Automated Theorem Proving for Hexagonal Run Length Constrained Capacity Computation

Zsolt Kukorelly
Switzerland

Kenneth Zeger
Dept. of Electrical and Computer Eng.
University of California, San Diego
La Jolla, CA 92093-0407
Email: zeger@ucsd.edu

***Abstract*— An automated theorem proving technique is developed and is used to show that the capacity of the hexagonal (d, k) constraint is zero whenever $k = d + 3$ for $d = 3, 4, 5, 7, 9, 11$.**

I. Introduction

For integers $k \geq d \geq 0$, a binary sequence *satisfies the* (d, k) *constraint* if two consecutive ones are separated by at least d and at most k zeros. A two-dimensional binary pattern arranged in a rectangle satisfies the (d, k) constraint if it satisfies the one-dimensional (d, k) constraint both horizontally and vertically. In general, for a two-dimensional constraint S, the *capacity of* S is defined by

$$C(S) = \lim_{n,m\to\infty} \frac{\log_2(N_S(n, m))}{mn} \tag{1}$$

where $N_S(n, m)$ is the number of $n \times m$-rectangles satisfying S. This is asymptotically the quantity of information given by a binary digit in a two-dimensional rectangle satisfying S.

In our setting, a codeword is a function $f : \mathbb{Z}^2 \longrightarrow \{0, 1\}$, i.e., a labeling of $\mathbb{Z}^2$ with zeros and ones. A *code* is a set of codewords. A (d, k)*-constrained code* is a code whose codewords all satisfy the (d, k) constraint.

The capacity of two-dimensional (d, k) constraints has been investigated by many several authors (e.g. [2], [4], [6]) and efficient coding methods have been proposed. In all of these cases, the code bits lie on a square lattice. However, by using a hexagonal lattice, it is possible to pack more code bits per area unit. As for square lattices, we can define a (d, k) constraint on a hexagonal lattice, in which case a 1 must be surrounded by at least d and at most k zeros in six directions.

Denote the capacity of a (d, k)-constraint for the square lattice by $C_{sq}(d, k)$ and for the hexagonal lattice by $C_{hex}(d, k)$ In this paper, we investigate the zero capacity region of hexagonal (d, k) constraints, i.e., the set of pairs (d, k) with $C_{hex}(d, k) = 0$.

A hexagonal lattice with a (d, k) constraint is topologically the same as the square lattice with a (d, k) constraint horizontally, vertically, and on the northwest-southeast diagonal. Any codeword that satisfies the hexagonal (d, k) constraint also satisfies the usual two-dimensional (d, k) constraint. Therefore, $C_{hex}(d, k) \leq C_{sq}(d, k)$ for all d and all k. In particular, since $C_{sq}(d, d+1) = 0$ (see [4]), we also have $C_{hex}(d, d+1) = 0$. Also, $C_{hex}(0, 1)$ is known exactly [1] and is positive. It is known [5] that $C_{hex}(d, d+2) = 0$ for all $d \geq 1$, and

1) For $d \geq 2$ even, $C_{hex}(d, 2d+1) > 0$;
2) For $d \geq 1$ odd, $C_{hex}(d, 2d+3) > 0$;
3) For any $d \geq 1$, $C_{hex}(d, \infty) > 0$.

Note that an implication is that $C_{hex}(2, 5) > 0$. Also, $C_{hex}(0, 3) > 0$ follows trivially from the fact that $C_{hex}(0, 1) > 0$. It can also be shown that $C_{hex}(1, 4) > 0$. It has been unknown, however, whether $C_{hex}(d, d+3)$ is zero or positive for $d \geq 3$. We note that a recent result due to Censor and Etzion [3] shows that $C_{hex}(d, d+4) = 0$ for all even $d \geq 6$ (and hence $C_{hex}(d, d+5) = 0$ for all odd $d \geq 5$).

Table I shows the published results for small d and k, including those presented in this paper and in [3].

$d\backslash k$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
0	0	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
1		0	0	0	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
2			0	0	0	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
3				0	0	0	0	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
4					0	0	0	0		+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
5						0	0	0	0		+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
6							0	0	0		+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
7								0	0	0	0		+	+	+	+	+	+	+	+	+	+	+	+	+
8									0	0	0		+	+	+	+	+	+	+	+	+	+	+	+	+
9										0	0	0	0		+	+	+	+	+	+	+	+	+	+	+
10											0	0	0		+	+	+	+	+	+	+	+	+	+	+
11												0	0	0	0		+	+	+	+	+	+	+	+	+
12													0	0	0		+	+	+	+	+	+	+	+	+
13														0	0	0			+	+	+	+	+	+	+
14															0	0	0		+	+	+	+	+	+	+
15																0	0	0			+	+	+	+	+
16																	0	0	0		+	+	+	+	+

TABLE I

Summary of the hexagonal (d, k)-constrained capacity results for small d and k. A '+' indicates positive capacity and a '0' indicates zero capacity. The blank spaces are unsolved cases.

II. Main Results

In this section, we state the result that $C_{hex}(d, d+3) = 0$ for $d = 3, 4, 5, 7, 9, 11$. It remains, however, an open problem whether $C_{hex}(d, d+3) = 0$ is true general. We give proof sketches for the cases $d = 3, 4$ due to space limitations. The basic idea is to prove that a run of exactly d zeros can never occur without violating the (d, k) hexagonal constraint, and therefore $C_{hex}(d, d+3) = C_{hex}(d+1, d+3) = 0$.

A computer program was written that helps in arriving at rigorous proofs for the case $k = d+3$. The program assumes the existence of a horizontal string 10^d1 and attempts to derive a contradiction. If a contradiction is achieved then only runs of $d+1$, $d+2$, or $d+3$ zeros in a row are possible, which is equivalent to the $(d+1, d+3)$ hexagonal constraint, which is known to have zero capacity. To arrive at a contradiction, the program makes (hopefully clever) assumptions about the values of certain positions.

For example, it may choose a particular location not yet labeled as 0 or 1 and then assume it must be labeled as 1. This assumption then leads to many consequences in the form of forced labellings of other locations (in order to avoid violating the $(d, d+3)$ constraint). A contradiction is arrived at, if for example, a run of length $d+4$ is obtained. When a contradiction is found, the converse of the most recent assumption is validated, conditioned on all of the previous assumptions. In this manner a “stack” of pushes and pops of assumptions keeps track of the state of the proof.

The choice as to which locations to make assumptions was implemented by picking locations near other “active locations”, ie. near other locations that have already been labeled. This technique seems to improve the chances of quickly finding contradictions.

The procedure uses the following facts:

1) A one forces d zeros in six directions;
2) A run of $d+3$ zeros forces two ones at its ends;
3) Two runs of zeros totaling at least $d+3$ zeros and separated by a blank space force a one into that space;
4) A run of $j \geq 4$ zeros forces on both sides, $d+4-j$ positions after the end of the run, another run of $j-3$ zeros. Indeed, if any of those positions had a one, this would produce a run of $d+4$ zeros. For instance, if $d=5$ and the locations $\{(j, 0) : \ 0 \leq j \leq 4\}$, have label 0, then $(-6, 0), (-5, 0), (9, 0)$, and $(10, 0)$ must be zero.

Although the program is limited to a finite array (typically of size 40×40), all conclusions are valid because the number of zeros and ones forced can only increase with larger arrays.

Sometimes, the automated procedure finds partial proofs or “lemmas”, in the form of conditions statements of the form: “If the label of position X is 1, then the label of position Y is 1”. These lemmas can be used by humans to construct rigorous induction arguments about the hexagonal capacities.

However, several times, the automated procedure actually created the entire theorem (i.e. $C_{hex}(d, d+3) = 0$). The formal proofs were quite long, running thousands or millions of lines in length. In fact, each “line” in the proof often required thousands of verifiable sub-steps. Thus billions of steps were performed in the longer proofs. At present, we do not see a way to obtain proofs in these cases by hand.

The main results are summarized in the following theorem.

Theorem II.1. *$C_{hex}(d, d+3) = 0$ for $d = 3, 4, 5, 7, 9, 11$.*

For the $(3, 6)$ constraint, the automated procedure let to a particularly nice and short proof, which we give below. It is difficult to realize, from reading the proof, that it was computer generated in this case.

Proof of Theorem II.1 for $d = 3$: We will prove that no codeword can contain the string 10001 along any line. Suppose to the contrary that f is a codeword such that 10001 appears horizontally from $(0, 0)$ to $(4, 0)$. We prove by a sequence of assumptions and contradictions that this is impossible.

Suppose that $f(1, 1) = 1$; this forces a certain number of zeros but does not lead to a contradiction. Suppose further that $f(2, -1) = 0$; this forces $f(-1, -1) = f(6, -1) = 1$ and this in turn implies a run of seven zeros in row two, which is not allowed. Thus, we must have $f(2, -1) = 1$. This creates zeros which force ones, which in turn force zeros, and so on. Successively, one finds that the following points must have label one: $(5, -2)$ and $(3, 2)$; $(5, 3)$, $(8, -3)$, $(1, -3)$, and $(-2, 3)$; $(-1, 4)$ and $(6, 1)$. But then this implies a horizontal run of seven zeros from $(0, 4)$ to $(6, 4)$.

Both assumptions $f(2, -1) = 0$ and $f(2, -1) = 1$ led to a contradiction and we conclude that $f(1, 1) = 1$ is impossible. Thus, $f(1, 1) = 0$. By symmetry, also $f(3, -1) = 0$.

Next, we assume that $f(2, 1) = 0$. This forces successively ones in $(-2, 1)$ and $(5, 1)$; in $(3, -2)$; and in $(2, 3)$. But this implies a diagonal run of seven zeros from $(-2, 4)$ to $(4, -2)$. Thus, we must have $f(2, 1) = 1$. By symmetry, we must also have $f(2, -1) = 1$. This is impossible.

Hence, $f(1, 1) = 1$ and $f(1, 1) = 0$ are both impossible and we conclude that our initial assumption, the run of three zeros, cannot occur. Therefore, $C_{hex}(3, 6) = C_{hex}(4, 6) = 0$. ☐

For the $(4, 7)$ constraint, the computer-generated proof is relatively long and tedious and in fact supplies only a lemma, but a particularly useful one. Due to space limitations, we omit many of the proof details.

Proof of Theorem II.1 for $d = 4$: We give here a brief sketch of the proof, emphasizing the automated contribution. Without loss of generality, we assume there is a horizontal run of four zeros surrounded by two ones spanning $(0, 0)$ to $(5, 0)$. By repeatedly pushing assumptions onto a stack and popping them off when contradictions result, we are able to conclude that, under the stated assumption, if $f(1, 1) = 1$, then $f(4, -1) = 1$ and $f(-1, -1) = 1$. The automated procedure produces the output shown below and a step-by-step illustration of it is shown in Figures 1–8.

$f(1,1) = 1$ ASSUMED
  $f(4,-1) = 0$ ASSUMED
    $f(-1,-1) = 0$ ASSUMED
    $f(-1,-1) = 1$ FORCED
  $f(4,-1) = 1$ FORCED
  $f(-1,-1) = 0$ ASSUMED
    $f(6,1) = 0$ ASSUMED
      $f(3,-2) = 0$ ASSUMED
      $f(3,-2) = 1$ FORCED
    $f(6,1) = 1$ FORCED
    $f(2,-3) = 0$ ASSUMED
    $f(2,-3) = 1$ FORCED
  $f(-1,-1) = 1$ FORCED.

By induction and symmetry arguments, we can show that two parallel diagonal lines of ones are forced through $(0,0)$ and $(5,0)$. In fact, we can show after a length argument that the codeword consists of parallel lines of ones and that one horizontal line determines the whole plane. From this, it can be shown that the original assumption of four zeros surrounded by ones is invalid. This then implies that $C_{hex}(4,7) = C_{hex}(5,7) = 0$. □

We note that for $d = 7, 9, 11$, the proofs $C_{hex}(d, d+3) = 0$ are 1490, 48682, and 1033772 lines long, respectively.

(Submitted to ISIT06 on January 16, 2006.)

REFERENCES

[1] R. J. Baxter, "Hard Hexagons: Exact Solution", J. Phys. A, vol. 13, pp. L61-L70, 1980.

[2] N. J. Calkin and H. S. Wilf, "The number of independent sets in a grid graph", *SIAM Journal on Discrete Mathematics*, vol. 11, pp. 54-60, February 1998.

[3] K. Censor and T. Etzion, "The positive capacity region of two-dimensional run length constrained channels", *IEEE International Symposium on Information Theory (ISIT)*, Seattle, WA, June 2006 (submitted).

[4] A. Kato and K. Zeger, "On the capacity of two-dimensional run length constrained channels", *IEEE Transactions on Information Theory*, vol. 45, pp. 1527-1540, July 1999.

[5] Zs. Kukorelly and K. Zeger, "The capacity of some hexagonal (d,k)-constraints", *IEEE International Symposium on Information Theory (ISIT)*, Washington, DC, p. 64, June 2001.

[6] W. Weeks IV and R. E. Blahut, "The capacity and coding gain of certain checkerboard codes", *IEEE Transactions on Information Theory*, vol. 42, pp. 1546-1548, September 1996.

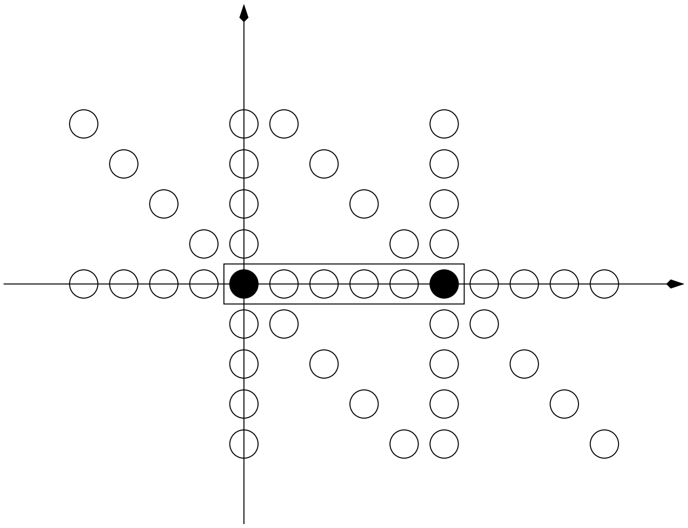

Fig. 1. First step in the proof of Theorem II.1 that $C_{hex}(4,7) = 0$. Solid dots represent ones and hollow dots represent zeros. An assumption is made that 100001 occurs horizontally from $(0,0)$ to $(5,0)$ as shown in the box. All resulting forced bits are also shown. Figures 2–8 continue the process.

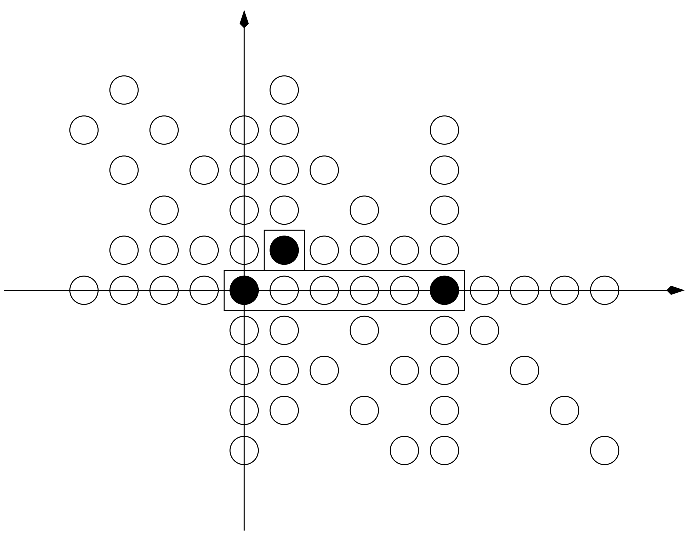

Fig. 2. The second assumption made is that $f(1,1) = 1$, as shown in the box. All resulting forced bits are also shown.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation. We thank Tuvi Eztion for providing a preprint of [3].

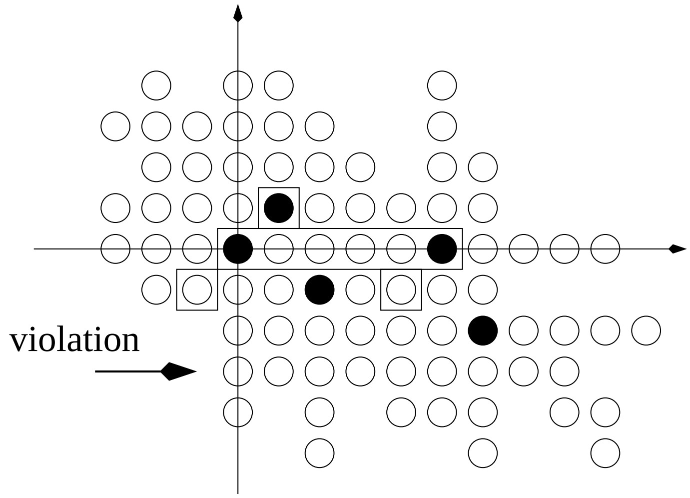

Fig. 3. The third and fourth assumptions made are that $f(4,-1) = 0$ and $f(-1,-1) = 0$, in that order, as shown in the boxes. All resulting forced bits are also shown. A horizontal string of 9 zeros occurs from $(0,-3)$ to $(8,-3)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(-1,-1) = 0$ is false, given the 3 previous assumptions before it. Thus we must have $f(-1,-1) = 1$.

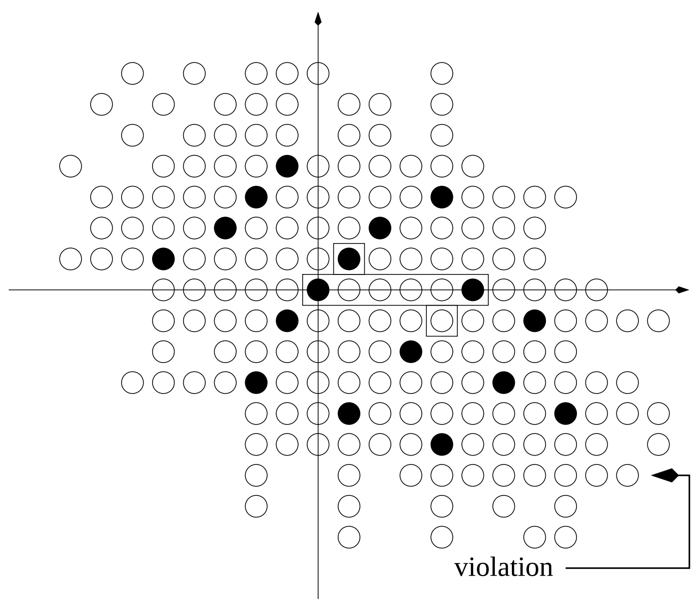

Fig. 4. We have set $f(-1,-1) = 1$, and retained the 3 existing assumptions shown in boxes. All resulting forced bits are also shown. A horizontal string of 8 zeros occurs from $(3,-6)$ to $(10,-6)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(4,-1) = 0$ is false, given the 2 previous assumptions before it. Thus we must have $f(4,-1) = 1$.

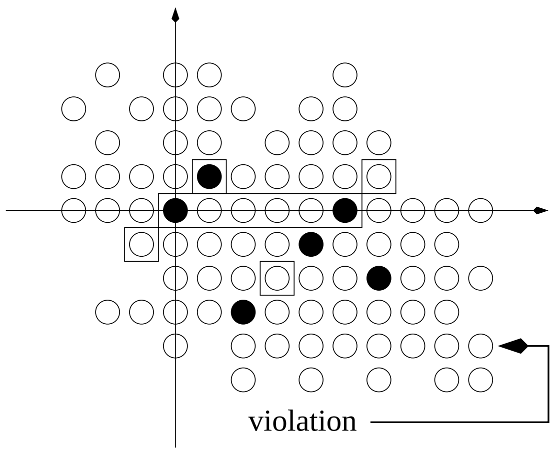

Fig. 5. We have set $f(4,-1)=1$, and retained the 2 existing assumptions shown in boxes. In addition, we have made 3 new assumptions, namely that $f(-1,-1)=0$, $f(6,1)=0$, and $f(3,-2)=0$, in that order, also shown in boxes. All resulting forced bits are also shown. A horizontal string of 8 zeros occurs from $(2,-4)$ to $(9,-4)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(3,-2)=0$ is false, given the 4 previous assumptions before it. Thus we must have $f(3,-2)=1$.

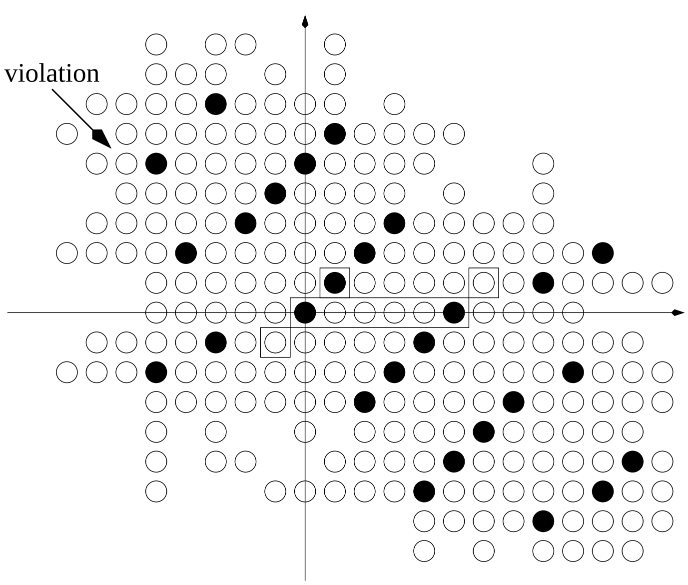

Fig. 6. We have set $f(3,-2)=1$, and retained the 4 existing assumptions shown in boxes. All resulting forced bits are also shown. A diagonal string of 8 zeros occurs from $(-6,5)$ to $(1,-2)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(6,1)=0$ is false, given the 3 previous assumptions before it. Thus we must have $f(6,1)=1$.

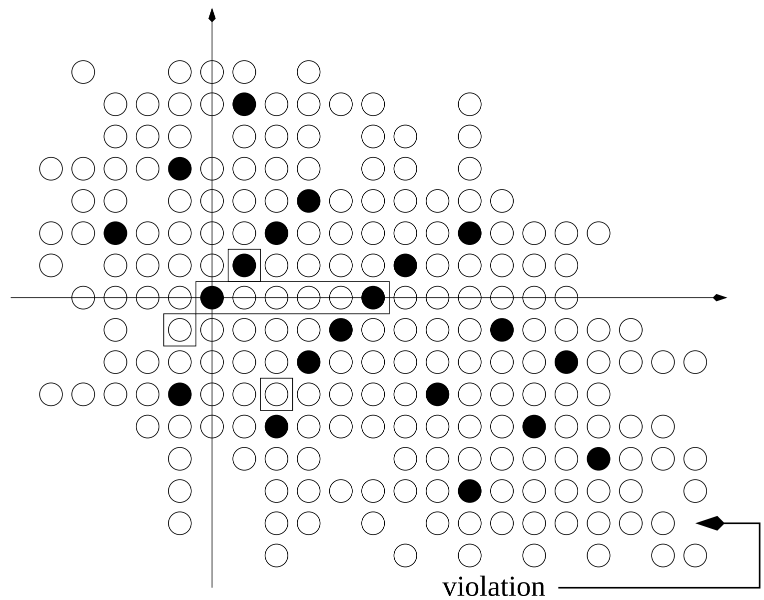

Fig. 7. We have set $f(6,1)=1$, and retained the 3 existing assumptions shown in boxes. In addition, we have made 1 new assumption, namely that $f(2,-3)=0$, also shown in the box. All resulting forced bits are also shown. A horizontal string of 8 zeros occurs from $(7,-7)$ to $(14,-7)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(2,-3)=0$ is false, given the 3 previous assumptions before it. Thus we must have $f(2,-3)=1$.

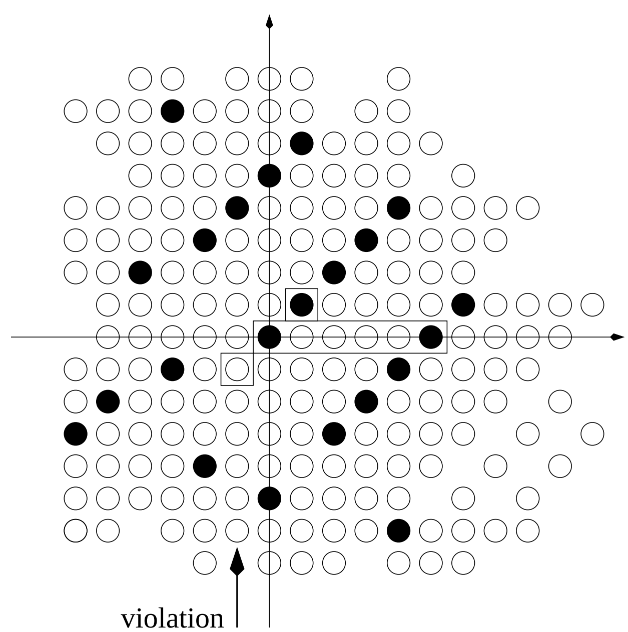

Fig. 8. We have set $f(2,-3)=1$, and retained the 3 existing assumptions shown in boxes. All resulting forced bits are also shown. A vertical string of 10 zeros occurs from $(-1,-6)$ to $(-1,3)$ (as indicated by the arrow), which violates the hexagonal $(4,7)$ constraint. Thus, the most recent assumption, namely $f(-1,-1)=0$ is false, given the 2 previous assumptions before it. Thus we must have $f(-1,-1)=1$.