Constrained Codes for Two-Dimensional Channels

Keren Censor

Constrained Codes for Two-Dimensional Channels

Research Thesis

Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Computer Science

Keren Censor

Submitted to the Senate of the
Technion - Israel Institute of Technology

Adar 5766 Haifa March 2006

The research thesis was done under the supervision of Prof. Tuvi Etzion in the Department of Computer Science.

I thank Prof. Tuvi Etzion for his devoted guidance.

I thank little Ofek, Noga, Neta and Almog, for the light they shine.

The generous financial help of the Technion is gratefully acknowledged.
I am also grateful to the Jeanette and Samuel Lubell Foundation.

Contents

List of Figures

Abstract

In digital data storage systems, such as magnetic and optical storage devices, the recorded data has to satisfy certain constraints that are imposed by the physical structure of the media. One of the most frequently investigated type of constraints are the *(d,k) run-length limited* (RLL) constraints. A binary sequence satisfies a one-dimensional (d, k) constraint if every run of zeroes has length at least d and at most k.

Recent developments in optical storage, especially in holographic memory, regard the recorded data as two-dimensional. A one-dimensional constraint has to be satisfied in each of the array directions. Similarly to the one-dimensional case, the capacity of a two-dimensional constraint Θ is defined as:

$$C(\Theta) = \lim_{n,m\to\infty} \frac{\log_2 N(n, m \mid \Theta)}{nm},$$

where $N(n, m \mid \Theta)$ is the number of arrays of size $n \times m$ that satisfy Θ. Few connectivity models have been proposed in the literature to handle two-dimensional data: the diamond model, the square model, the hexagonal model, and the triangular model. The constraints may be asymmetric, i.e. vary among the different directions.

In this work, we derive some new methods for determining zero and positive capacity. We generalize a technique for proving zero capacity, which is based on scanning a Θ-constrained array whose labels are partially known, and counting the number of possible ways to label the rest of the array. This method provides an upper bound for the number of constrained arrays of size $n \times m$, which is small enough to determine that $C(\Theta) = 0$.

For proving positive capacity of some constraints, we define shapes which can tile the plane. Given such a shape, we find two different valid ways to label it. We then show that tiling the plane with copies of the shape, where each copy can have either one of the two labels, results in a Θ-constrained

array. This provides a lower bound for the number of constrained arrays of size $n \times m$, which is large enough to determine that $C(\Theta) > 0$.

We apply the above methods to the different connectivity models in order to characterize their zero/positive capacity regions. We consider asymmetric constraints in the diamond model, and provide an almost complete characterization of the positive capacity region.

In the triangular model, we show that $C(d, d+3) = 0$ for every $d \geq 3$. For $d \equiv 1(mod\ 4)$, $d \geq 5$, we show a tight characterization: $C(d, k) > 0$ if and only if $k \geq d + 4$. Together with the former result, it implies that for other values of d, the gaps between the known zero and positive capacity regions are relatively small.

Finally, in the square model we show that $C(d, d+3) = 0$ for every $d \geq 1$.

Abbreviations and Notations

$N(n \mid \Theta)$	number of Θ-constrained sequences of length n
$N(n, m \mid \Theta)$	number of Θ-constrained arrays of size $n \times m$
$C(\Theta)$	capacity of the constraint Θ
(d, k)-RLL	constraint in which the number of *zeroes* between every pair of consecutive *ones* is at least d and at most k
(d_1, k_1, d_2, k_2)-RLL	two-dimensional constraint where (d_1, k_1) is the horizontal constraint and (d_2, k_2) is the vertical constraint
$C_{\diamond}(d, k)$	capacity of the (d, k) constraint in the diamond model
$C_{\boxplus}(d, k)$	capacity of the (d, k) constraint in the square model
$C_{\hexagon}(d, k)$	capacity of the (d, k) constraint in the hexagonal model
$C_{\triangle}(d, k)$	capacity of the (d, k) constraint in the triangular model
$[n \times m, k \times l]$ tile	$n \times m$ array from which a $k \times l$ array was removed from the upper right corner

Chapter 1

Introduction

Constraint coding is widely used in digital storage applications, particularly magnetic and optical storage devices [10, 11]. In such systems, the physical structure of the storage device imposes constraints on the recorded data. This chapter introduces the field of constrained coding, and describes our main lines of research.

1.1 Physical Constraints in Digital Storage Systems

Magnetic storage devices consist of tracks of magnets. When the data is recorded, a bit *one* is represented by a reversal of the magnetic polarity along the data track, and no reversal of the polarity represents a *zero*. While reading the data, the head which reads responds to a polarity change by an induced voltage. When no change occurs, no voltage is produced. A sufficiently high voltage is considered as a *one*, and otherwise the bit is considered to be a *zero*. On one hand, if successive *ones* are too close, the voltage levels read by them might interfere with each other. Hence there is a lower bound on the number of *zeroes* between successive *ones* that are allowed in the recorded data. On the other hand, the clock of the device is adjusted when high voltages are read and a *one* is detected. To avoid clock drifting, that might cause erroneous recovery of data, there is an upper bound on the number of *zeroes* between successive *ones* that are allowed in the recorded data.

When recording data on an optical device such as a CD, the bit *one* is represented as a peak on the surface. In order to read the data, a laser beam is projected. The light is reflected from the surface, and when reading a peak, a destructive interference occurs. Therefore the detector sees darkness and interprets it as the bit *one*, and otherwise the bit is a *zero*. On one hand, in order for the detector not to miss the peak, the peak has to be wide enough, which implies a lower bound on the number of *zeroes* between successive *ones* that are allowed in the recorded data. On the other hand, reading peaks allows the detector to adjust the speed of rotation of the CD according to the distance of the track from the center. Hence, there is an upper bound on the number of *zeroes* between successive *ones* that are allowed in the recorded data.

1.2 (d, k)-RLL constraints

The constraints that are implied from the discussion above are called (d, k)-*RLL constraints*. Formally, a binary sequence satisfies a (d, k)-RLL constraint (or a (d, k) constraint), if every run of *zeroes* between successive *ones* has length at least d and at most k. At the beginning and end of the sequence, the runs are only required to be of length at most k.

Example 1 *The sequence* 00100010000010 *is* $(2, 5)$-*constrained.*

Indeed, there are many standard storage devices that use (d, k)-RLL constraints.

Example 2

- *Floppy-disks are* $(1, 7)$ *or* $(2, 7)$-*constrained.*
- *DVDs are* $(2, 10)$-*constrained.*

1.3 Encoding

The user of a storage system may wish to record any binary data on the device, and therefore it has to be changed in order to comply with the constraints. This is called *encoding*. An *encoder* (see Fig. 1.1) is required to transform any binary sequence of length p into a constrained sequence of length n (a *codeword*). Usually $n \geq p$, since not all sequences are valid. The encoding procedure has to be reversible in order to later read the recorded data correctly, hence a *decoder* is required, which converts the constrained sequences of length n back into the original sequences of length p.

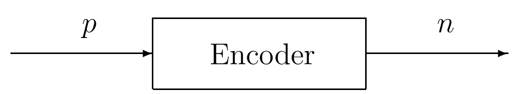


Figure 1.1: An encoder with rate $\frac{p}{n}$.

The ratio $\frac{p}{n}$ is the *rate* of the encoder. A higher rate implies that fewer bits are written per one input bit, which decreases the amount of space needed to record the data. Given a constraint Θ, we are interested in finding the maximal rate possible for an encoder. The *capacity* of a one-dimensional constraint Θ is defined as:

$$C(\Theta) = \lim_{n\to\infty} \frac{\log_2 N(n \mid \Theta)}{n},$$

where $N(n \mid \Theta)$ is the number of codewords of length n that satisfy Θ. Given $N(n \mid \Theta)$ output words of length n, the maximum length p of input sequences can be at most $\log_2 N(n \mid \Theta)$. Hence, the capacity $C(\Theta)$ upper bounds the rate of any encoder for the constraint Θ. Therefore given a constraint Θ, we are interested in finding the capacity $C(\Theta)$.

1.4 One-Dimensional Constrained Coding

Given a one-dimensional (d, k) constraint we construct the following graph (see Fig. 1.2):

- The set of nodes is $\{0, \ldots, k\}$.
- For $0 \le i \le k-1$, there is an edge from node i to node $i+1$, that is labelled by 0.

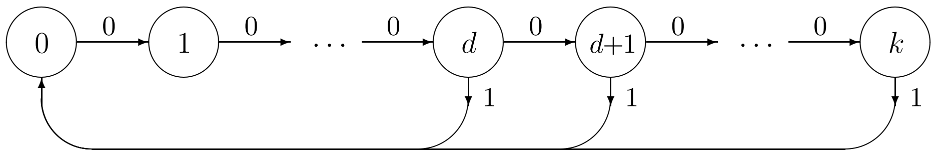


Figure 1.2: A graph for the (d, k)-RLL constraint.

- For $d \leq i \leq k$, there is an edge from node i to node 0, that is labelled by 1.

The graph describes the one-dimensional (d, k) constraint in the following sense: any walk in the graph produces a (d, k) codeword by the sequence of labels on the edges of the walk, and any (d, k) codeword has a corresponding walk.

The capacity of a (d, k) constraint is known to be equal to $\log_2 \lambda$, where λ is the Perron eigenvalue of the adjacency matrix of the graph.

1.5 Two-Dimensional Constrained Coding

Recent developments in optical storage – especially in the area of holographic memory – increase recording density by exploiting the fact that the recording device is a surface. In this new model, the recorded data is regarded as two-dimensional, as opposed to the track-oriented one-dimensional recording paradigm. This new approach, however, necessitates the introduction of new types of constraints which are two-dimensional rather than one-dimensional. While the one-dimensional case has been widely explored, results in the two-dimensional case have been slower to arrive. This is mainly due to the fact that imposing constraints in a few directions makes the coding problem much more difficult. Nevertheless, in the last decade there has been a considerable progress in the study of two-dimensional constraints.

Similarly to the one-dimensional definition, a two-dimensional surface is said to satisfy a (d, k) constraint, if each direction defined by its connectivity model satisfies a one-dimensional (d, k) constraint. The capacity of a two-dimensional constraint Θ is defined by:

$$C(\Theta) = \lim_{n,m\to\infty} \frac{\log_2 N(n, m \mid \Theta)}{nm} ,$$

where $N(n, m \mid \Theta)$ is the number of $n \times m$ arrays satisfying the constraint Θ. An array which satisfies the constraint Θ is called Θ-*constrained* or a Θ *array.*

In general, the algebraic tools used to compute the capacity of one-dimensional constraints, cannot be similarly used in the two-dimensional case. This work focuses on the reduced task of characterizing the region of parameters d, k for which $C(d, k) > 0$. We describe the different connectivity models in the following section.

1.5.1 Connectivity Models

Data should be organized on a two-dimensional surface in some order. This order will be defined by the way in which the data is read. For this purpose four connectivity models are defined. The diamond model, the square model, and the hexagonal model are frequently considered in the literature, e.g., for constrained codes they were considered first by Weeks and Blahut [12]. The triangular model was considered by [19] for constrained codes and for other applications in [6]. Some other papers which consider capacities of constraints in such models are [7, 13, 14, 18, 20].

The first connectivity model is the *diamond* model. In this model, a point $(i, j) \in \mathbb{Z}^2$ has the following four neighbors:

$$\{(i + 1, j), (i - 1, j), (i, j + 1), (i, j - 1)\} .$$

When (i, j) is a boundary point, the neighbor set is reduced to points within the array. In this model the data is organized in the two-dimensional rectangular grid, and it is read horizontally and vertically.

The second model is called the *square* model, in which each point $(i, j) \in \mathbb{Z}^2$ has eight neighbors:

$$\{(i + 1, j), (i - 1, j), (i, j + 1), (i, j - 1),\\ (i + 1, j + 1), (i - 1, j + 1), (i + 1, j - 1), (i - 1, j - 1)\}.$$

In this model the data is organized in the two-dimensional rectangular grid and it is read horizontally, vertically, and in the two diagonal directions.

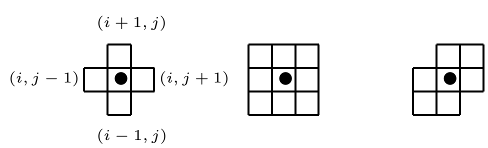
The third model is called the *hexagonal* model. Instead of the rectangular grid we have used up to now, we define the following graph. We start by tiling the plane $\mathbb{R}^2$ with regular hexagons. The vertices of the graph are the center points of the hexagons. These points define the hexagonal lattice [5]. We connect two vertices if and only if their respective hexagons are adjacent. In this way, each vertex has exactly six neighboring vertices.

We will use an isomorphic representation of the model. This representation includes $\mathbb{Z}^2$ as the set of vertices. Each point $(i, j) \in \mathbb{Z}^2$ has the following neighboring vertices:

$$\{(i+1, j), (i-1, j), (i, j+1), (i, j-1), (i-1, j-1), (i+1, j+1)\}.$$

It can be shown that the two models are isomorphic [21]. From now on, by abuse of notation, we will also call the last model – the hexagonal model. In this isomorphic model the data is organized in the two-dimensional rectangular grid and it is read horizontally, vertically, and in one of the diagonals direction called *right diagonal*.

The neighbor sets of the three different models are summarized in Fig. 1.3. A square with a dot is the point (i, j). In all models, rows and columns of the arrays will be indexed in ascending order, bottom to top and left to right.



(a) (b) (c)

Figure 1.3: Neighbors of position (i, j) in the: (a) diamond model, (b) square model, (c) hexagonal model.

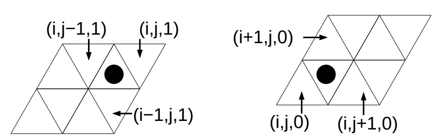
The fourth model is called the *triangular* model. Again, we start by tiling the plane $\mathbb{R}^2$ with regular hexagons. The vertices of the graph are now the vertices of the hexagons, rather than their centers. The edges between the vertices are the sides of the hexagons. Hence, each vertex has exactly three neighboring vertices. If we connect the centers of the hexagons with lines we will obtain a tiling of the $\mathbb{R}^2$ with equilateral triangles. The vertices of the graph are the center points of the equilateral triangles. The set of vertices is also a union of two translates of the hexagonal lattice. Clearly, a point in this model can be represented by a triple $(i, j, s) \in \mathbb{Z}^2 \times \{0, 1\}$. Each point $(i, j, 0) \in \mathbb{Z}^2 \times \{0\}$ has the following neighboring vertices:

$$\{(i, j, 1), (i - 1, j, 1), (i, j - 1, 1)\}.$$

Each point $(i, j, 1) \in \mathbb{Z}^2 \times \{1\}$ has the following neighboring vertices:

$$\{(i, j, 0), (i + 1, j, 0), (i, j + 1, 0)\}.$$

The neighbor sets in this model are illustrated in Fig. 1.4.



(a) (b)

Figure 1.4: Neighbors of positions $(i, j, 0)$ and $(i, j, 1)$ in the triangular model.

As the vertices are two translates of the hexagonal lattice, one can consider the model as having six directions. We will consider it slightly differently. Instead of data stored in the centers of the triangles, the data will occupy the whole area of the triangle. Therefore, in this interpretation there are three directions in this model. Finally, we note that in the triangular model an $n \times m$ array has $2nm$ points. Therefore the definition of the capacity in this model is accordingly adjusted to be:

$$C(\Theta) = \lim_{n,m \to \infty} \frac{\log_2 N(n, m \mid \Theta)}{2nm} .$$

1.5.2 Previous Work

Let $C_\diamond(d,k)$ denote the capacity of the (d,k) two-dimensional constraint in the diamond model. The value of $C_\diamond(1,\infty)$ has been investigated in many works. Calkin and Wilf [3] showed that

$$0.587890... \leq C_\diamond(1,\infty) \leq 0.588339...$$

Weeks and Blahut improved these results in [12], showing that

$$0.58789116177527... \leq C_\diamond(1,\infty) \leq 0.58789149494390...$$

Then they used a numerical convergence-speeding technique called Richardson Extrapolation to estimate that $C_\diamond(1,\infty) \approx 0.587891161775$ and that this approximation is correct up to 12 digits.

For $d \geq 1$, Siegel and Wolf [22], and Halevy, Chen, Roth, Siegel and Wolf [9], bounded $C_\diamond(d,\infty)$ by studying bit-stuffing encoders. Kato and Zeger [13] also showed bounds for these capacities.

For $k \geq 1$, the value of $C_\diamond(0,k)$ was investigated by Talyansky [23], and by Kato and Zeger [13].

For other values of d the capacity of $C_\diamond(d,k)$ is generally unknown. Kato and Zeger [13] characterized the positive capacity region of (d,k) constraints in the diamond model, by proving that $C_\diamond(d,k) > 0$ if and only if $k \geq d+2$.

We are interested in asymmetric constraints in this model, in which there can be different constraints for rows and for columns. $C_\diamond(d_1,k_1,d_2,k_2)$ denotes the capacity of the asymmetric (d_1,k_1,d_2,k_2) constraint in the diamond model, i.e., horizontally the constraint is (d_1,k_1) and vertically the constraint is (d_2,k_2). These constraints were handled in [14].

$C_\boxplus(d,k)$, $C_\hexagon(d,k)$, and $C_\triangle(d,k)$ denote the capacity of the (d,k) constraint in the square model, hexagonal model, and triangular model, respectively. In the hexagonal model, the exact value of $C_\hexagon(1,\infty)$ was given by Baxter [1]. The positive capacity region of hexagonal constraints has been studied by Kukorelly and Zeger in [16, 15].

Finally, the capacity of the hard-triangle constraint (isolated *ones*) was shown in [19] to be bounded by $0.628831217 \leq C_\triangle(1,\infty) \leq 0.634775895$.

1.5.3 Description of the Work

The rest of the chapters are organized as follows. In Chapter 2 we present the known basic techniques to prove zero or positive capacity. We generalize these techniques, so that they could be applied to more complicated cases which we will have in succeeding chapters. In Chapter 3 we examine asymmetric constraints in the diamond model and provide an almost complete solution for the zero/positive capacity region problem. In Chapters 4, and 5 we examine capacities of constraints in the square model and the triangular model, respectively. Discussion and open problems are in Chapter 6.

Chapter 2

Basic Techniques

In this chapter we will survey the known techniques, except for ad-hoc methods, used to prove zero capacity, and those used to prove positive capacity. We will generalize these techniques in a way that will enable them to handle more complicated scenarios. The first lemma which appeared in [14] is an immediate consequence of the definition of the (d, k) constraint.

Lemma 1 *Let Θ be a constraint with minimum runlength d and maximum runlength k in direction Δ. Let $\tilde{\Theta}$ be a constraint with minimum runlength $\tilde{d} \leq d$ and maximum runlength $\tilde{k} \geq k$ in direction Δ, and the same constraints as in Θ in the other directions. Then $C(\Theta) \leq C(\tilde{\Theta})$.*

2.1 Positive Capacity

An $[n \times m, k \times \ell]$ *skeleton tile* is a tile which consists of an $n \times m$ array from which a $k \times \ell$ array was removed from the upper right corner. If $\ell = 1$ we simply have an $[n \times m, k]$ skeleton tile. An example of a $[7 \times 12, 3 \times 5]$ skeleton tile is given in Fig. 2.1.

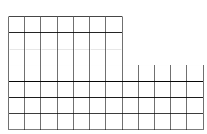


Figure 2.1: A $[7 \times 12, 3 \times 5]$ skeleton tile.

For two points $z_1 = (x_1, y_1)$ and $z_2 = (x_2, y_2)$, $z_1, z_2 \in \mathbb{Z}^2$, let $\mathcal{L}(z_1, z_2) = \{(ix_1 + jx_2, iy_1 + jy_2) \ : \ i, j \in \mathbb{Z}\}$ be *the set of points spanned by* z_1, z_2. This is the lattice defined by z_1 and z_2 (see [5, 8]). Note, that by abuse of notation, the first coordinate is for the row index and the second is for the column index. The following lemma can be easily verified.

Lemma 2 *Let $\mathcal{A}$ be an $[n \times m, k \times \ell]$ skeleton tile. If we place the bottom leftmost point of $\mathcal{A}$ on the points of $\mathcal{L}((n-k, m-\ell), (n, -\ell))$, then a tiling of $\mathbb{Z}^2$ with copies of $\mathcal{A}$ is obtained.*

The tiling obtained by Lemma 2 will be called the *standard tiling*. If $\mathcal{A}$ is an $n \times m$ array (a *skeleton array*) then the standard tiling is obtained by substituting $k = 0$ and $\ell = 0$ in the skeleton tile of lemma 2. Clearly, we can also use a parallelogram instead of a rectangle. A standard tiling can use a few tiles with the same shape and different labels. In this case each one of the tiles can have any one of the labels. The next lemma is a straightforward generalization of similar lemmas for skeleton arrays, given in [7, 14].

Lemma 3 *Let $\mathcal{A}$ and $\mathcal{B}$ be two identical tiles with different labels, and Θ a two-dimensional constraint. If any standard tiling with $\mathcal{A}$ and $\mathcal{B}$ yields a two-dimensional array which is Θ-constrained, then $C(\Theta) > 0$. Moreover, if we can use t identical tiles with different labels $\mathcal{A}_1, \cdots, \mathcal{A}_t$, and the number of points in $\mathcal{A}_i$ is N, then $C(\Theta) \geq \frac{1}{N} log_2 t$.*

2.2 Zero Capacity - The Scanning Method

The most effective method to prove zero capacity was given by Blackburn [2] for specific constraints. However, this method can be formulated to handle general two-dimensional constraints.

Assume we want to show that the capacity of a two-dimensional constraint Θ is zero. We consider an $(n + r_1 + r_2) \times (m + t_1 + t_2)$ array $\mathcal{A}$ which is Θ-constrained, where t_1, t_2, r_1, and r_2 are constants which might depend on the runlength constraints, but do not depend on n and m. Assume further that the labels at positions of the first r_1 rows, the last r_2 rows, the first t_1 columns, and the last t_2 columns, are known. We now scan the other positions of $\mathcal{A}$. We scan the other n rows from bottom to top, and the m positions in a row are scanned from left to right. We assume that all positions in the array are scanned, i.e. we omit arrays in which not all positions can be labelled. If each position is determined by the known labels and the positions which are already scanned, then the capacity of the constraint Θ is zero. We will not give a proof to the claim, since we will prove a much stronger result. This technique will be called *scanning*. The strength of scanning is demonstrated by providing a very short proof to the following theorem by Kato and Zeger [13].

Theorem 1 $C_\diamond(d, d+1) = 0$ *for every* $d \geq 1$.

Proof. Consider an $n \times m$ array $\mathcal{A}$ which is $(d, d+1)$-constrained. We will show that the labels of $\mathcal{A}$ are determined by the labels at positions (i, j), where $0 \leq i \leq d$ or $0 \leq j \leq d-1$ or $j = m-1$.

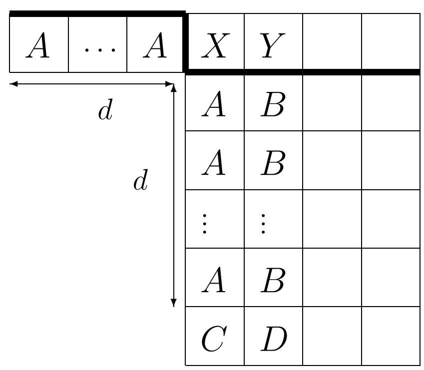


Figure 2.2: Scanning of a $(d, d+1)$ array.

We show that for every $d+1 \leq i$, $d \leq j \leq m-2$, the label of the position marked by X (see Fig. 2.2) is determined by the labels to the left of it and the labels below it. Assume the contrary that X can be a *zero* and can be a *one*. It implies that all the positions marked by A are *zeroes* and either X or Y is a *one*. Since Y can be a *one*, it follows that all positions marked by B are *zeroes*. Since X can be a *zero* it follows by the vertical constraint that C is a *one*. Similarly, since Y can be a *zero*, it follows that D is a *one*, a contradiction to the horizontal constraint. Hence, $C_\diamond(d, d+1) = 0$. ■

We strengthen the technique as follows:

Theorem 2 *Assume the scanning method is applied to a two-dimensional constraint* Θ. *If for the label in each scanned position* (i, j), *one of the following three states holds:*

(s1) The label in position (i, j) *is completely determined;*

(s2) The label can be either zero or one, but with one of these labels the suffix of the row is completely determined;

(s3) The label can be either zero or one, but the prefix of the row before position (i, j) *is a given sequence* $\mathcal{P}(i, j)$*;*

then $C(\Theta) = 0$.

Proof. Assume ρ positions, numbered by $0, 1, \ldots, \rho - 1$, are scanned in a row, as depicted in Fig. 2.3.

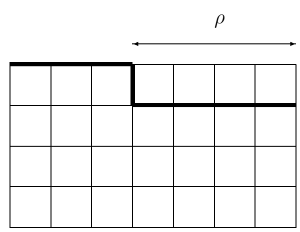


Figure 2.3: Scanning of ρ positions in a row.

Let $\mathcal{T}$ be a directed tree with $\rho + 1$ levels defined as follows. The root of $\mathcal{T}$ (level 0) represents position 0. For $\ell < \rho$, the vertices in level ℓ represent position ℓ. The vertices in level ρ represent all the valid labels of all the ρ positions in the row. A vertex v which is not a leaf has out-degree one or two depending whether the label of the corresponding position is completely determined or not, respectively. The edge which connects a vertex v in level ℓ to vertex u in level $\ell + 1$ is labelled with one of the possible labels of the position represented by v. If the out-degree of v is two then one edge is labelled by a *zero* and one edge is labelled by a *one*. Each vertex v is labelled with the ordered labels of the path from the root to v. Hence, each leaf corresponds to a valid sequence of labels for the ρ positions. We now bound the number of leaves in the tree, which gives an upper bound to the number of possible rows in the scanning.

An example of a tree $\mathcal{T}$ that represents the scanning of ρ positions with no constraints is illustrated in Fig. 2.4. When no constraints are imposed

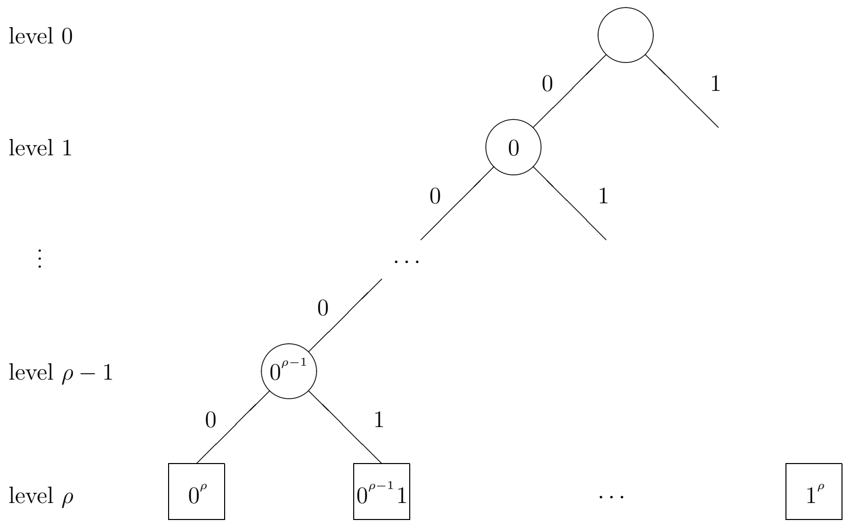


Figure 2.4: The tree $\mathcal{T}$ when no constraints are imposed.

every row is valid, therefore the tree is a complete binary tree. The number of leaves equals the number of all rows of length ρ, which is 2^{ρ}.

We now bound the number of leaves in any tree $\mathcal{T}$. First, we note that the label on a vertex v, which represents position (i, j), represents the labels of the positions before position (i, j). A vertex representing a position in which state (s1) holds has exactly one son. A vertex representing a position in which state (s2) holds has two sons, but one of them is a chain of vertices representing positions in state (s1). Hence, the number of leaves of a subtree whose root is in level ℓ, and does not have vertices which represent positions in state (s3), is at most $\rho - \ell + 1$.

If state (s3) holds in position (i, j) represented by v, then the label on v must be $\mathcal{P}(i, j)$. Therefore, in each level there is at most one vertex which represents a position in which state (s3) holds.

Now, we construct a tree $\mathcal{T}'$ from $\mathcal{T}$ by swapping subtrees of $\mathcal{T}$, with roots on the same level. Clearly, the number of leaves in $\mathcal{T}'$ is equal to the number of leaves in $\mathcal{T}$. $\mathcal{T}'$ will be constructed in a way that all vertices which correspond to positions in which state (s3) holds, are on the same path (see Fig. 2.5). The total number of leaves of $\mathcal{T}'$, which are not on this path, is at most $\sum_{\ell=1}^{\rho}(\rho-\ell+1)=\frac{(\rho+1)\rho}{2}$.

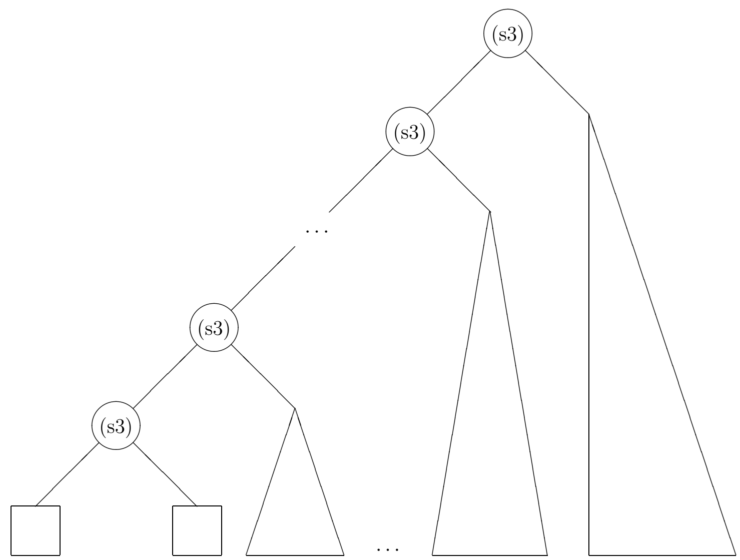


Figure 2.5: The tree $\mathcal{T}'$. Every subtree which does not include vertices on the leftmost path, does not have vertices that represent positions that are in state (s3).

The number of leaves in $\mathcal{T}$ is equal to the number of different labels for a row in the $(n+r_1+r_2)\times(m+t_1+t_2)$ array which is Θ-constrained (for $\rho=m$).

We now have that the total number of possible different labels for an $(n+r_1+r_2)\times(m+t_1+t_2)$ array which is Θ-constrained is at most $(\frac{(m+1)m}{2}+1)^n$, which implies that $C(\Theta)=0$. $\blacksquare$

Chapter 3

Asymmetric Run-length Constrained Channels

The positive capacity region of (d, k) constraints in the diamond model has been determined by Kato and Zeger in [13]: for every $d \geq 1$, $C_\diamond(d, k) > 0$ if and only if $k \geq d + 2$.

In this chapter we investigate asymmetric constraints in the diamond model. The zero/positive region of asymmetric constraints, denoted by $C_\diamond(d_1, k_1, d_2, k_2)$, has been studied by Kato and Zeger in [14]. They have summarized their results in which seven cases remained unsolved:

(u1)	$\mathbf{d_1} = 1,$	$\mathbf{k_1} = 3,$	$\mathbf{d_2} = 2,$	$\mathbf{k_2} = 3.$
(u2)	$2 \leq \mathbf{d_1},$	$\mathbf{k_1} = d_1 + 1,$	$\mathbf{d_2} = d_1,$	$\mathbf{k_2} \leq 2d_2.$
(u3)	$2 \leq \mathbf{d_1},$	$d_1 + 2 \leq \mathbf{k_1} \leq 2d_1,$	$\mathbf{d_2} = d_1,$	$\mathbf{k_2} = d_2 + 1.$
(u4)	$2 \leq \mathbf{d_1},$	$d_1 + 2 \leq \mathbf{k_1} \leq 2d_1,$	$d_1 < \mathbf{d_2} < k_1 - 1,$	$\mathbf{k_2} = d_2 + 1.$
(u5)	$2 \leq \mathbf{d_1},$	$d_1 + 2 \leq \mathbf{k_1} \leq 2d_1,$	$\mathbf{d_2} = k_1 - 1,$	$\mathbf{k_2} \leq 2d_2.$
(u6)	$2 \leq \mathbf{d_1},$	$2d_1 < \mathbf{k_1},$	$d_1 < \mathbf{d_2} < k_1 - 1,$	$\mathbf{k_2} = d_2 + 1.$
(u7)	$2 \leq \mathbf{d_1},$	$2d_1 < \mathbf{k_1},$	$\mathbf{d_2} = k_1 - 1,$	$\mathbf{k_2} \leq 2d_2.$

In this chapter we solve most of these cases.

3.1 Constructions for Proving Positive Capacity

Lemma 4 *$C_\diamond(d, 2d+1, 2d, 2d+1) > 0$ for every $d \geq 1$.*

Proof. Let T_n be the following $(2n-2) \times (2n)$ array. $T_n(1, 2n-2) = 1$ and $T_n(0, n-2) = 1$; if $T_n(i, j) = 1$ then $T_n(i+2, j-1) = 1$ provided that $i + 2 \leq 2n - 3$. All other positions of T_n are *zeroes*. T_4 is illustrated in Fig. 3.1.

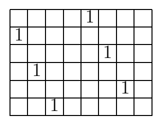


Figure 3.1: The array T_4.

Consider the $[(4d+4) \times (2d+3), 2d+3]$ skeleton tile shown in Fig. 3.2. Let $\mathcal{A}$ and $\mathcal{B}$ be the two $[(4d+4) \times (2d+3), 2d+3]$ tiles obtained from

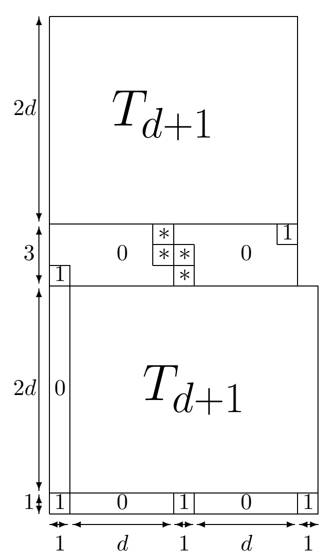


Figure 3.2: The skeleton tile for the $(d, 2d+1, 2d, 2d+1)$ constraint.

this skeleton tile by substituting the two skew tetrominoes shown in Fig. 3.3 instead of the four asterisks. We claim that any standard tiling with the

$$\begin{array}{|c|c|} \cline{1-1} 1 & \multicolumn{1}{c}{} \\ \hline 0 & 1 \\ \hline \multicolumn{1}{c|}{} & 0 \\ \cline{2-2} \end{array} \qquad\qquad \begin{array}{|c|c|} \cline{1-1} 0 & \multicolumn{1}{c}{} \\ \hline 1 & 0 \\ \hline \multicolumn{1}{c|}{} & 1 \\ \cline{2-2} \end{array}$$

Figure 3.3: Two skew tetrominoes for substitution in the skeleton tile.

arrays $\mathcal{A}$ and $\mathcal{B}$ yields a $(d, 2d+1, 2d, 2d+1)$-constrained array. One can easily verify that it is sufficient to prove that the $[(4d+4) \times (2d+3), 2d+3]$ skeleton tile is a $(d, 2d+1, 2d, 2d+1)$ tile, and that the constraint is not violated on rows and columns crossing two different skeleton tiles, on the positions marked in bold in Fig. 3.4.

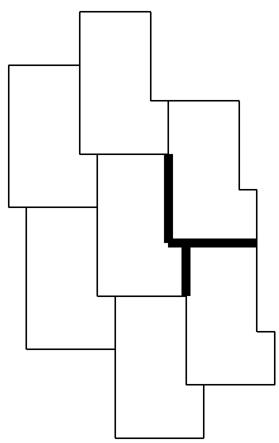


Figure 3.4: Tiling the plane with skeleton tiles.

Now, consider the portions of the rows that cross two skeleton tiles. The scenario is depicted in Fig. 3.5. First note that in Figures 3.2 and 3.5 all the gaps between *ones*, in which at least one of the *ones* is not in T_{d+1} are calculated and written. Therefore, we only have to calculate the gaps between *ones* in the rectangles depicted in Fig. 3.6. In each one of the two figures Fig. 3.6(a),(b), let α be the leftmost copy of T_{d+1}, and β the rightmost copy. In each one of the two figures Fig. 3.6(c),(d), let α be the upper copy of T_{d+1}, and β the lower copy.

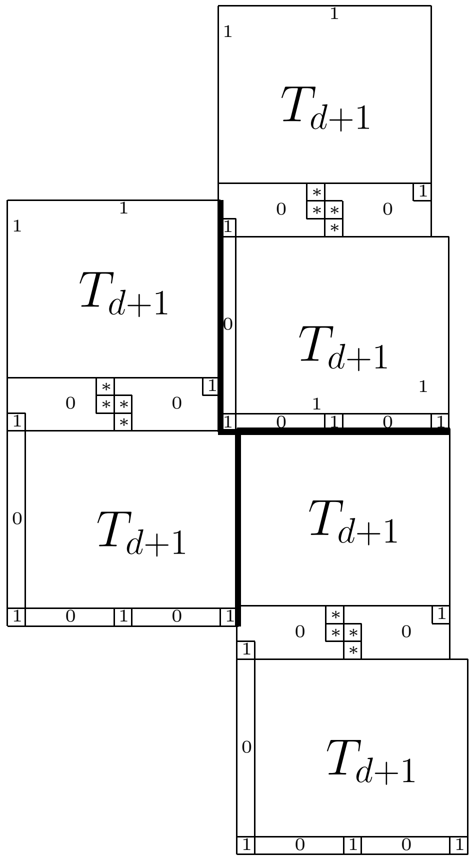


Figure 3.5: Areas crossing two tiles for the $(d, 2d+1, 2d, 2d+1)$ constraint.

1. We start with the *ones* of Fig. 3.6(a). We calculate the gaps between *ones* of α and β. α and β are separated by one column, and since the width of T_{d+1} is $2d+2$, and β is shifted down by two rows, the gaps between *ones* are $2d+1$.

2. In Fig. 3.6(b), the gaps between *ones* of α and β are also $2d+1$, since the width of T_{d+1} is $2d+2$.

3. In Fig. 3.6(c), α and β are separated by three rows, and since the height of T_{d+1} is $2d$, and β is shifted to the right by one column, the vertical gaps between *ones* are $2d$.

4. In Fig. 3.6(d), the vertical gaps between *ones* are also $2d$, since the height of T_{d+1} is $2d$, and α and β are separated by one row.

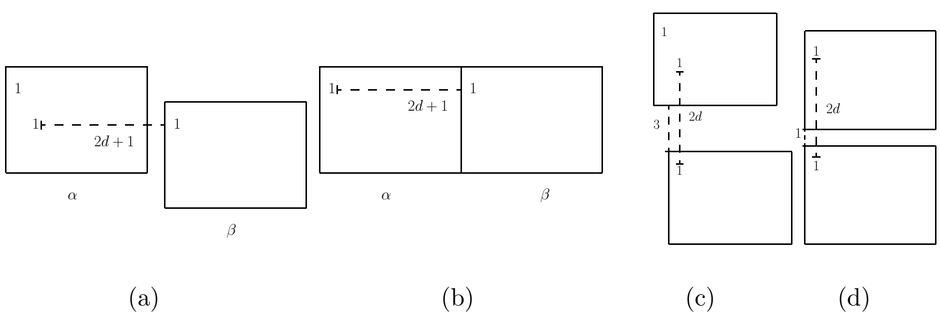


Figure 3.6: Relative locations of T_{d+1} arrays.

Hence, any standard tiling with $\mathcal{A}$ and $\mathcal{B}$ is a $(d, 2d+1, 2d, 2d+1)$ array. Therefore, by Lemma 3 we have $C_\diamond(d, 2d+1, 2d, 2d+1) \geq \frac{1}{(4d+4)(2d+3)-(2d+3)} = \frac{1}{8d^2+18d+9}$. ■

Lemma 5 $C_\diamond(d, 2d+2, 2d+1, 2d+2) > 0$ *for every* $d \geq 1$.

Proof. Consider the $(4d+5) \times (2d+3)$ skeleton array of Fig. 3.7. Let $\mathcal{A}$ and $\mathcal{B}$ be the two $(4d+5) \times (2d+3)$ arrays, obtained from the skeleton array by substituting a *one* instead of one of the asterisks and a *zero* instead of the other.

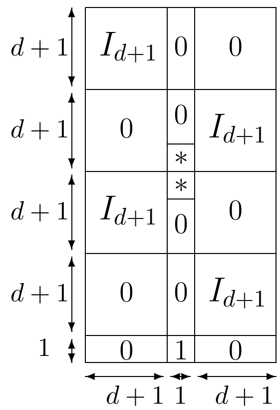


Figure 3.7: A skeleton array for the $(d, 2d+2, 2d+1, 2d+2)$ constraint.

Consider any standard tiling of the plane with $\mathcal{A}$ and $\mathcal{B}$. Every row that does not contain an asterisk, has the pattern $(10^{2d+2})^\infty$, which is $(d, 2d+2)$-constrained. The rows that contain asterisks have the pattern $(0^{d+1} * 0^d 1)^\infty$ or $(10^d * 0^{d+1})^\infty$, which will be valid in any substitution of *zeros* and *ones* instead of the asterisks.

Similarly, every column that does not contain an asterisk, has the pattern $(10^{2d+2}10^{2d+1})^\infty$, which is $(2d+1, 2d+2)$-constrained. The columns that contain asterisks have the pattern $(10^{2d+1} * *0^{2d+1})^\infty$, which will be valid in any substitution of one *zero* and one *one* instead of each consecutive asterisks.

Hence, any standard tiling of the plane with $\mathcal{A}$ and $\mathcal{B}$ yields a two-dimensional $(d, 2d+2, 2d+1, 2d+2)$-constrained array. Therefore, by Lemma 3 $C_\diamond(d, 2d+2, 2d+1, 2d+2) \geq \frac{1}{(4d+5)(2d+3)} = \frac{1}{8d^2+22d+15}$. ∎

Lemma 6 *If* $d_1 \geq 1$, $k_1 > 2d_1$, $d_2 = k_1 - 1$ *and* $k_1 \leq k_2 \leq 2d_2$ *then* $C_\diamond(d_1, k_1, d_2, k_2) > 0$.

Proof. Assume $d_1 \geq 1$, $k_1 = 2d_1 + t$, $t > 0$, $d_2 = k_1 - 1$, and $k_2 = k_1$. We distinguish between two cases:
Case 1: $t = 2r + 1$, $r \geq 0$.
By Lemma 4 we have $C_\diamond(d_1 + r, 2d_1 + 2r + 1, 2d_1 + 2r, 2d_1 + 2r + 1) > 0$. Therefore, by Lemma 1 we have $C_\diamond(d_1, 2d_1+2r+1, 2d_1+2r, 2d_1+2r+1) > 0$.
Case 2: $t = 2r + 2$, $r \geq 0$.
By Lemma 5 we have $C_\diamond(d_1 + r, 2d_1 + 2r + 2, 2d_1 + 2r + 1, 2d_1 + 2r + 2) > 0$. Therefore, Lemma 1 implies $C_\diamond(d_1, 2d_1+2r+2, 2d_1+2r+1, 2d_1+2r+2) > 0$.

Hence, $C_\diamond(d_1, 2d_1 + t, 2d_1 + t - 1, 2d_1 + t) > 0$ and thus by Lemma 1 we have that if $d_1 \geq 1$, $k_1 > 2d_1$, $d_2 = k_1 - 1$ and $k_1 \leq k_2 \leq 2d_2$ then $C_\diamond(d_1, k_1, d_2, k_2) > 0$. ■

Lemma 7 *If $d \geq 2$ and $d-1 \geq r \geq 1$, then $C_\diamond(d, 2d+1, d+r, d+r+1) > 0$.*

Proof. We begin by recursively defining a $(d + r - 1) \times d$ array, $H_{d,r}$, as follows. For $\rho \geq 1$ let,

$$H_{\delta,2\rho} = \begin{pmatrix} & & & 0 \\ & H_{\delta-1,2\rho-1} & & \vdots \\ & & & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \quad H_{\delta,2\rho+1} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ 0 & & & \\ \vdots & & H_{\delta-1,2\rho} & \\ 0 & & & \end{pmatrix},$$

where $H_{\delta,1} = I_\delta$. $H_{8,6}$ is illustrated in Fig. 3.8.

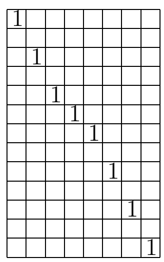


Figure 3.8: The array $H_{8,6}$.

Next, we define the $(d + r - 1) \times d$ array $H'_{d,r}$, by rotation of $H_{d,r}$ by 180°. Note that $H_{d,r} = H'_{d,r}$ if and only if r is odd. Also, in the "center" of $H_{d,r}$ ($H'_{d,r}$) there is the identity matrix I_{d-r+1}. This part of the array will be called *center*.

Consider the $[(2d+2r+4) \times (3d+2), 2d+2r+1]$ skeleton tile of Fig. 3.9. Let $\mathcal{A}$ and $\mathcal{B}$ be the two $[(2d + 2r + 4) \times (3d + 2), 2d + 2r + 1]$ tiles obtained from the skeleton tile by substituting the two skew tetrominoes of Fig. 3.3 instead of the four asterisks.

As in the proof of Lemma 4 we have to prove that any standard tiling with $\mathcal{A}$ and $\mathcal{B}$ is a $(d, 2d+1, d+r, d+r+1)$-constrained array. One can easily

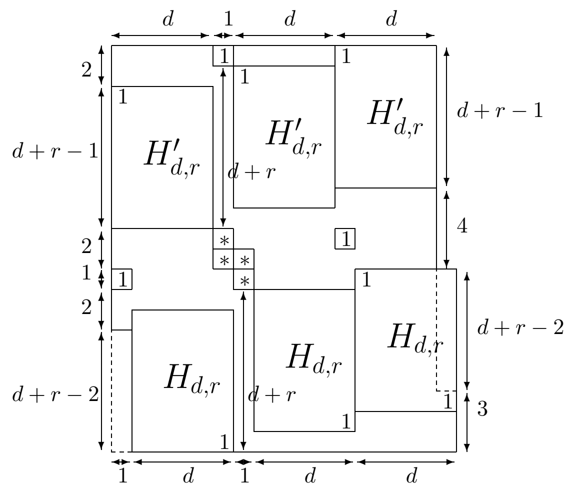


Figure 3.9: The skeleton tile for $(d, 2d+1, d+r, d+r+1)$ constraint.

verify that it is sufficient to prove that the $[(2d+2r+4)\times(3d+2), 2d+2r+1]$ skeleton tiles are $(d, 2d+1, d+r, d+r+1)$ tiles, and that the constraint is not violated on rows and columns crossing two different skeleton tiles on the positions marked in bold in Fig. 3.4. The scenario is depicted in Fig. 3.10.

First note that rotating the plane by 180°, around any of the tetrominoes (while the tetrominoes are still labelled with the asterisks) leaves the plane with exactly the same labels. Note also that in Figures 3.9 and 3.10 all the gaps between *ones*, in which at least one of the *ones* is not in $H_{d,r}$ or $H'_{d,r}$ are calculated and written. Therefore, we only have to calculate the gaps between *ones* in the rectangles depicted in Fig. 3.11. In each one of the three figures (Fig. 3.11(a),(b),(c)), let α be the leftmost copy of $H_{d,r}$, β the middle copy, and γ the rightmost copy of $H_{d,r}$.

1. We start with the *ones* of Fig. 3.11(a). We calculate the gaps between *ones*, where one of the *ones* is in α. If the second *one* is in β then both *ones* belong to the center of $H_{d,r}$, and hence the gap between them is d. If the corresponding row in β consists only of *zeroes*, then the corresponding row in γ contains a *one* as depicted in Fig. 3.11(a). The gap between these two *ones* is $2d$. The gaps between *ones* of β and γ are the same as the gaps between the *ones* of α and β.

2. The gaps between the *ones* of α and β in Fig. 3.11(b) are the same

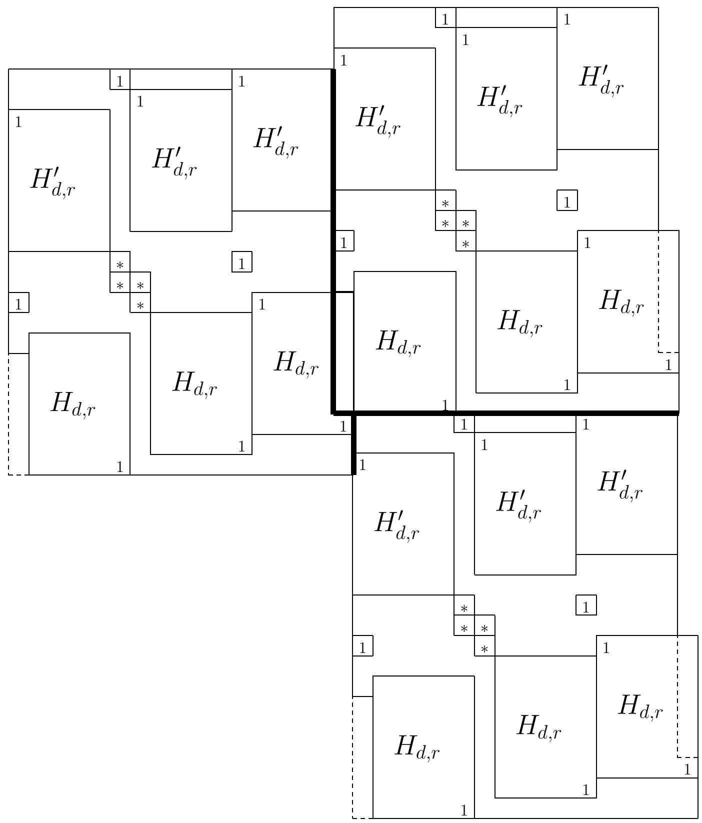


Figure 3.10: Areas crossing two tiles for the $(d, 2d+1, d+r, d+r+1)$ constraint.

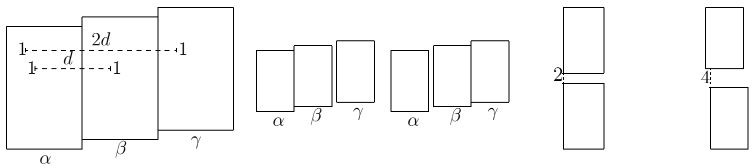


Figure 3.11: Relative locations of $H_{d,r}$ arrays.

as the gaps between the *ones* of α and β in Fig. 3.11(a). The gaps between the *ones* of α and γ in Fig. 3.11(b), where the corresponding row of β has *zeroes* are greater by one than the gaps between the *ones* of α and γ in Fig. 3.11(a), and hence these gaps have length $2d+1$. Similarly, the gaps between β and γ are $d+1$.

3. The gaps between the *ones* of α and β in Fig. 3.11(c) are greater by one than gaps between the *ones* of α and β in Fig. 3.11(a), and hence these gaps have length $d+1$. The gaps between the *ones* of α and γ in Fig. 3.11(c), where the corresponding row of β has *zeroes* are the same as the gaps between the *ones* of α and γ in Fig. 3.11(a). Similarly, the gaps between β and γ are $d+1$.

4. Since the height of $H_{d,r}$ is $d+r-1$, it follows that the vertical gaps between *ones* in Fig. 3.11(d) is $d+r$ if r is odd. If r is even, then the gap between two *ones* is $d+r$ if at least one of them is not in the center of its shape, and $d+r+1$ between the other *ones*.

5. The vertical gaps between *ones* in Fig. 3.11(e) is $d+r$ if r is even. If r is odd, then the gap between two *ones* is $d+r$ if at least one of them is not in the center of its shape, and $d+r+1$ between the other *ones*.

This completes the proof that any standard tiling with $\mathcal{A}$ and $\mathcal{B}$ is a $(d, 2d+1, d+r, d+r+1)$-constrained array. Therefore, by Lemma 3 we have $C_\diamond(d, 2d+1, d+r, d+r+1) \geq \frac{1}{(2d+2r+4)(3d+2)-(2d+2r+1)} = \frac{1}{6d^2+6dr+14d+2r+7}$. ■

Lemma 8 *If* $d_1 \geq 2$, $k_1 > 2d_1$, $d_1 < d_2 < k_1 - 1$ *and* $k_2 = d_2 + 1$, *then* $C_\diamond(d_1, k_1, d_2, k_2) > 0$.

Proof. Assume $d_1 \geq 2$, $k_1 > 2d_1$, $d_1 < d_2 < k_1 - 1$, and $k_2 = d_2 + 1$. We distinguish between two cases:
Case 1: $d_1 < d_2 < 2d_1$.
By Lemma 7 we have $C_\diamond(d_1, 2d_1 + 1, d_2, d_2 + 1) > 0$. Since $k_1 \geq 2d_1 + 1$, by Lemma 1 we have $C_\diamond(d_1, k_1, d_2, d_2 + 1) > 0$.
Case 2: $2d_1 \leq d_2 < k_1 - 1$.
Since $d_2 \geq 2d_1$ then by Lemma 6 we have $C_\diamond(d_1, d_2+1, d_2, d_2+1) > 0$. In this case $k_1 > d_2 + 1$, and therefore Lemma 1 implies $C_\diamond(d_1, k_1, d_2, d_2 + 1) > 0$. ■

3.2 Proving Zero Capacity

Proposition 1 *If* $d_1 \geq 2$, $k_1 \leq 2d_1$, $d_1 \leq d_2 \leq k_1 - 1$, *and* $k_2 = d_2 + 1$ *then* $C_\diamond(d_1, k_1, d_2, k_2) = 0$.

Proof. Consider an array $\mathcal{A}$ which is (d_1, k_1, d_2, k_2)-constrained. We will show that the label X at position (i, j) is determined by the d_1 labels to the left of it, and the labels of the $(d_2+1) \times (d_1+1)$ array below it (see Fig. 3.12).

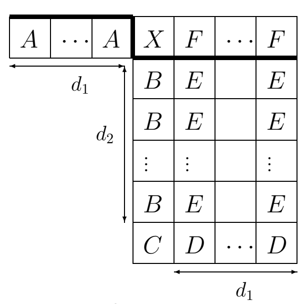


Figure 3.12: Labels of the array in Proposition 1.

Assume the contrary that X can be labelled by a *zero* and can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*. If any of them was labelled with a *one*, it would imply that X is a *zero*, in order to avoid a pattern which violates the horizontal constraint. The same argument vertically implies that all the positions marked by B are *zeroes*.

If the position marked by C is a *zero*, then the positions marked by B or C form a run of $d_2 + 1 = k_2$ *zeroes*, which implies that X is a *one*. Hence, C is a *one* and all the positions marked by D are *zeroes*, in order to satisfy the horizontal constraint.

Consider the d_2 positions marked by E in one of the corresponding d_1 columns. If all these d_2 positions are *zeroes*, then the position marked by F in the same column should be labelled with a *one* by the vertical constraint, and X is a *zero* by the horizontal constraint. Therefore, in each column with positions marked by E, one of these positions is a *one* which implies that all the positions marked by F are labelled by *zeroes*. Since all positions marked by A are also *zeroes*, it follows that X is a *one*, which contradicts our assumption.

Thus, by Theorem 2 we have $C_\diamond(d_1, k_1, d_2, k_2) = 0$. ■

3.3 Summary of Results for the Diamond Model

The results in this chapter produce solutions to most of the seven unsolved cases of [14]:

(u1) is solved in Lemma 4,
(u2), (u3), and (u4) in Proposition 1,
(u6) in Lemma 8,
(u7) in Lemma 6,
and (u5) was solved when $k_2 = d_2 + 1$, in Proposition 1.

The only case which remains unsolved is when $2 \leq d_1$, $d_1 + 2 \leq k_1 \leq 2d_1$, $d_2 = k_1 - 1$, $d_2 + 2 \leq k_2 \leq 2d_2$.

Chapter 4

The Square Model

In this model the data is organized in the two-dimensional rectangular grid, and is read horizontally, vertically, and in the two diagonal directions.

4.1 Proving Zero Capacity

Recall that the hexagonal model can be represented as a rectangular grid with 3 directions. Therefore, any (d, k)-constrained array in the square model is also a (d, k)-constrained array in the hexagonal model, which implies the following lemma:

Lemma 9 *For every d, k, $C_{\boxplus}(d, k) \leq C_{\hexagon}(d, k)$.*

In particular, Lemma 9 implies that if $C_{\hexagon}(d, k) = 0$ then $C_{\boxplus}(d, k) = 0$. We will use this in proving zero capacity for some constraints in the square model.

Theorem 3 *$C_{\boxplus}(d, d + 3) = 0$ for every $d \geq 1$.*

Proof. We begin by proving for $d = 1$. Kukorelly and Zeger [16] showed that $C_{\hexagon}(d, d + 2) = 0$ for every $d \geq 1$. In particular, $C_{\hexagon}(2, 4) = 0$ and therefore by Lemma 9 we have $C_{\boxplus}(2, 4) = 0$. Hence, if $C_{\boxplus}(1, 4) > 0$ then there exists a $(1, 4)$ array that has a run of exactly 1 *zero* (see Fig. 4.1). Each *one* implies *zeroes* in each of its 8 neighbors, therefore all the positions marked by A are *zeros*. This creates a run of 5 *zeros* horizontally, which is a contradiction. Hence, $C_{\boxplus}(1, 4) = 0$.

A	A	A	A	A
A	1	0	1	A
A	A	A	A	A

Figure 4.1: Proving $C_{\boxplus}(1,4) = 0$.

For $d = 2$, the proof is similar. Again we have by Lemma 9 that $C_{\boxplus}(3,5) = 0$, hence if $C_{\boxplus}(2,5) > 0$ then there exists a $(2,5)$ array that has a run of exactly 2 *zeroes* (see Fig. 4.2). Each *one* implies *zeroes* in each of its 8

A	A	A	A	A	A
A	1	0	0	1	A
A	A	A	A	A	A

Figure 4.2: Proving $C_{\boxplus}(2,5) = 0$.

neighbors, therefore all the positions marked by A are *zeros*. This creates a run of 6 *zeros* horizontally, which is a contradiction. Hence, $C_{\boxplus}(2,5) = 0$.

In [15], Kukorelly and Zeger prove that $C_{\hexagon}(d, d+3) = 0$ for $d = 3,4,5$. Therefore by Lemma 9, we have that $C_{\boxplus}(d, d+3) = 0$ for $d = 3,4,5$.

The rest of the proof assumes $d \geq 6$. Consider an array $\mathcal{A}$ which is $(d, d+3)$-constrained. We will show that the label X at position (i,j) is determined by the labels to the left of it and labels below it (see Figure 4.3). Assume the contrary, i.e. that X can be labelled by a *zero* and can be labelled

	d					
A	$\cdots$	A	X	Y_1	Y_2	Y_3

Figure 4.3: Scanning of a $(d, d+3)$ array.

by a *one*. It implies that all the positions marked by A are *zeroes* and either X or one of the three positions to the right of X is a *one*. Therefore, at least one of the following three cases must be valid.

Case 1: X can be a *one* and Y_1 can be a *one* (see Fig. 4.4). Clearly, all

			B	B
A	$\cdots$	A	X	Y_1
			B	B
		d	$\vdots$	$\vdots$
			B	B
			C_1	D
			C_2	D

Figure 4.4: Case 1 of Theorem 3.

positions marked by B are *zeroes*. X can be a *zero*, and therefore by the vertical constraint either C_1 or C_2 is a *one*. This implies that both positions marked by D are *zeroes*, which will create a vertical run of $d+4$ *zeroes* when Y_1 will be a *zero*, which is a contradiction.

Case 2: X can be a *one* and Y_2 can be a *one* (see Fig. 4.5). As in case

			F_2						F_1					
				E	D	E	E	E	C_1					
					E	B	E	B						
						C_2	B	D						
A		$\cdots$			A	X	B	Y_2						
				E	B	B		B	B					
				B		B		B	E	B				
		$\iddots$												
		$\iddots$			d	$\vdots$		$\vdots$			$\ddots$	$\ddots$		
E												E		
B						B		B						B

Figure 4.5: Case 2 of Theorem 3.

1, the positions marked by B are *zeroes*. Also, if X will be a *zero* then the positions marked by C_1 and C_2 will be *ones*, and if Y_2 will be a *zero* then the positions marked by D will be *ones*. Therefore, the positions marked by E must be *zeroes*. This implies, by the diagonals constraints, that if C_2 will be a *zero* then both F_1 and F_2 will be *ones*, a contradiction to the horizontal constraint since the gap between them is of length 5.

Case 3: X can be a *one* and Y_3 can be a *one* (see Fig. 4.6). Clearly, all

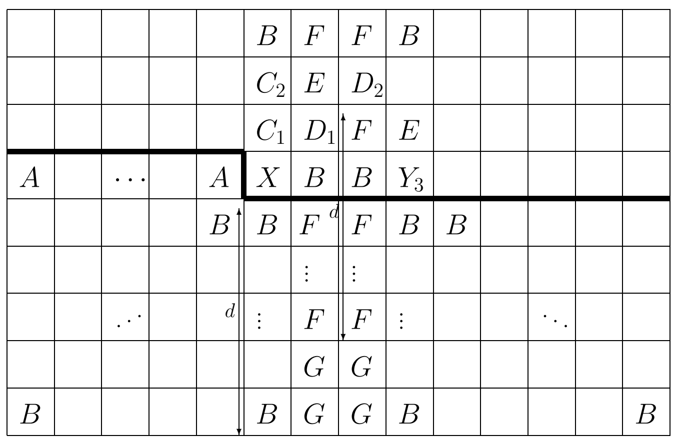


Figure 4.6: Case 3 of Theorem 3.

positions marked by B are *zeroes*. If X will be a *zero*, then by the vertical constraint either C_1 or C_2 will be a *one*, and by the right diagonal constraint either D_1 or D_2 will be a *one*, which implies that C_1 and D_2 will be *ones*. Similarly, if Y_3 will be a *zero*, then by the vertical constraint and the left diagonal constraint, the positions marked by E will be *ones*. This implies that D_1 and all positions marked by F must be *zeroes*. Hence, similarly to case 1, in order to avoid a vertical run of $d+4$ *zeroes*, two of the four positions marked by G must be *ones*, which is clearly impossible.

By Theorem 2, this completes the proof that $C_{\boxplus}(d, d+3) = 0$, for every $d \geq 1$. ■

4.2 Summary of Results for the Square Model

The following positive capacity results in the square model appear in [4]:

- $C_{\boxplus}(d, d+6) > 0$, for every $d \equiv 1, 21 (mod\ 30)$
- $C_{\boxplus}(d, d+8) > 0$, for every $d \equiv 2, 30 (mod\ 42)$
- $C_{\boxplus}(d, d+16) > 0$, for every $d \equiv 2, 46 (mod\ 66)$
- $C_{\boxplus}(d, d+18) > 0$, for every $d \equiv 3, 55 (mod\ 78)$

Theorem 3 proves that $C_{\boxplus}(d, d+3) = 0$, for every $d \geq 1$. Thus, there is still a gap between the known zero and positive capacity regions.

Chapter 5

The Triangular Model

In this chapter we investigate the positive capacity region of the triangular model.

Let $\mathcal{A}$ be an $n \times n$ triangular array. We say that $\mathcal{A}$ has n rows, n right columns, and n left columns. $\mathcal{A}(i, j, s)$ belongs to row i, right column j, and left column $[i + j + s]_n$ (see Fig. 5.1).

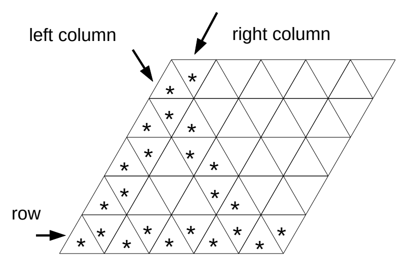


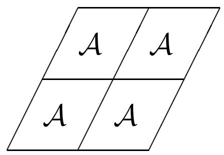
Figure 5.1: A triangular array

5.1 A Construction for Proving Positive Capacity

An $n \times n$ triangular array is called a *doubly periodic non-attacking triangle queens array* if each row, right column, and left column has exactly one *one*.

Lemma 10 *An $n \times n$ doubly periodic non-attacking triangle queens array exists if and only if a $(2n-1, 2n-1)$ triangular array exists.*

Proof. Let $\mathcal{A}$ be an $n \times n$ doubly periodic non-attacking triangle queens array. Consider the following $2n \times 2n$ triangular array:

$\mathcal{B} =$ 

Clearly, each row (right column) of $\mathcal{B}$ has two *ones* separated by $2n-1$ *zeroes*. Now, consider the bottom right and the upper left copies of $\mathcal{A}$. Each left column which has a *one* in these arrays has two *ones* on the corresponding left column of $\mathcal{B}$. They are separated by $2n-1$ *zeroes* as the other two copies of $\mathcal{A}$ cannot have a *one* on the same left column of $\mathcal{B}$.

Note that any run of $2n$ symbols in the tiling has a representation in $\mathcal{B}$. Therefore, *ones* in each row of the tiling are separated by $2n-1$ *zeroes*, and the same is true for right and left columns. ■

Lemma 11 *If $\mathcal{A}$ is an $n \times n$ (d, d) triangular array then any exchanges of copies of the patterns shown in Fig. 5.2 in disjoint positions of $\mathcal{A}$ will result in a $(d-2, d+2)$ array.*

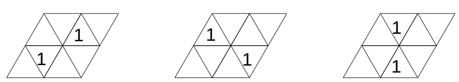


Figure 5.2: Three 2×2 exchangeable triangular arrays.

Proof. The *ones* in all three triangular arrays occupy the same rows, and right and left columns. In each direction, the difference between the arrays, is a change of at most 2 positions for the label *one*.

In a (d, d) array, any two adjacent copies of the patterns above must be identical. Therefore, by exchanges of copies of the above patterns in disjoint positions of a (d,d) array, the length of any given run of *zeroes* may increase or decrease by at most 2. This results in a $(d-2,d+2)$ array. ■

We now make use of Lemmas 10 and 11, to construct tiles that imply positive capacity of some constraints in the triangular model.

Lemma 12 *If* $d \equiv 1 \ (mod\ 4)$ *then* $C_{\triangle}(d,d+4) \geq \frac{1}{2(d+3)} log_2 3$.

Proof. For even n we construct an $n \times n$ doubly periodic non-attacking triangle queens array $\mathcal{T}_n$, where $\mathcal{T}_n(i,i,s) = 1$ if $s \not\equiv i \ (mod\ 2)$, for every $0 \leq i \leq n-1$ ($\mathcal{T}_6$ is illustrated in Fig. 5.3). By Lemma 10, the standard

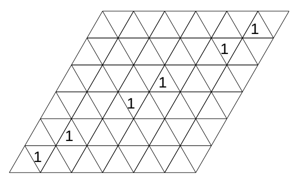


Figure 5.3: The triangular array $\mathcal{T}_6$.

tiling with $\mathcal{T}_n$ is a $(2n-1,2n-1)$ array. By Lemma 11, any exchanges of copies of the pattern shown in Fig. 5.2 in disjoint positions of $\mathcal{A}$ will result in a $(2n-3,2n+1)$ array. The total number of different $(2n-3,2n+1)$ arrays used in the tiling is $3^{\frac{n}{2}}$. Hence, by Lemma 3 we have that $C_{\triangle}(2n-3,2n+1) \geq \frac{1}{4n} log_2 3$. ■

The following lemma shows that when $d \equiv 3 \ (mod\ 4)$, a similar construction to the one above does not exist.

Lemma 13 *If n is odd then there is no $n \times n$ doubly periodic non-attacking triangle queens array which contains an appearance of any of the patterns shown in Fig. 5.2 .*

Proof. Assume that n is odd and an $n \times n$ doubly periodic non-attacking triangle queens array $\mathcal{A}$ exists. We write $\mathcal{A}$ as a sequence $a_0, a_1, \cdots, a_{n-1}$, where $a_i = (j_i, s_i)$ if $\mathcal{A}(i, j_i, s_i) = 1$. Since $\mathcal{A}$ is a doubly periodic non-attacking triangle queens array, it follows that for every $0 \leq r < \ell \leq n-1$, we have $j_r \neq j_\ell$ because there cannot be 2 *ones* in the same right column, and $j_r + r + s_r \not\equiv j_\ell + \ell + s_\ell (mod\ n)$ because there cannot be 2 *ones* in the same left column. Therefore, $j_0, j_1, \cdots, j_{n-1}$ and $[j_0 + 0 + s_0]_n, [j_1 + 1 + s_1]_n, \cdots, [j_{n-1} + (n-1) + s_{n-1}]_n$ are permutations of $0, 1, \cdots, n-1$. For any given permutation $p_0, p_1, \cdots, p_{n-1}$ of $0, 1, \cdots, n-1$ we have

$$\sum_{i=0}^{n-1} p_i = \frac{(n-1)n}{2} \equiv 0(mod\ n),$$

since n is odd. Therefore,

$$\sum_{i=0}^{n-1} s_i = \sum_{i=0}^{n-1} (j_i + i + s_i) - \sum_{i=0}^{n-1} j_i - \sum_{i=0}^{n-1} i \equiv 0(mod\ n).$$

Hence, either $s_i = 0$ for each $0 \le i \le n-1$, or $s_i = 1$ for each $0 \le i \le n-1$, implying that all the positions that are *ones* have the same orientation. Thus, there is no doubly periodic $n \times n$ non-attacking triangle queens array which contains an appearance of any 2×2 array shown in Fig. 5.2. ∎

5.2 Proving Zero Capacity

In this section we prove that $C_\triangle(d, d+3) = 0$ for every $d \ge 5$. For technical reasons, the proof is divided into two parts, one proof for even values of d, and another for odd ones.

The following lemma will be used in Lemma 15, when proving that $C_\triangle(d, d+3) > 0$ for even $d \ge 6$.

Lemma 14 *Let $d \ge 6$ be an even integer, $h = \frac{d+6}{2}$, and let $\mathcal{A}$ be an infinite $(d, d+3)$ array. If $\mathcal{A}$ contains an $r \times h$ sub-array $\mathcal{B}$ whose first two rows form the pattern PEven (see Fig. 5.4), then the first two and the last two right columns of $\mathcal{B}$ are substrings of $(10^{d+1})^\infty$.*

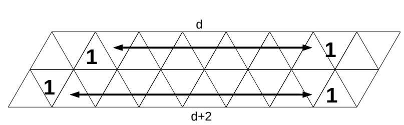


Figure 5.4: The pattern PEven.

Proof. Let $\mathcal{C}$ be an $r \times (h+1)$ sub-array of $\mathcal{A}$ with the pattern PEven as depicted in Fig. 5.5. Clearly the positions marked by A are *zeroes*. By the left column constraint either B_1 or B_2 will be a *one* and hence all positions marked by C are *zeroes*. Assume the position marked by D is a *one*. Then,

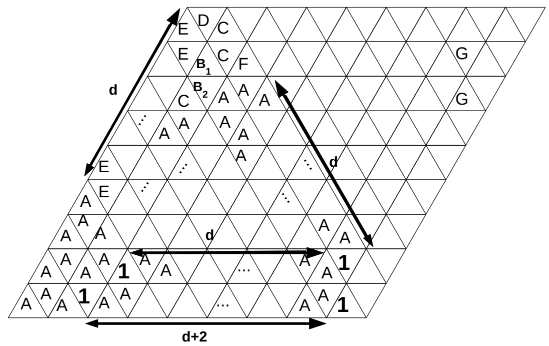


Figure 5.5: Labels implied by the pattern PEven.

all positions marked by E will be *zeroes* which will create a run of $d+7$ zeroes in the right column, a contradiction. Hence, D is a *zero*, F is a *one*, B_1 is a *zero*, and B_2 is a *one*.

The four *ones* in the left columns of B_2 and F form the pattern PEven and hence by the same arguments the two positions marked by G are *ones*. The positions marked by B_2, F, and G form again the pattern PEven. The claim of the lemma is proved now by induction. ■

Lemma 15 *If $d \geq 6$ is even then $C_{\triangle}(d, d+3) = 0$.*

Proof. We use the scanning technique again, and show that one of the three states of Theorem 2 occurs in each scanned position. Assume we have to label the next scanned position marked by X. We have to distinguish between two different types of orientations of the position as depicted in Fig. 5.6.

Figure 5.6: The possible orientations of a scanned position.

Case 1: Assume that X, as depicted in Figure 5.7 (to simplify the picture, the array is drawn in a different orientation), is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, and either B_1 or B_2 is a *one*. Therefore, the positions marked by C are *zeroes* and at least one of the following three cases must be valid.

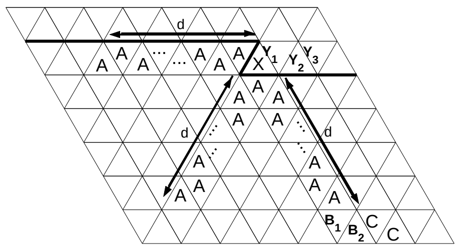


Figure 5.7: Case 1 of Lemma 15.

Case 1a: X can be a *one* and Y_1 can be a *one* (see Fig. 5.8). Clearly, all positions marked by D are *zeroes*. Y_1 can be a *zero*, therefore E is a *one*, and hence B_2 is a *zero* and B_1 is a *one*. Therefore, the seven positions marked by F are *zeroes*.

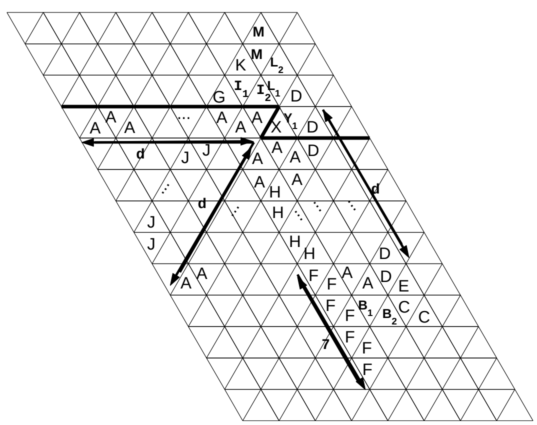


Figure 5.8: Case 1a of Lemma 15.

Assume G is a *one*. Then the d positions below it (ending with the $d-5$ positions marked by H) are *zeroes*, creating a run of $d+7$ *zeroes*, a contradiction. Hence, G is a *zero*.

If X will be a *zero* then either I_1 or I_2 will be a *one*. Assume I_1 will be a *one*. Then, all the positions marked by J are *zeroes*. If X will be a *one* then I_1 and K will be *zeroes*, Y_1 will be a *zero*, either L_1 or L_2 will be a *one* and the two positions marked by M will be labelled by *zeroes*. Therefore, there is a run of $d+4$ zeroes is the right column of K, a contradiction. Hence, if X will be a *zero* then I_1 will be a *zero*, I_2 and Y_1 will be *ones*. E, B_1, I_2 and Y_1 will form the pattern Peven, and hence by Lemma 14 the suffix of the current row is completely determined, and we are in state (s2).

Case 1b: X can be a *one* and Y_2 can be a *one* (see Fig. 5.9). If Y_2 will be a *one* then all positions marked by D are *zeroes*. If X will be a *one* then Y_2 will be a *zero*, and hence there is a run of $d+5$ zeroes in the left column of Y_2, a contradiction. Thus, Y_2 cannot be a *one*.

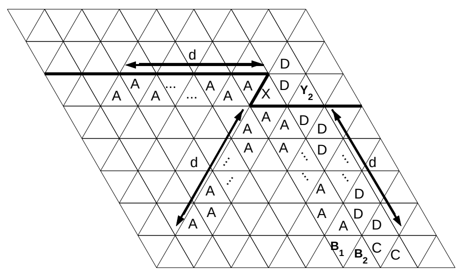


Figure 5.9: Case 1b of Lemma 15.

Case 1c: X can be a *one* and Y_3 can be a *one* (see Fig. 5.10). Clearly, all

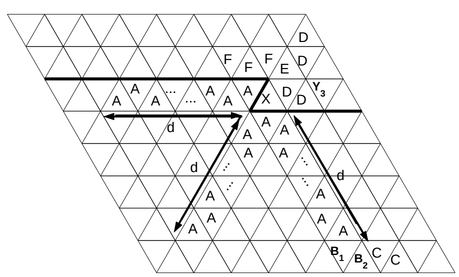


Figure 5.10: Case 1c of Lemma 15.

positions marked by D are *zeroes*. If X will be a *zero*, then Y_3 will be a *one* and the position marked by E will be a *one*, and hence all positions marked by F are *zeroes*. Therefore, there is a run of length $d+4$ in left column of X, a contradiction. Thus, X cannot be a *zero*.

Case 2: Assume that X, as depicted in Fig. 5.11, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, and at least one of the following three cases must be valid.

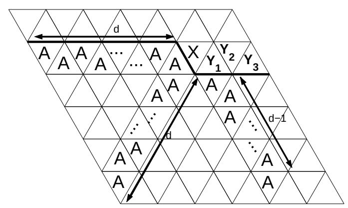


Figure 5.11: Case 2 of Lemma 15.

Case 2a: X can be a *one* and Y_1 can be a *one*. Clearly, all positions marked by B are *zeroes* (see Fig. 5.12). X can be a *zero* and hence either C_1 or C_2

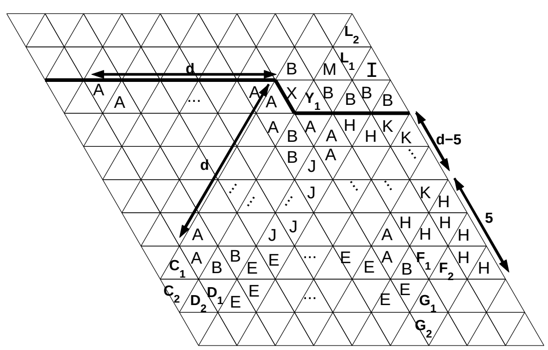


Figure 5.12: Case 2a of Lemma 15.

is a *one*. Y_1 can be a *zero* and therefore either D_1 or D_2 is a *one*. It implies that C_1 and D_1 are *ones*. Hence, all positions marked by E are *zeroes*. By the horizontal constraint either F_1 or F_2 is a *one*. Since Y_1 can be a *zero*, it follows that either G_1 or G_2 is a *one*. Hence, F_1 is a *one* and all positions marked by H are *zeroes*.

Assume I will be a *one*. Then all positions marked by J will be *zeroes*, creating a run of $d+4$ zeroes in their right column, a contradiction. Therefore, I is labelled by a *zero*.

Assume all the $d-5$ positions marked by K are *zeroes*. Then L_1 is labelled by a *one* and Y_1 cannot be a *one*, a contradiction. Hence, one of the positions marked by K is a *one*, L_1 and L_2 are labelled by *zeroes*.

Therefore, if X will be a *one* then Y_1 will be a *zero* and by its right column constraint M will be a *one*. M, X, C_1, and D_1 will form the pattern PEven, and hence by Lemma 14 all the prefix of the row before X is a given sequence $\mathcal{P}(i,j)$, and we are in state (s3).

Case 2b: X can be a *one* and Y_2 can be a *one*. Clearly, all positions marked by B are *zeroes* (see Fig. 5.13). X can be a *zero* and hence exactly one of the C_i's is a *one*, and exactly one of the D_i's is a *one*. Y_2 can be a *zero* and therefore exactly one of the E_i's is a *one*, and exactly one of the F_i's is a *one*. Clearly, D_3 and E_3 cannot be *ones*.

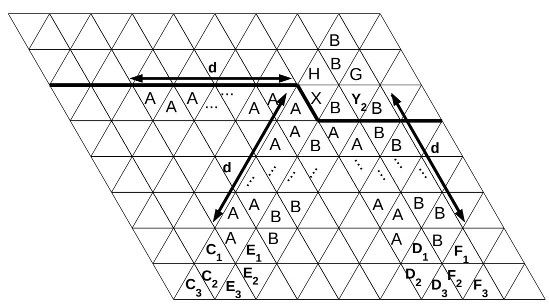


Figure 5.13: Case 2b of Lemma 15.

- If E_2 is a *one* then C_1 is a *one*. If X will be a *one* then Y_2 will be a *zero* and by its left column constraint G will be a *one*. E_2, C_1, X, and G will form the pattern PEven, and hence by Lemma 14 all the prefix of the row before X is a given sequence $\mathcal{P}(i,j)$, and we are in state (s3).
- If D_2 is a *one* then F_1 is a *one*. If Y_2 will be a *one* then X will be a *zero* and by its right column constraint H will be a *one*. D_2, F_1, Y_2, and H

will form the pattern PEven, and hence by Lemma 14 the suffix of the current row is completely determined, and we are in state (s2).

- If D_2, D_3, E_2, and E_3 are *zeroes* then D_1 and E_1 are *ones* which is impossible since the gap between them is $d-1$ and the horizontal constraint will be violated.

Case 2c: X can be a *one* and Y_3 can be a *one*. Clearly, all positions marked

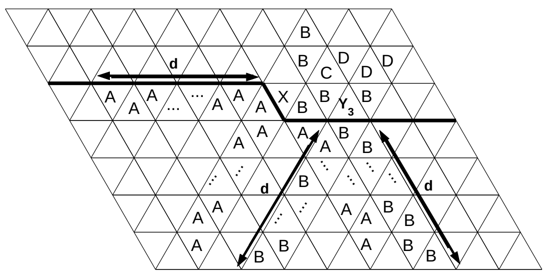


Figure 5.14: Case 2c of Lemma 15.

by B are *zeroes* (see Fig. 5.14). If X will be a *one* then Y_3 will be a *zero* and by its left column constraint C will be a *one*. Hence, all the positions marked by D will be labelled by *zeroes*, creating a run of $d+4$ *zeroes* in the right column of Y_3, a contradiction.

Thus, by Theorem 2, $C_\triangle(d, d+3) = 0$ for every even $d \geq 6$. ■

Similarly to Lemma 14, the following lemma will be used in Lemma 17, when proving that $C_\triangle(d, d+3) > 0$ for odd $d \geq 5$.

Lemma 16 *Let $d \geq 5$ be an odd integer, $h = \frac{d+7}{2}$, and let $\mathcal{A}$ be an infinite $(d, d+3)$ array. If $\mathcal{A}$ contains an $r \times h$ sub-array $\mathcal{B}$ whose first two rows form the pattern POdd (see Fig. 5.15), then the first two and the last two right columns of $\mathcal{B}$ are substrings of $(10^{d+2})^\infty$.*

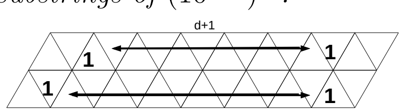


Figure 5.15: The pattern POdd

Proof. Let $\mathcal{C}$ be an $(r+2) \times h$ right sub-array of $\mathcal{A}$ with the pattern POdd as depicted in Fig. 5.16. Clearly the positions marked by A are *zeroes*.

Assume the position marked by B is a *one*. Then the $d-4$ positions marked by C will be *zeroes*, creating a run of $d+4$ *zeroes* in their right column, a contradiction. Therefore, B is a *zero* and either D_1 or D_2 is a *one*.

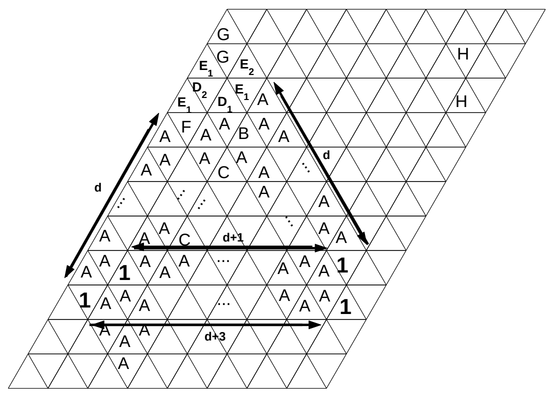


Figure 5.16: Labels implied by the pattern POdd.

Assume D_1 is a *one*. Then D_2 and all positions marked by E_1 or E_2 will be *zeroes*. Hence, by the right column constraint, F will be a *one* and the two positions marked by G will be *zeroes*, and it will create a run of $d+4$ *zeroes* in their left column, a contradiction. Therefore, D_1 is a *zero* and D_2 is a *one*. It implies that all positions marked by E_1 or G are *zeroes*, and hence E_2 is a *one*.

The four *ones* in the left columns of D_2 and E_2 form the pattern POdd and hence by the same arguments the two positions marked by H are *ones*. The positions marked by D_2, E_2, and H form again the pattern POdd. The claim of the lemma is proved now by induction. ■

Similarly to Lemma 15 we have the following lemma.

Lemma 17 *If $d \geq 5$ is odd then $C_{\triangle}(d, d+3) = 0$.*

Proof. We will use the scanning technique again. Assume we have to label the next position marked by X. We have to distinguish between two different types of positions as depicted in Figure 5.6.

Case 1: Assume that X, as depicted in Figure 5.17, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, and either B_1 or B_2 is a *one*. Therefore, the positions marked by C are *zeroes* and at least one of the following three cases must be valid.

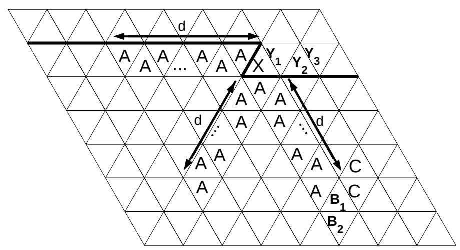


Figure 5.17: Case 1 of Lemma 17.

Case 1a: X can be a *one* and Y_1 can be a *one* (see Fig. 5.18). Clearly, all positions marked by D are *zeroes*, E is a *one*, and hence B_1 is a *zero* and B_2 is a *one*. If X will be a *zero* then Y_1 and F will be *ones*. E, B_2, Y_1 and F will form the pattern Podd, and hence by Lemma 16 the suffix of the current row is completely determined, and we are in state (s2).

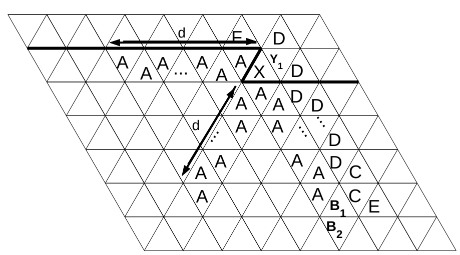


Figure 5.18: Case 1a of Lemma 17.

Case 1b: X can be a *one* and Y_2 can be a *one* (see Fig. 5.19). If Y_2 will be a *one* then all positions marked by D are *zeroes*, and hence E will be a *one*. Therefore, the positions marked by F are *zeroes*, and since also X will be a *zero*, it follows that there is a run of $d+4$ zeroes in the left column of X, a contradiction.

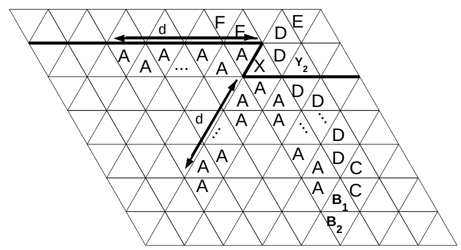


Figure 5.19: Case 1b of Lemma 17.

Case 1c: X can be a *one* and Y_3 can be a *one* (see Fig. 5.20). Clearly, all

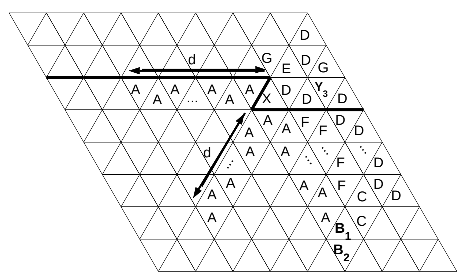


Figure 5.20: Case 1c of Lemma 17.

positions marked by D are *zeroes*. If X will be a *zero*, then Y_3 will be a *one* and the position marked by E will be a *one*, and hence all positions marked by F are *zeroes*. If X will be a *one*, then Y_3 will be a *zero* and E will be a *zero*, and to avoid a run of length $d+4$ in the left columns we must have *ones* in the positions marked by G, which is impossible.

Case 2: Assume that X, as depicted in Fig. 5.21, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, and at least one of the following three cases must be valid.

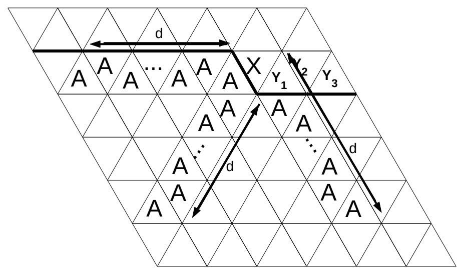


Figure 5.21: Case 2 of Lemma 17.

Case 2a: X can be a *one* and Y_1 can be a *one* (see Fig. 5.22). Clearly, all positions marked by B are *zeroes*. If Y_1 will be a *one* then X will be a *zero*, and therefore either D_1 or D_2 is a *one*. If X will be a *one* then Y_1 will be a *zero*, and therefore either E_1 or E_2 is a *one*. Hence, D_2 and E_2 are *ones*. If X will be a *one* then F will be a *one*. D_2, E_2, X and F will form the pattern Podd, and hence by Lemma 14 all the prefix of the row before X is completely determined, and we are in state (s3).

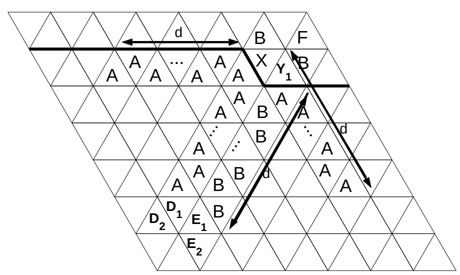


Figure 5.22: Case 2a of Lemma 17.

Case 2b: X can be a *one* and Y_2 can be a *one* (see Fig. 5.23). Clearly, all positions marked by B are *zeroes*. X can be a *zero* and hence exactly one of the C_i's is a *one*, and exactly one of the D_i's is a *one*. Y can be a *zero* and hence exactly one of the E_i's is a *one*, and exactly one of the F_i's is a *one*. Clearly, C_1 and F_1 cannot be *ones*.

- If C_2 is a *one* then E_3 is a *one*. If X will be a *one*, then by the left column constraint G will be a *one*. C_2, E_3, X and G will form the pattern Podd, and hence by Lemma 16 all the prefix of the row before X is completely determined, and we are in state (s3).

- If F_2 is a *one* then D_3 is a *one*. If Y_2 will be a *one* then by the right column constraint H will be a *one*. F_2, D_3, Y_2 and H will form the pattern Podd, and hence by Lemma 16 the suffix the current row is completely determined, and we are in state (s2).

- If C_2 and F_2 are *zeroes* then C_3 and F_3 are *ones* which is impossible since the gap between them is $d + 4$ and the horizontal constraint will be violated.

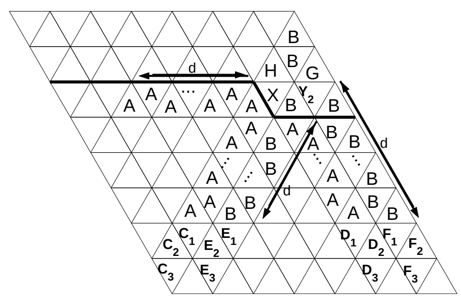


Figure 5.23: Case 2b of Lemma 17.

Case 2c: X can be a *one* and Y_3 can be a *one* (see Fig. 5.24). Clearly, all

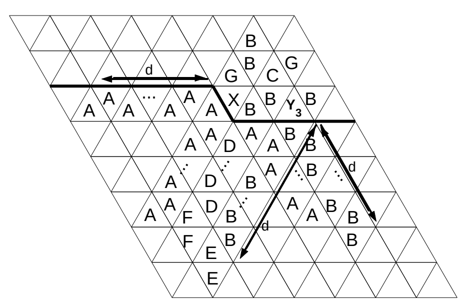


Figure 5.24: Case 2c of Lemma 17.

positions marked by B are *zeroes*. If X will be a *one* then C will be a *one* by the left column constraint, and hence all the positions marked by D are *zeroes*; Y_3 will be a *zero* and hence one of the positions marked by E is a *one* and the positions marked by F are *zeroes*. If Y_3 will be a *one* then C will be a *zero* and hence by the right columns constraint both positions marked by G should be *ones* which is impossible by the horizontal constraint.

Thus, by Theorem 2, $C_\triangle(d, d+3) = 0$ for every odd $d \geq 5$. ■

Corollary 1 $C_\triangle(d, d+3) = 0$ *for* $d \geq 5$.

5.3 The Capacity for Small Values of d

For small values of d the zero/positive capacity region is slightly different, and is described in this section.

Lemma 18 $C_\triangle(1, k) > 0$ *if and only if* $k \geq 3$.

Proof. We first show that $C_\triangle(1, 2) = 0$. This is done by a simple scanning argument. Assume we have to label the next scanned position marked by X. We have to distinguish between the two different types of orientations of the position as depicted in Fig. 5.6.

Case 1: Assume that X, as depicted in Figure 5.25, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that the positions marked by A are *zeroes*. If X will be a *zero*, there will be a run of 3 *zeroes* in the left column, a contradiction.

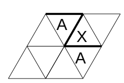


Figure 5.25: Case 1 of Lemma 18.

Case 2: Assume that X, as depicted in Figure 5.26, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that the position marked by A is a *zero*. If X will be a *zero*, Y will be a *one* by the horizontal constraint, and therefore B is a *zero*. Moreover, if X will be a *zero* there will be 2 *zeroes* in the right column, hence C is a *one*. Similarly, Y can be a *zero*, which implies that D is a *one*. C and D are adjacent *ones*, a contradiction.

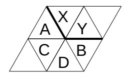


Figure 5.26: Case 2 of Lemma 18.

This completes the proof that $C_{\triangle}(1,2) = 0$. This proof can be trivially generalized to show that $C_{\triangle}(d, d+1) = 0$ for every odd d, but our results for the triangle model are stronger, and hence the generalization is omitted.

We now show that $C_{\triangle}(1,3) > 0$. Consider the $(1,1)$ array of size $n \times n$, where $(i, j, 0) = 1$ (see Fig. 5.27). Clearly, any change of nonconsecutive *ones* into

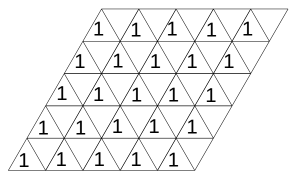


Figure 5.27: The array for the proof that $C_{\triangle}(1,3) > 0$.

zeroes, results in a $(1,3)$ array. Any tiling of the plane with the lattice points $\{(x,y) \; : \; x = i, \; y = i + 3j, \; i,j \in \mathcal{Z}\}$, using the two triangular tiles of Fig. 5.28, corresponds to some array constructed in the above manner. By

Figure 5.28: Two triangular tiles to prove that $C_{\triangle}(1,3) > 0$.

Lemma 3, this tiling implies that $C_{\triangle}(1,3) \geq \frac{1}{6}$.

In this proof, we make use of the fact that any change of nonconsecutive *ones* into *zeroes* in the $(1,1)$ array, results in a $(1,3)$ array. But the lower bound on the capacity that is achieved by the corresponding tiling, can be much improved by noticing the following. The *ones* in the $(1,1)$ array form a hexagonal lattice, where two consecutive *ones* correspond to adjacent hexagons. Any set of nonconsecutive *ones* is an independent set in the hexagonal lattice. The exact number of independent sets in the hexagonal model, also known as the number of arrays in the hard hexagonal model, has been given by Baxter in [1]. Since the hexagonal lattice induced by the *ones* is a hexagonal $n \times m$ array, we have the following bound on the capacity:

$$C_{\triangle}(1,3) = \lim_{n,m\to\infty} \frac{\log_2 N(n,m \mid (1,3))}{2nm} \geq$$
$$\geq \frac{1}{2} \cdot C_{\hexagon}(1,\infty) \approx \frac{1}{2} \cdot 0.480767... \approx 0.240383...,$$

which is better than the bound of $\frac{1}{6}$ given by the tiling. ■

Lemma 19 *$C_{\triangle}(2,k) > 0$ if and only if $k \geq 4$.*

Proof. We first show that $C_{\triangle}(2,3) = 0$. Clearly $C_{\triangle}(3,3) = 0$, hence if $C_{\triangle}(2,3) > 0$, then there exists a $(2,3)$ array that has a run of *zeroes* whose length is exactly 2. We analyze such an array, and show that a run of *zeroes* whose length is 4 must exists. Let $\mathcal{A}$ be an $n \times n$ array with a run of *zeroes* of length 2 as depicted in Fig. 5.29. The leftmost *one* implies that the position

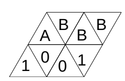


Figure 5.29: A forced run of 4 *zeroes* in a $(2,k)$ triangular array.

marked by A is a *zero*, and the rightmost *one* implies that the positions marked by B are *zeroes*, which creates a run of 4 *zeroes* horizontally. Hence, $C_{\triangle}(2,3) = 0$.

We now show that $C_{\triangle}(2,4) > 0$. Any tiling of the plane with the lattice points $\{(x,y) \ : \ x = 3i, \ y = 3i + 9j, \ i,j \in \mathcal{Z}\}$, using the two triangular arrays of Fig. 5.30 is a valid $(2,4)$ array. By Lemma 3, this tiling implies

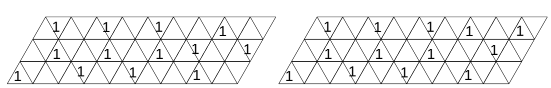


Figure 5.30: Two triangular arrays to prove that $C_{\triangle}(2,4) > 0$.

that $C_{\triangle}(2,4) \geq \frac{1}{54}$. ■

Lemma 20 $C_\triangle(3,k) > 0$ *if and only if* $k \geq 7$.

Proof. First, we use the scanning technique again to prove that $C_\triangle(3,6) = 0$. Assume we have to label the next scanned position marked by X. We have to distinguish between the two different types of orientations of the position as depicted in Fig. 5.6.

Case 1: Assume that X, as depicted in Figure 5.31, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, therefore at least one of the following three cases must be valid.

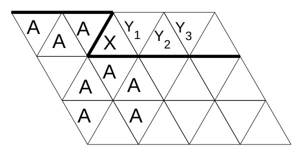


Figure 5.31: Case 1 of Lemma 20.

Case 1a: X can be a *one* and Y_1 can be a *one* (see Fig. 5.32). Clearly, all positions marked by B are *zeroes*. X can be a *zero*, therefore either C_1 or C_2 is a *one*, and D is a *zero*. Y_1 can be a *zero*, therefore E is a *one*, C_1 is a *zero*, C_2 is a *one*, and the positions marked by F are *zeroes*. Hence, by the right column constraint G is a *one*, which implies that H is a *zero*. This implies that either I_1 or I_2 is a one, and the positions marked by J are zeroes, which creates a run of 8 zeroes in the left column when X is a *zero*, a contradiction.

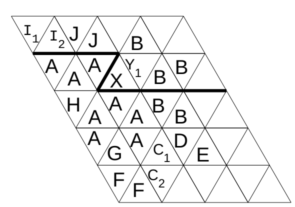


Figure 5.32: Case 1a of Lemma 20.

Case 1b: X can be a *one* and Y_2 can be a *one* (see Fig. 5.33). Clearly, all positions marked by B are *zeroes*. X can be a *zero*, therefore C is a *one*, and similarly Y_2 can be a *zero*, therefore D is a *one*. This implies that all positions marked by E are *zeroes*. By the horizontal constraint F is a *one*, and therefore G is a *zero*, which creates a run of 7 zeroes in the left column when X is a *zero*, a contradiction.

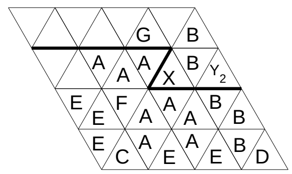


Figure 5.33: Case 1b of Lemma 20.

Case 1c: X can be a *one* and Y_3 can be a *one* (see Fig. 5.34). Clearly, all positions marked by B are *zeroes*, therefore C is a *one*, and all positions marked by D are *zeroes*, which creates a run of 7 zeroes in the left column when X is a *zero*, a contradiction.

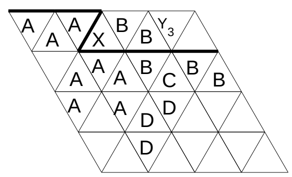


Figure 5.34: Case 1c of Lemma 20.

Case 2: Assume that X, as depicted in Figure 5.35, is not uniquely determined, i.e., it can be labelled by a *zero* and it can be labelled by a *one*. It implies that all the positions marked by A are *zeroes*, either X or one of the three positions to the right of X is a *one*, therefore at least one of the following three cases must be valid.

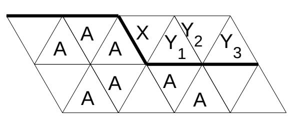


Figure 5.35: Case 2 of Lemma 20.

Case 2a: X can be a *one* and Y_1 can be a *one* (see Fig. 5.36). Clearly, all positions marked by B are *zeroes*. X can be a *zero*, therefore either C_1 or C_2 is a *one*. Y_1 can be a *zero*, therefore either D_1 or D_2 is a *one*. This implies that C_2 and D_2 are *ones*, and the positions marked by E are *zeroes*. By the horizontal constraint F is a *one*, and the positions marked by G are *zeroes*. By the left column constraint H is a *one*, and the positions marked by I are *zeroes*. X can by a *zero*, therefore by the horizontal constraint J is a *one*, and K is a *zero*. If X will be a *one*, Y_1 will be a *zero*, hence L will be a *one* and the positions marked by M will be *zero*, which will create a horizontal run of 7 zeroes, a contradiction.

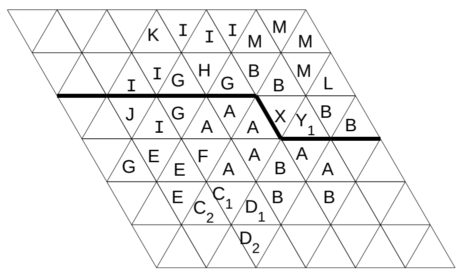


Figure 5.36: Case 2a of Lemma 20.

Case 2b: X can be a *one* and Y_2 can be a *one* (see Fig. 5.37). Clearly, all positions marked by B are *zeroes*, which creates a horizontal run of 7 zeroes, a contradiction.

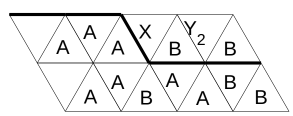


Figure 5.37: Case 2b of Lemma 20.

Case 2c: X can be a *one* and Y_3 can be a *one* (see Fig. 5.38). Clearly, all positions marked by B are *zeroes*, therefore C is a *one*, and all positions marked by D are *zeroes*. Y_3 can be a *zero*, therefore by the right column constraint, E is a *one*, and all positions marked by F are *zeroes*. X can be a *zero*, therefore by the right column constraint G is a *one*, all positions marked by H are *zeroes*, and either I_1 or I_2 is a *one*. This implies that J is a *zero*, which creates a run of 7 zeroes in the right column when X is a *zero*, a contradiction. This completes the proof that $C_{\triangle}(3,6) = 0$.

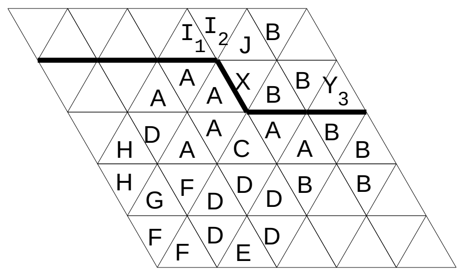


Figure 5.38: Case 2c of Lemma 20.

We now show that $C_{\triangle}(3,7) > 0$. Consider the $(3,3)$ array of size $n \times n$, where $(2i, 2j, 1) = 1$ and $(2i+1, 2j+1, 0) = 1$ (see Fig. 5.39). Clearly, any change

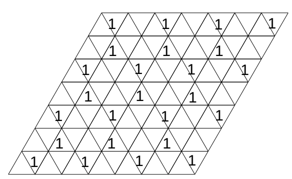


Figure 5.39: The array for the proof that $C_{\triangle}(3,7) > 0$.

of nonconsecutive *ones* into *zeroes*, results in a $(3,7)$ array. Any tiling of the plane with the lattice points $\{(x,y) \ : \ x = 2i, \ y = 2i + 6j, \ i,j \in \mathcal{Z}\}$, using the four triangular tiles of Fig. 5.40, corresponds to some array constructed in the above manner. By Lemma 3, this tiling implies that $C_{\triangle}(3,7) \geq \frac{1}{12}$.

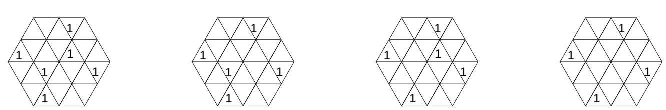


Figure 5.40: Four triangular tiles to prove that $C_{\triangle}(3,7) > 0$.

As in Lemma 20, the lower bound on the capacity that is achieved by the corresponding tiling, can be much improved by noticing that the *ones* in the $(3,3)$ array form two hexagonal lattices, where two consecutive *ones* correspond to adjacent hexagons. Since each hexagonal lattice induced by the *ones* is a hexagonal $\frac{n}{2} \times \frac{m}{2}$ array, we have the following bound on the capacity:

$$C_{\triangle}(3,7) = \lim_{n,m \to \infty} \frac{\log_2 N(n,m \mid (3,7))}{2nm} \geq$$

$$\geq \frac{1}{2} \cdot \left[\frac{1}{4} C_{\hexagon}(1,\infty) + \frac{1}{4} C_{\hexagon}(1,\infty) \right] =$$

$$= \frac{1}{4} \cdot C_{\hexagon}(1,\infty) \approx \frac{1}{4} \cdot 0.480767... \approx 0.120191...,$$

which is better than the bound of $\frac{1}{12}$ given by the tiling. ■

Lemma 21 *$C_{\triangle}(4,k) > 0$ if and only if $k \geq 9$.*

Proof. By Lemma 17 we have that $C_{\triangle}(5,8) = 0$. Hence if $C_{\triangle}(4,8) > 0$, then there exists a $(4,8)$ array that has a run of *zeroes* whose length is exactly 4. We analyze such an array, and show that a run of *zeroes* whose length is 9 must exists. Let $\mathcal{A}$ be an $n \times n$ array with a run of *zeroes* of length 4 as depicted in Fig. 5.41. Clearly the positions marked by A are *zeroes*. Assume

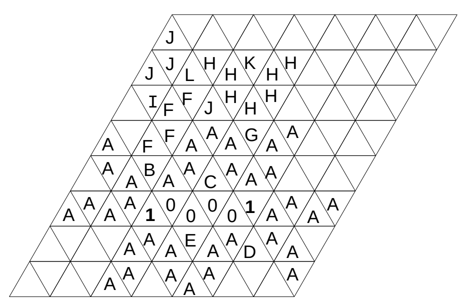


Figure 5.41: Proving $C_{\triangle}(4,8) = 0$.

the position marked by B is a *zero*. Then, by the horizontal constraint C is a *one*, and by the left and right columns, D and E are *zeroes*, which creates a run of 9 *zeroes* horizontally. Hence, B is a *one*, all the positions marked by F are *zeroes*, C and E are *zeroes*, and D is a *one*. This implies that G must be a *one* by the right column constraint, and all the positions marked by H are *zeroes*. I is a *one* by the left column constraint, and hence all the positions marked by J are *zeroes*. Therefore, K is a *one* by the right column constraint, and L is a *zero*, which creates a run of 9 *zeroes* in that left column. Hence, $C_\triangle(4, 8) = 0$.

By using $d = 5$ in Lemma 12 we have that $C_\triangle(5, 9) > 0$. Therefore Lemma 1 implies that $C_\triangle(4, 9) > 0$, which completes the proof. ■

5.4 Summary of Results for the Triangular Model

This chapter shows a tight characterization for $C_\triangle(d, k)$ when $d \equiv 1 \ (mod\ 4)$, given by Lemmas 12 and 17:

Corollary 2 *For every* $d \equiv 1 \ (mod\ 4)$, $d \geq 5$ *we have:* $C_\triangle(d, k) > 0$ *if and only if* $k \geq 4$.

For other values of d, by Lemmas 12 and 1, we have:

Corollary 3

- $C_\triangle(d, d + 5) > 0$ *if* $d \equiv 0 \ (mod\ 4)$
- $C_\triangle(d, d + 6) > 0$ *if* $d \equiv 3 \ (mod\ 4)$
- $C_\triangle(d, d + 7) > 0$ *if* $d \equiv 2 \ (mod\ 4)$

By Corollary 1 we have that $C_\triangle(d, d + 3) = 0$ for all $d \geq 5$, hence the remaining gaps are relatively small.

Chapter 6

Discussion and Open Problems

In this work we considered the positive capacity region of two-dimensional run-length constrained channels in a few connectivity models – the diamond, square, and triangular models. We have managed to find some regions where the capacity is positive and some in which the capacity is zero, by using generalizations and modifications of known techniques.

6.1 The Scanning Method

The main contribution regarding techniques for proving zero capacity, is the generalization of the scanning method of [2] in Theorem 2. Previous techniques for proving zero capacity, strongly depended on the specific constraint they were applied to. The proofs were much longer and required the consideration of many different cases. The alternative proof in chapter 2 for the result of Kato and Zeger that $C_{\diamond}(d, d+1) = 0$, shows the efficiency of the scanning method. Perhaps more important, is that the generalization of scanning method allows to determine zero capacity for constraints Θ that have larger values of $N(n, m \mid \Theta)$. An interesting path for further research is generalizing the scanning method to handle constraints in which the number of constrained arrays is much larger.

6.2 Bounding the Capacity

When proving positive capacity, we find tiles with different labels, and show that tiling the plane with them induces valid arrays. This implies a bound on the capacity as described in Lemma 3. The proofs of Lemmas 18 and 20 show that the bound induced by the tiling could be far from the actual capacity.

How good are these bounds for other constraints? For example, can the bound of Lemma 12, $C_{\triangle}(d, d+4) \geq \frac{1}{2(d+3)} log_2 3$ for $d \equiv 1 \ (mod\ 4)$, be improved?

6.3 The Connectivity Models

6.3.1 The Diamond Model

We considered asymmetric constraints in the diamond model in Chapter 3. We solved most of the open cases of [14], using the techniques presented in Chpater 2, and showed a characterization of the zero/positive capacity region in which only one case remains unsolved. We would like to see the capacity of the last case determined:

for $2 \leq d_1$, $d_1 + 2 \leq k_1 \leq 2d_1$, $d_2 = k_1 - 1$, $d_2 + 2 \leq k_2 \leq 2d_2$, is $C_{\diamond}(d_1, k_1, d_2, k_2) = 0$ or $C_{\diamond}(d_1, k_1, d_2, k_2) > 0$?

6.3.2 The Square Model

The gaps between the known zero and positive capacity regions in the square model are relatively large. In Chapter 4 we proved that $C_{\boxplus}(d, d+3) = 0$ for every $d \geq 1$, but the known positive capacities are much farther.

Further research should attempt to find an infinite set S of positive integers, and an integer r, such that $C_{\boxplus}(d, d+r) = 0$ and $C_{\boxplus}(d, d+r+1) > 0$ for each $d \in S$.

6.3.3 The Triangular Model

We considered the triangular model in Chapter 5 and showed a tight characterization of the positive capacity region for many values of d. We showed that $C_{\triangle}(d, k) > 0$ if and only if $k \geq d + 4$, for every $d \equiv 1 (mod\ 4)$. Together with the proof that $C_{\triangle}(d, d+3) = 0$ for every $d \geq 3$, it implies that the gaps between the zero and positive capacity regions in this model are relatively small. A full characterization in the triangular model is yet to be determined.

Bibliography

[1] R. J. Baxter. Hard hexagons: Exact solution. *J. Phys. A: Math. Gen.*, 13:L61–L70, 1980.

[2] S. R. Blackburn. Two-dimensional runlength constraint arrays with equal horizontal and vertical constraints. *IEEE Transactions on Information Theory, submitted*, 2004.

[3] N. J. Calkin and H. S. Wilf. The number of independent sets in a grid graph. *SIAM J. Discret. Math.*, 11(1):54–60, 1998.

[4] K. Censor and T. Etzion. The positive capacity region of two-dimensional run length constrained channels. *IEEE Symposium on Information Theory, Seattle WA, submitted*, 2006.

[5] J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups.* New York, Springer-Verlag, 1988.

[6] S. I. R. Costa, M. Muniz, E. Agustini, and R. Palazzo. Graphs, tessellations, and perfect codes on flat tori. *IEEE Transactions on Information Theory*, 50(10):2363–2377, 2004.

[7] T. Etzion and K. G. Paterson. Zero/positive capacities of two-dimensional runlength-constrained arrays. *IEEE Transactions on Information Theory*, 51(9):3186–3199, 2005.

[8] T. Etzion and A. Vardy. Two-dimensional interleaving schemes with repetitions: Constructions and bounds. *IEEE Transactions on Information Theory*, 48(2):428–457, 2002.

[9] S. Halevy, J. Chen, R. M. Roth, P. H. Siegel, and J. K. Wolf. Improved bit-stuffing bounds on two-dimensional constraints. *IEEE Transactions on Information Theory*, 50(5):824–838, 2004.

[10] K. A. S. Immink. *Coding Techniques for Digital Recorders*. New York, Prentice Hall, 1991.

[11] K. A. S. Immink. *Codes for Mass Data Storage Systems*. The Netherlands, Shannon Foundation Publishers, 1999.

[12] W. Weeks IV and R. E. Blahut. The capacity and coding gain of certain checkerboard codes. *IEEE Transactions on Information Theory*, 44(3):1193–1203, 1998.

[13] A. Kato and K. Zeger. On the capacity of two-dimensional run-length constrained channels. *IEEE Transactions on Information Theory*, 45(5):1527–1540, 1999.

[14] A. Kato and K. Zeger. Partial characterization of the positive capacity region of two-dimensional asymmetric run length constrained channels. *IEEE Transactions on Information Theory*, 46(7):2666–2670, 2000.

[15] Zs. Kukorelly and K. Zeger. Automated theorem proving for hexagonal run length constrined capacity computation. *IEEE Symposium on Information Theory, Seattle WA, submitted*, 2006.

[16] Zs. Kukorelly and K. Zeger. The capacity of some hexagonal (d,k) constraints. *IEEE International Symposium on Information Theory, Washington DC*, page 263, June 2001.

[17] B. H. Marcus, R. M. Roth, and P. H. Siegel. An introduction to coding for constrained systems. *Lecture Notes*, 2001.

[18] Zs. Nagy and K. Zeger. Asymptotic capacity of two-dimensional channels with checkerboard constraints. *IEEE Transactions on Information Theory*, 49(9):2115–2125, 2003.

[19] Zs. Nagy and K. Zeger. Capacity bounds for the hard triangle model. *IEEE International Symposium on Information Theory, Chicago Illinois*, page 162, June 2004.

[20] R. M. Roth, P. H. Siegel, and J. K. Wolf. Efficient coding schemes for the hard-square model. *IEEE Transactions on Information Theory*, 47(3):1166–1176, 2001.

[21] M. Schwartz and T. Etzion. Two-dimensional cluster-correcting codes. *IEEE Transactions on Information Theory*, 51(6):2121–2132, 2005.

[22] P. H. Siegel and J. K. Wolf. Bit-stuffing bounds on the capacity of 2-dimensional constrained arrays. *IEEE International Symposium on Information Theory, Cambridge MA*, page 323, Aug. 1998.

[23] R. Talyansky. Coding for two-dimensional constraints. *M.Sc. Thesis (in Hebrew), Computer Science Department, Technion, Haifa, Israel*, 1997.

צפנים עם אילוצים לערוצים דו-מימדיים

קרן צנזור

צפנים עם אילוצים לערוצים דו-מימדיים

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים במדעי המחשב

קרן צנזור

הוגש לסנט הטכניון - מכון טכנולוגי לישראל

אדר תשס״ו חיפה מרץ 2006

המחקר נעשה בהנחיית פרופ׳ טובי עציון בפקולטה למדעי המחשב.

אני מודה מקרב לב לפרופ׳ טובי עציון על הנחייתו המסורה.

תודה לאחייניי - אופק, נגה, נטע ואלמוג, על אורם.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי.
כמו כן, אני מודה לקרן ז׳נט וסמואל לובל.

תוכן עניינים

רשימת איורים

תקציר

במערכות לאחסון מידע דיגיטלי, כגון מערכות מגנטיות או אופטיות, המידע המאוחסן צריך לקיים אילוצים מסוימים הנובעים מהמבנה הפיסיקלי של המדיה. אחד מסוגי האילוצים הנחקרים ביותר הם אילוצי ***(d,k) run-length limited (RLL)***. מחרוזת בינארית מקיימת את אילוץ (d, k) החד-מימדי אם כל רצף של אפסים הוא באורך לפחות d ולכל היותר k.

קידוד חד-מימדי

כדי לאחסן את המידע דרוש **מקודד**, שיכול לקבל כקלט כל מחרוזת בינארית באורך p, וצריך לפלוט מחרוזת בינארית באורך n שמקיימת את האילוץ.

הקצב של המקודד הוא היחס $\frac{p}{n}$. המטרה היא למצוא מהו הקצב המקסימלי האפשרי עבור מקודד לאילוץ נתון. **הקיבול** של אילוץ חד-מימדי θ מוגדר על ידי:

$$C(\theta) = \lim_{n \to \infty} \frac{\log_2 N(n \mid \theta)}{n}$$

כאשר $N(n \mid \theta)$ הוא מספר המחרוזות הבינאריות מאורך n שמקיימות את האילוץ θ.

הקיבול $C(\theta)$ מהווה חסם עליון עבור קצב אפשרי של מקודד עבור האילוץ θ.

התחום של אילוצים חד-מימדיים נחקר בצורה מקיפה, בייחוד אילוצים מסוג RLL. ניתן לחשב במדויק את הקיבול של אילוצים מסוג זה בעזרת שיטות אלגבריות.

קידוד דו-מימדי

בהתפתחויות האחרונות בשימוש במדיה אופטית, במיוחד בתחום של אחסון הולוגרפי, מתייחסים למידע הנשמר בצורה דו-מימדית. מערך דו-מימדי מקיים את האילוץ (d, k), אם מתקיים האילוץ (d, k) החד-מימדי בכל אחד מכיווני המערך, הנקבעים על פי המודל הגיאומטרי. באופן דומה למקרה החד-מימדי, מוגדר הקיבול של אילוץ דו-מימדי θ באופן הבא:

$$C(\theta) = \lim_{n,m\to\infty} \frac{\log_2 N(n,m\mid\theta)}{nm}$$

כאשר $N(n,m \mid \theta)$ הוא מספר המערכים הבינאריים מגודל $n \times m$ שמקיימים את האילוץ θ.

בניגוד לאילוצים החד-מימדיים שנחקרו בצורה מעמיקה, עדיין לא ידועות שיטות כלליות לחישוב הקיבול של אילוצים דו-מימדיים. עבודה זו עוסקת בבעיה המצומצמת של קביעה האם הקיבול של אילוץ דו-מימדי כלשהו הוא אפס או חיובי, במודלים השונים.

מודלים גיאומטריים

קיימים מספר מודלים גיאומטריים לאחסון מידע דו-מימדי. מודל ארבעת השכנים, מודל שמונת השכנים, מודל המשושים, ומודל המשולשים. האילוצים יכולים להיות אסימטריים, כלומר יכולים להשתנות בין הכיוונים השונים של המודל.

מודל ארבעת השכנים מתואר על ידי $\mathbb{Z}^2$. לכל תא (i, j) במערך יש ארבעה תאים שכנים:

$(i+1, j)$, $(i-1, j)$, $(i, j+1)$, ו $(i, j-1)$.

במודל זה קריאת המידע נעשית בשני כיוונים – בכיוון אופקי ובכיוון אנכי. מערך דו-מימדי במודל ארבעת השכנים מקיים את האילוץ (d, k) אם מתקיים האילוץ (d, k) החד-מימדי בכל שורה ובכל עמודה. במודל זה נעסוק גם באילוצים אסימטריים, כלומר, מערך במודל ארבעת השכנים מקיים את האילוץ (d_1, k_1, d_2, k_2) אם הוא מקיים את האילוץ (d_1, k_1) החד-מימדי בכל שורה, ואת האילוץ (d_2, k_2) החד-מימדי בכל עמודה.

מודל שמונת השכנים מתואר גם הוא על ידי $\mathbb{Z}^2$, אלא שכאן יש לכל תא במערך שמונה שכנים:

$(i+1, j)$, $(i-1, j)$, $(i, j+1)$, $(i, j-1)$,
$(i+1, j+1)$, $(i-1, j+1)$, $(i+1, j-1)$, ו $(i-1, j-1)$.

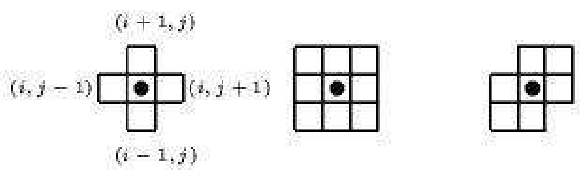
קריאת המידע מתבצעת בארבעה כיוונים – בכיוון אופקי, בכיוון אנכי ובשני האלכסונים.

מודל המשושים מתקבל מריצוף המישור על ידי משושים משוכללים. כל מרכז של משושה מהווה תא, ולכל תא יש ששה שכנים והמידע נקרא בשלושה כיוונים. קיים ייצוג נוסף למודל המשושים שהוא איזומורפי לייצוג הנ"ל. בייצוג זה המודל מתואר על ידי $\mathbb{Z}^2$, ובו לכל תא יש ששה שכנים:

$(i + 1,\ j)$, $(i - 1,\ j)$, $(i,\ j + 1)$, $(i,\ j - 1)$, $(i + 1, j + 1)$ ו $(i - 1, j - 1)$.

גם כאן קריאת המידע מתבצעת בשלושה כיוונים – בכיוון אופקי, בכיוון אנכי ובכיוון של האלכסונים הימניים.

להלן מתוארים השכנים של כל תא במודל ארבעת השכנים, מודל שמונת השכנים ומודל המשושים, בהתאמה:



מודל המשולשים מתקבל באופן הבא. אנו מתחילים מריצוף המישור על ידי משושים משוכללים, כמו במודל המשושים. לאחר מכן מחברים את מרכזי המשושים בעזרת קטעים, כך שמתקבל ריצוף של המישור על ידי משולשים שווי צלעות. כל משולש מהווה תא, ששכניו הם שלושת המשולשים הצמודים אליו. באופן זה, כל תא ניתן לייצוג על ידי שלישיה (i, j, s) ב $\mathbb{Z}^2 \mathrm{x}\{0,1\}$,

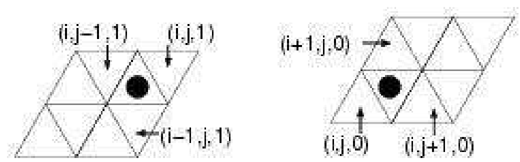
כאשר לתא (i, j, 0) יש שלושה שכנים:

$(i,\ j,\ 1)$, $(i - 1,\ j,\ 1)$, $(i,\ j - 1,\ 1)$,

ולתא (i, j, 1) יש שלושה שכנים:

$(i,\ j,\ 0)$, $(i + 1,\ j,\ 0)$, $(i,\ j + 1,\ 0)$.

קריאת המידע מתבצעת בשלושה כיוונים שונים על ידי מעבר משכן לשכן. להלן מתוארים השכנים של כל תא במודל המשולשים:



במודל זה, במערך של *n* שורות ו *m* עמודות יש *2nm* תאים, ולכן הגדרת הקיבול משתנה בהתאם, והיא:

$$C_\Delta(\theta) = \lim_{n,m\to\infty} \frac{\log_2 N(n,m|\theta)}{2nm}.$$

נסמן על ידי $C_\diamond(d,k)$ את קיבול האילוץ (d,k) הדו-מימדי במודל ארבעת השכנים. באופן דומה יסמנו $C_\boxplus(d,k)$, $C_\hexagon(d,k)$ ו $C_\Delta(d,k)$ את קיבול האילוץ (d,k) הדו-מימדי במודל שמונת השכנים, במודל המשושים ובמודל המשולשים, בהתאמה.

עבודות קודמות

במודל ארבעת השכנים נחקר ערך הקיבול $C_\diamond(1,\infty)$ בעבודות רבות. Calkin ו Wilf הראו ב [3] ש

$$0.587890\ldots \le C_\diamond(1,\infty) \le 0.588339\ldots.$$

Weeks ו Blahut שיפרו חסמים אלו ב [12], והגיעו לתוצאה:

$$0.58789116177527\ldots \le C_\diamond(1,\infty) \le 0.58789149494390\ldots,$$

ואז בעזרת כלי מתמטי להאצת התכנסות של סדרות, הנקרא: "Richardson Extrapolation", הראו כי ככל הנראה הערך: $C_\diamond(1,\infty) = 0.587891161775$ הוא מדויק עבור 12 ספרות אחרי הנקודה.

עבור $d \geq 1$, Siegel ו Wolf [22] ו Halevy, Chen, Roth, Siegel ו Wolf [9], הראו חסמים עבור ערך הקיבול $C_{\diamond}(d, \infty)$ בעזרת ניתוח מקודדים בשיטת Bit-Stuffing. בנוסף, הראו גם Kato ו Zeger [13] חסמים על ערכי קיבולים אלו.

עבור $k \geq 1$, ערך הקיבול $C_{\diamond}(0, k)$ נחקר על ידי Talyansky [23], ועל ידי Kato ו Zeger [13]. עבור ערכים אחרים של d ושל k, Kato ו Zeger סיפקו ב [13] אפיון מדויק של הפרמטרים עבורם הקיבול חיובי: $C_{\diamond}(d, k) > 0$ אם ורק אם $k \geq d + 2$. הקיבול של אילוצים אסימטריים במודל זה נחקר על ידם ב [14].

במודל המשושים ידוע הערך המדויק של $C_{\bigcirc}(1, \infty)$ והוא ניתן על ידי Baxter [1]. אילוצים אחרים במודל זה נחקרו על ידי Kukorelly ו Zeger ב [15,16].

במודל המשולשים, נחקר הערך של $C_{\triangle}(1, \infty)$ ב [19] על ידי Nagy ו Zeger. הם הראו ש:

$$0.628831217\ldots \leq C_{\triangle}(1, \infty) \leq 0.634775895\ldots$$

שיטות העבודה

בעבודה זו, אנו מציגים מספר שיטות כדי לקבוע שהקיבול של אילוץ מסוים שווה לאפס או חיובי. אנו מכלילים שיטה של Blackburn [2] שתיקרא ***שיטת הסריקה***, שבעזרתה ניתן לקבוע שהקיבול של אילוץ כלשהו θ הוא אפס. שיטה זו מתבססת על סריקה סדרתית של שורות של מערך שמקיים את האילוץ θ. כל תא במערך יכול להכיל את התווית *אפס* או *אחד*, באופן שמתקיים האילוץ בכל אחד מכיווני המערך. אנו מניחים שידוע מראש מהן התוויות בתאים שנמצאים ב r_1 השורות הראשונות במערך, r_2 השורות האחרונות, t_1 העמודות הראשונות, ו t_2 העמודות האחרונות, כאשר r_1, r_2, t_1 ו t_2 הם קבועים שאינם תלויים בגודל המערך. נסמן ב δ את המספר הכולל של תאים שבהם ידועות התוויות. עתה, מונים את מספר האפשרויות בהן ניתן לרשום תוויות בשאר התאים במערך, כך שהאילוץ θ ימשיך להתקיים. Blackburn הראה עבור אילוצים מסוימים שהתווית בכל תא שנסרק נקבעת בצורה יחידה על סמך התוויות הידועות ואלו שנסרקו עד כה. מכאן שלכל אוסף אפשרי של תוויות במקומות הידועים קיימת לכל היותר אפשרות אחת לשאר התוויות במערך, ולכן מספר המעריכים האפשריים מגודל ***n*** x ***m*** שמקיימים

את האילוץ θ הוא לא יותר מאשר 2^{δ}. לפי הגדרת הקיבול מספר זה הוא קטן מספיק על מנת לקבוע ש $C(d, k) = 0$.

אנו מרחיבים שיטה זו בכך שאנו מאפשרים לתא הנסרק שלא תיקבע בו התווית ביחידות, כל עוד מתקיים אחד משני תנאים נוספים. תנאים אלו מאפשרים מצב שבו גם התווית *אפס* וגם התווית *אחד* מתאימות בתא הנסרק, אך מגבילים את שאר התוויות במערך באופן הבא. התנאי הראשון דורש שבחירת אחת משתי התוויות האפשריות תקבע ביחידות את התוויות בכל שאר התאים באותה שורה. התנאי השני דורש שכל התוויות בתאים בשורה שקדמו לתא הנסרק מהוות רצף קבוע וידוע מראש. הרחבת האפשרויות עבור התווית של התא הנסרק עדיין מגבילה את מספר המערכים האפשריים מגודל $n \times m$ שמקיימים את האילוץ θ, כך שניתן לקבוע שהקיבול הוא אפס. היתרון הוא שמתאפשר לקבוע שהקיבול שווה לאפס גם עבור אילוצים שבהם מספר המערכים שמקיימים את האילוץ הוא גדול יותר.

שיטות קודמות לקביעה שקיבול של אילוצים שווה לאפס התבססו במידה ניכרת על האילוץ הספציפי שעבורו מוכיחים. שיטות אלו היו ארוכות מאוד ודרשו חלוקה להרבה מקרים וניתוחים. כדוגמא לאפקטיביות של השיטה, אנו מציגים הוכחה למשפט של Kato ו Zeger שניתן ב [13]: $C_{\diamond}(d, d+1) = 0$ לכל $d \geq 1$. ההוכחה מתבססת על שיטת הסריקה, והיא קצרה בצורה ניכרת מההוכחה המקורית.

כדי לקבוע שקיבול הוא חיובי, אנו מגדירים צורות שאיתן ניתן לרצף את המישור. בהנתן צורה A כזו, אנו מוצאים t אפשרויות לאוסף התוויות על הצורה. אנו מראים שכל ריצוף שבו אוסף התוויות בכל עותק של הצורה יכול להיות כל אחת מתוך t האפשרויות, נותן מערך שמקיים את האילוץ θ.

זה מבטיח שמספר המערכים האפשריים מגודל $n \times m$ שמקיימים את האילוץ θ, הוא לפחות $t^{1/N}$, כאשר N הוא מספר התאים בצורה A, ומכאן נובע ש $C(d, k) \geq \frac{1}{N}\log_2 t$. בחלק מן ההוכחות אנו מראים, בנוסף לחסם הנובע מהריצוף, חסם תחתון נוסף הנובע משיקולים ספציפיים לגבי האילוץ. מקרים אלו מראים שהמרחק בין החסם הנובע מן הריצוף לבין הערך האמיתי של הקיבול, יכול להיות די גדול.

תוצאות עבור המודלים השונים

את השיטות הנ״ל אנו מיישמים על המודלים השונים במטרה לאפיין את הפרמטרים בכל מודל שבהם הקיבול הוא אפס או חיובי. במודל ארבעת השכנים, Kato ו Zeger סיפקו ב [13] אפיון מדויק של הפרמטרים עבורם הקיבול חיובי: $C_{\diamond}(d, k) > 0$ אם ורק אם $k \geq d + 2$. אנו מתמקדים באילוצים אסימטריים במודל זה. אילוצים אלו נחקרו על ידי Kato ו Zeger [14], שהשאירו מספר מקרים עבורם לא ידוע האם הקיבול הוא אפס או חיובי. עבור חלק מהמקרים אנו מראים שהקיבול הוא חיובי על ידי מציאת צורות שעבורן יש שתי אפשרויות שונות לתוויות, כך שריצוף המישור בעזרתן נותן מערכים שמקיימים את האילוץ. עבור מקרים אחרים אנו משתמשים בשיטת הסריקה על מנת להראות שהקיבול הוא אפס.
אנו מספקים אפיון כמעט מלא של הפרמטרים עבורם הקיבול הוא אפס או חיובי במודל זה. נותר אוסף פרמטרים יחיד עבורו השאלה נשארת פתוחה.

במודל המשולשים, אנו מראים ש $C_{\triangle}(d, d + 3) = 0$ לכל $d \geq 3$ על ידי שימוש בהרחבה של שיטת הסריקה. אנו מוצאים דפוס של תוויות, ומוכיחים שאם דפוס זה מופיע במערך שמקיים את האילוץ $(d, d + 3)$ אזי אותו דפוס ממשיך להופיע בצורה מחזורית על פני מספר שורות במערך. לאחר מכן אנו מבצעים את שיטת הסריקה, ומתקבל שלא תמיד נקבעת התווית בתא הנסרק בצורה יחידה. אך כאשר היא לא נקבעת, מתקיים שהדפוס המתואר מופיע, ובעזרת המחזוריות שהוא משרה על התוויות מובטח שאנו נמצאים באחד משני המצבים הנוספים ששיטת הסריקה המורחבת מאפשרת. כך מובטח שמספר המערכים שמקיימים את האילוץ קטן מספיק על מנת לקבוע שקיבול האילוץ שווה לאפס.

עבור $d \geq 5$ שמקיים $d \equiv 1 (\mathrm{mod}\ 4)$, אנו מראים שהקיבול הוא חיובי על ידי מציאת צורות שעבורן יש מספר אפשרויות שונות לתוויות, כך שריצוף המישור בעזרתן נותן מערכים שמקיימים את האילוץ. תוצאה זו משרה אפיון הדוק עבור $d \geq 5$ שמקיים $d \equiv 1 (\mathrm{mod}\ 4)$: $C_{\triangle}(d, k) > 0$ אם ורק אם $k \geq d + 4$. ביחד עם התוצאה הקודמת מתקבל שעבור שאר הערכים של d הפערים בין האיזורים הידועים של קיבול אפס וחיובי הינם יחסית קטנים.

לבסוף, במודל שמונת השכנים אנו מראים ש $C_{\boxplus}(d, d + 3) = 0$ לכל $d \geq 3$ על ידי שימוש בשיטת הסריקה.