignment in layer  $l$ . Without loss of generality, we assume that the largest block at hand has size  $[l, m]$  for some  $m \leq l$ . The only containers which can contain the block must be in the same layer. If there is a container with size equals to the size of the block, i.e.,  $\mathbf{A}_{\log_{q_1} l, \log_{q_2} m} > 0$ , then that container is accountable for the block assignment since it is the smallest container which can contain the block. For the case  $\mathbf{A}_{\log_{q_1} l, \log_{q_2} m} = 0$ , we look for the smallest  $j > \log_{q_2} m$  such that  $\mathbf{A}_{\log_{q_1} l, j} > 0$ , which takes  $\mathcal{O}(\log_{q_2} l)$  time by sequential search. Recall that the blocks are sorted according to their size in descending order. Due to the fact that an  $l \times q_2^j$  container can be used to pack  $q_2^j/m$  number of  $l \times m$  blocks, we can perform successive assignments of  $l \times q_2^j$  blocks without actually applying  $\sigma$  to the container after each assignment. After all the  $l \times m$  blocks are packed, if the next block to be packed has size  $l \times (m/q_2^k)$  for some  $k$ , then the successive assignment for the new block still works. Hence, we can simulate  $\sigma$  by simple subtraction until the width of the new block is no longer  $l$ . A similar argument can be applied when the largest block at hand has size  $[m, l]$ .

We now calculate an upper bound on the time complexity of Algorithm 1. We have to search a container for every block, where the worst case for a search can be upper bounded by  $\mathcal{O}(\max\{\log_{q_1} w_{\max}, \log_{q_2} h_{\max}\})$ . At the end of a successive assignment sequence, we have to cut the unused region of the container, where the cut takes either  $\mathcal{O}(\log_{q_1} l)$  or  $\mathcal{O}(\log_{q_2} l)$  time depending on whatever we cut the width or the height. The time complexity for the cuts after successive assignments is upper bounded by  $\mathcal{O}(\log_{q_1} w_{\max} + \log_{q_1} (w_{\max}/q_1) + \dots + \log_{q_1} q_1 + \log_{q_2} h_{\max} + \log_{q_2} (h_{\max}/q_2) + \dots + \log_{q_2} q_2) = \mathcal{O}(\log_{q_1}^2 w_{\max} + \log_{q_2}^2 h_{\max})$ . The time of  $\sigma$  for moving the layer takes  $\mathcal{O}((\log_{q_1} w_{\max})(\log_{q_2} h_{\max}))$ , which can be absorbed by  $\mathcal{O}(\log_{q_1}^2 w_{\max} + \log_{q_2}^2 h_{\max})$ . Let  $W_{\max}$  and  $H_{\max}$  be the maximum width and height among the input containers. The overall time complexity is  $\mathcal{O}(m \max\{\log_{q_1} w_{\max}, \log_{q_2} h_{\max}\} + \log_{q_1}^2 w_{\max} + \log_{q_2}^2 h_{\max} + |\mathcal{C}|(\log_{q_1} W_{\max})(\log_{q_2} H_{\max}))$ .



At last, we show the time complexity when we apply the rectangle packing problem to solve our original goal: Decide whether a two-channel prefix code exists for a given multiset of codeword lengths. Note that  $\ell_1^{\max} = \log_{q_1} w_{\max}$  and  $\ell_2^{\max} = \log_{q_2} h_{\max}$ . At the beginning, we only have one regular aligned container  $\text{Reg}(0, 0, w_{\max}, h_{\max})$ . Including the time for sorting, the time complexity is  $\mathcal{O}(m \log m + m \max\{\ell_1^{\max}, \ell_2^{\max}\} + (\ell_1^{\max})^2 + (\ell_2^{\max})^2)$ .

## V. CONCLUDING REMARKS

In this paper, we closed the gap where the two-channel Kraft inequality fails by formulating a decision procedure for the existence of two-channel prefix codes via solving a rectangle packing problem with dimension-wise alignment constraints.

## APPENDIX

We first prove the Kraft inequality.

*Proof:* Let  $N$  be an arbitrary positive integer. Consider

$$\begin{aligned} & \left( \sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} \right)^N = \sum_{j_1=1}^m \sum_{j_2=1}^m \cdots \sum_{j_N=1}^m \left( \prod_{i=1}^n q_i^{-\sum_{k=1}^N \ell_i^{j_k}} \right) \\ & = \sum_{k_1=1}^{N\ell_1^{\max}} \sum_{k_2=1}^{N\ell_2^{\max}} \cdots \sum_{k_n=1}^{N\ell_n^{\max}} A_{k_1, k_2, \dots, k_n} \prod_{i=1}^n q_i^{-k_i}, \end{aligned}$$

where  $A_{k_1, k_2, \dots, k_n}$  is the coefficient of  $\prod_{i=1}^n q_i^{-k_i}$  in  $(\sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j})^N$ .

Note that  $A_{k_1, k_2, \dots, k_n}$  gives the total number of sequences of  $N$  codewords with a total length of  $k_i$  symbols in the  $i$ -th channel for all  $i = 1, 2, \dots, n$ . Since the code is uniquely decodable, these code sequences must be distinct. So, the number  $A_{k_1, k_2, \dots, k_n}$  must be no more than the total number of distinct sequences where there are  $k_i$  symbols in the  $i$ -th channel for all  $i = 1, 2, \dots, n$ . That is,

$$A_{k_1, k_2, \dots, k_n} \leq \prod_{i=1}^n q_i^{k_i}. \quad (4)$$

Now, substitute (4) into (4) and get

$$\left( \sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} \right)^N \leq \sum_{k_1=1}^{N\ell_1^{\max}} \sum_{k_2=1}^{N\ell_2^{\max}} \cdots \sum_{k_n=1}^{N\ell_n^{\max}} 1.$$

Since this inequality holds for any  $N$ , so we let  $N \rightarrow \infty$  and obtain our desired Kraft inequality. ■

With the help of the Kraft inequality, we can now prove the entropy bound.

*Proof:* Recall that  $H_D(Z) = -\sum_{j=1}^m p_j \log_D p_j$ . Consider

$$\begin{aligned} & \sum_{j=1}^m p_j \sum_{i=1}^n \log_D q_i^{\ell_i^j} - H_D(Z) \\ & = \sum_{j=1}^m p_j \log_D \left( \prod_{i=1}^n q_i^{\ell_i^j} \right) + \sum_{j=1}^m p_j \log_D p_j \\ & = (\ln D)^{-1} \sum_{j=1}^m p_j \ln \left( p_j \prod_{i=1}^n q_i^{\ell_i^j} \right) \\ & \geq (\ln D)^{-1} \sum_{j=1}^m p_j \left( 1 - \left( p_j \prod_{i=1}^n q_i^{\ell_i^j} \right)^{-1} \right) \\ & = (\ln D)^{-1} \left( 1 - \sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} \right) \\ & \geq 0, \end{aligned}$$

where (5) follows the inequality  $\ln a \geq 1 - 1/a$  for any  $a > 0$ , and (5) follows the Kraft inequality (1).

The equality in (5) holds if and only if  $p_j \prod_{i=1}^n q_i^{\ell_i^j} = 1$  for all  $j$ , or equivalently,  $\sum_{i=1}^n \log_D q_i^{\ell_i^j} = -\log_D p_j$  for all  $j$ . Under this condition, we have

$$\sum_{j=1}^m \prod_{i=1}^n q_i^{-\ell_i^j} = \sum_{j=1}^m p_j = 1.$$

So, the equality in (5) also holds, which means that the bound is tight. ■

#### REFERENCES

- [1] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, Oct 1952.
- [2] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.
- [3] R. M. Fano, *The transmission of information*. Massachusetts Institute of Technology, Research Laboratory of Electronics, 1949.
- [4] J. Van Leeuwen, "On the construction of Huffman trees," in *ICALP*, 1976, pp. 382–410.
- [5] "List of recommendation ITU-T E.164 assigned country codes," International Telecommunications Union, 2011.
- [6] F. Yergeau, "UTF-8, a transformation format of ISO 10646," 2003.
- [7] L. G. Kraft, "A device for quantizing, grouping, and coding amplitude-modulated pulses," Ph.D. dissertation, Massachusetts Institute of Technology, 1949.
- [8] B. McMillan, "Two inequalities implied by unique decipherability," *IRE Transactions on Information Theory*, vol. 2, no. 4, pp. 115–116, 1956.
- [9] H. Yao and R. W. Yeung, "Zero-error multichannel source coding," in *2010 IEEE Information Theory Workshop on Information Theory (ITW 2010, Cairo)*, Jan 2010, pp. 1–5.
- [10] R. W. Yeung, *Information Theory and Network Coding*. Springer, 2008.
- [11] R. E. Korf, "Optimal rectangle packing: Initial results," in *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling*. AAAI Press, 2003, pp. 287–295.
- [12] J. Karush, "A simple proof of an inequality of McMillan," *IRE Transactions on Information Theory*, vol. 7, no. 2, pp. 118–118, 1961.
- [13] C. Ye and R. W. Yeung, "Some basic properties of fix-free codes," *IEEE Transactions on Information Theory*, vol. 47, no. 1, pp. 72–87, Jan 2001.
- [14] W. D. Blizard, "Multiset theory," *Notre Dame Journal of Formal Logic*, vol. 30, no. 1, pp. 36–66, 12 1988.