Cycles Identifying Vertices and Edges in Binary Hypercubes and 2-dimensional Tori

Iiro Honkala*  
Department of Mathematics  
University of Turku  
20014 Turku, Finland  
e-mail: honkala@utu.fi

Mark G. Karpovsky  
College of Engineering  
Boston University  
Boston, MA 02215, USA  
e-mail: markkar@bu.edu

Simon Litsyn  
Department of Electrical Engineering – Systems  
Tel-Aviv University  
Ramat-Aviv 69978, Israel  
e-mail: litsyn@eng.tau.ac.il

Abstract

A set of subgraphs $C_1, C_2, \ldots, C_k$ in a graph $G$ is said to identify the vertices (resp. the edges) if the sets $\{j : v \in C_j\}$ (resp. $\{j : e \in C_j\}$) are nonempty for all the vertices $v$ (edges $e$) and no two are the same set. We consider the problem of minimizing $k$ when the subgraphs $C_i$ are required to be cycles or closed walks. The motivation comes from maintaining multiprocessor systems, and we study the cases when $G$ is the binary hypercube, or the two-dimensional $p$-ary space endowed with the Lee metric.

1 Introduction

Assume that $G$ is an undirected graph, and that each vertex (node) contains a processor and each edge represents a connection (dedicated communication link) between two processors. We want to maintain the system, and consider the situation in which at most one of the processors (or alternatively, at most one of the connecting wires between the processors) is not working. This is a reasonable assumption if the probability of error is small, or if we make checking rounds regularly enough.

*Research supported by the Academy of Finland under grant 44002.
A lot of work has been done, when the identification of malfunctioning processors is made using balls in the following way: we choose some of the processors, and each of them checks its r-neighbourhood, i.e., all the processors that are within graphic distance r, and reports YES/NO depending on whether it has detected a problem or not. Based on these YES/NO answers, we want to be able to tell the exact location of the malfunctioning processor or that all the processors are fine (under the assumption that there is at most one malfunctioning processor). This approach has been suggested in [11], and further results for typical multiprocessor architectures have been presented, for example, in [1], [2], [10], [12], [13], [14] for binary hypercubes; in [4], [5], [6], [7], [8], [9] for the square grid, king grid, triangular grid and hexagonal mesh. Binary hypercubes and 2- and 3-dimensional meshes and tori are the the most popular architectures for multiprocessors at the present time. For more on testing and diagnosis in multiprocessor architectures, and the technical background, see [3], [17], [16].

In [17], the idea of using paths instead of balls is mentioned. The current paper is the first one using this approach — although, to be precise, instead of paths, we are actually using cycles (or closed walks). The mathematical problem can then be formulated as follows. We send test messages and can route them through this network in any way we like. What is the smallest number of messages we have to send if based on which messages safely come back (i.e., the idea is that the messages are routed to eventually reach the starting point) we can tell which vertex (resp. edge) is faulty (if any)? We call these the vertex identification and edge identification problems.

We consider two variants of the problem.

We call a sequence \( v_0 \epsilon_1 v_1 \epsilon_2 \ldots \epsilon_{n} v_n \) of vertices \( v_i \) in \( G \) and edges \( \epsilon_i = (v_{i-1}, v_i) \) in \( G \) a walk. If \( v_0 = v_n \), we call it a closed walk. In the first variant we wish to find a collection of closed walks \( C_1, C_2, \ldots, C_k \) such that every vertex (resp. edge) belongs to at least one of them and, moreover, the sets \( \{ j : v \in C_j \} \) (resp. \( \{ j : \epsilon \in C_j \} \)) are all pairwise different. We denote the minimum cardinality \( k \) by \( V'(G) \) (resp. \( E'(G) \)). Since a walk may contain the same vertex and the same edge more than once, it is equivalent to require that the subgraphs \( C_1, C_2, \ldots, C_k \) are connected (instead of closed walks).

For technical reasons ([3], [17]), we would like the walks to be cycles, i.e., closed walks \( v_0 \epsilon_1 v_1 \ldots v_n \), \( n \geq 3 \), where \( v_i \neq v_j \) whenever \( i \neq j \), except that \( v_0 = v_n \). Again, the requirement is that every vertex (resp. edge) belongs to at least one of the cycles \( C_1, C_2, \ldots, C_k \) and, moreover, the sets \( \{ j : v \in C_j \} \) (resp. \( \{ j : \epsilon \in C_j \} \)) are different. We denote the minimum cardinality \( k \) in this second — and more interesting — variant by \( V(G) \) (resp. \( E(G) \)).

In Section 2 we assume that \( G \) is the binary hypercube \( \mathbb{F}_2^n \), where \( \mathbb{F}_2 = \{0, 1\} \). Its vertices are all the binary words in \( \mathbb{F}_2^n \), and edge set consists of all pairs of vertices connecting two binary words that are Hamming distance one apart. We denote by \( d(x, y) \) the Hamming distance between the vectors \( x, y \in \mathbb{F}_2^n \) and \( w(x) \) the number of ones in \( x \). We give the exact answer to the vertex identification problem (using cycles), and lower and upper bounds
differing only by the constant two in the edge identification problem (using closed walks).

In Section 3 we consider 2-dimensional tori, i.e., the p-ary space $\mathbb{Z}_p^2$ with respect to the Lee metric. In the vertex identification problem (using cycles) our lower and upper bounds differ by at most two. The same is true for the edge identification problem (using closed walks); for this problem the lower and upper bounds coincide when $p \geq 3$ is a power of two.

## 2 Binary hypercubes

In this section we assume that $G$ is the binary hypercube $I_{2^n}$ and denote $V(G)$ and $E^*(G)$ by $V(n)$ and $E^*(n)$.

For arbitrary sets, we have the following trivial identification theorem:

**Theorem 1** A collection $A_1, A_2, \ldots, A_k$ of subsets of an $s$-element set $S$ is called identifying, if for all $x \in S$ the sets $\{i : x \in A_i\}$ are nonempty and different. Given $s$, the smallest identifying collection of subsets consists of $\lceil \log_2(s+1) \rceil$ subsets.

Of course, both the vertex and edge identification problems are special cases of this problem, and for the binary hypercube with $k = 2^n$ vertices we get the lower bound

$$V(n) \geq \lceil \log_2(2^n + 1) \rceil = n + 1.$$ 

**Theorem 2** $V(n) = n + 1$ for all $n \geq 3$.

**Proof** We construct $n + 1$ cycles all starting from the all-zero vector 0. Let $C_1$ be any cycle starting from 0 which visits the all-one vector 1. The $n$ cycles $C_1, C_2, \ldots, C_n$ are constructed as follows. Given $i$, let $C_i$ be a cycle of length $2^{n-1}$ which visits exactly once all the $2^{n-1}$ points whose $i$-th coordinate equals 0; this is simply an $(n-1)$-dimensional Gray code (see, e.g., [15, p. 155]).

These $n + 1$ cycles together have the required property: a point $x$ lies in $C_i$ if and only if $x_i = 0$. The cycle $C_0$ guarantees that also the all-one vector lies in at least one cycle.

In the previous proof the cycle $C_0$ was of course only needed to satisfy the condition that all the points lie in at least one cycle; if we were allowed to leave one point outside, then the exact answer would be $n$ instead of $n + 1$.

Consider now the edge identification problem using closed walks.

We can label the edges of the binary hypercube in the following way. The edge connecting the two points

$$x = (x_1, x_2, \ldots, x_i, \ldots, x_n)$$

$$y = (x_1, x_2, \ldots, x_i + 1, \ldots, x_n)$$
is denoted by 
\[(x_1, x_2, \ldots, x_{i-1}, *, x_{i+1}, \ldots, x_n).\]
The entries \(x_j = 0\) (\(j \neq i\)) are called the zeros of the edge. The number of edges is clearly \(n2^{n-1}\).

For the edge identification problem our trivial result gives the lower bound 
\[E^*(n) \geq \lceil \log_2(n2^{n-1} + 1) \rceil.\]
It is easy to verify that the right-hand side equals \(n + \lfloor \log_2 n \rfloor\). Indeed, if \(2^k - 1 \leq n < 2^k\), then they both equal \(n + k - 1\). Consequently,
\[E^*(n) \geq n + \lfloor \log_2 n \rfloor.\]

**Theorem 3** \(n + \lfloor \log_2 n \rfloor \leq E^*(n) \leq n + \lfloor \log_2 n \rfloor + 2.\)

**Proof** In view of the previous discussion it suffices to construct \(n + k + 2\) suitable closed walks, where \(k = \lfloor \log_2 n \rfloor\).

Let \(C_i, i = 2, \ldots, n\), be a closed walk that goes through all the vertices \(x\) with \(x_i = 0\) and the edges between them.

Let \(F\) be an \((n - 1)\)-dimensional Gray code, a closed walk of length \(2^{n-1}\) which goes through all the \(2^{n-1}\) vertices \(x\) with \(x_1 = 0\) exactly once.

In the cycle \(F\) there can be edges which have exactly the same zeros (the same number of them and in the same places). However, any two such edges can be viewed as edges from a vertex \(x\) of weight \(w + 1\) to two vertices \(y\) and \(z\) of weight \(w\). By the definition of the Gray code, there are exactly two edges adjacent to \(x\) and therefore for each zero pattern there can be at most two edges of \(F\) that share that pattern. We now go through the cycle \(F\) and take the second element in every pair of two consecutive edges that share the same zero pattern. By the construction no two of the chosen edges have a common vertex. We now take \(H\) as a closed walk that contains all the edges between all the points whose first coordinate is \(0\) except the chosen ones. Constructing such a closed walk is easy: when \(n \geq 3\), it is easy to see that we can for instance move from every vertex to the all-zero vertex. Hence our graph is connected and since we are allowed to use the same edges more than once, there is no problem going through all the edges.

In particular, \(F\) and \(H\) together go through all the vertices \(x\) with \(x_1 = 0\) and all the edges between them.

Let \(S = \{1, 2, \ldots, n\}\) and \(A_1, A_2, \ldots, A_{k+1}\) be subsets of \(S\) such that the sets \(\{j : i \in A_j\}, i = 1, 2, \ldots, n\) are all different and contain at most \(k\) elements. Moreover, we require that \(\{j : 1 \in A_j\} = \emptyset\). This is possible, because \(k + 1 \geq \lceil \log_2(n + 1) \rceil\). We can take \(A_j\) to be the set of integers \(i \in S\) such that the \(j\)-th bit in the binary representation of \(i - 1\) equals one. Using each \(A_j\) we construct a closed walk \(D_j, j = 1, 2, \ldots, k + 1\). Let \(j\) be fixed, and assume that \(|A_j| = m\). For every \(x \in \mathbb{F}_2^n\), denote by \(p(x) \in \mathbb{F}_2^m\) the projection of \(x\)
to the subspace obtained by deleting the coordinates $x_i$ of $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ for which $i \notin A_i$. The walk $D_j$ first goes through all the edges between all the vertices $\mathbf{x}$ such that $p(\mathbf{x}) = 0^m$, then moves along the Gray code $F$ (here we use the fact that $1 \notin A_j$) to a vertex $\mathbf{z}$ such that $p(\mathbf{z}) = 00\ldots 01 \in \mathbb{F}_2^m$, and goes through all the edges between all the vertices $\mathbf{x}$ such that $p(\mathbf{x}) = 00\ldots 01$ and so on. All in all, the closed walk $D_j$ contains an edge $e = (x, y) \notin F$ if and only if $p(\mathbf{x}) = p(\mathbf{y})$. This means that if $\star$ in $e$ is in the $i$-th coordinate and $e \notin F$, then $e \in D_j$ if and only if $i \notin A_j$.

We show that these $n + k + 2$ closed walks $C_2, \ldots, C_n, F, H, D_1, D_2, \ldots, D_{k+1}$, have the required property.

By the construction, for every $i$, $1 \leq i \leq n$, there is a set $A_i$ not containing $i$, and therefore all the edges are present in at least one of these closed walks. Hence, if there is an edge which is not working in the hypercube, then at least one of our messages does not come back.

We can therefore assume that we know that there is exactly one edge $e = (e_1, e_2, \ldots, e_k, \star, e_{k+1}, \ldots, e_n)$ which is faulty.

Clearly, for $i = 2, 3, \ldots, n$, $e_i = 0$ (and in particular $e_i \neq \star$) if and only if $e \in C_i$. Similarly, $e_i = 0$ if and only if $e \in F$ or $e \in H$.

Hence, based on the information whether or not $e \in C_i$ for $i = 1, 2, \ldots, n$, $e \in F$ and $e \in H$, we can tell the location of all the zeros in $e$.

Next, we check whether or not $e \in F$; if so, we can tell exactly what $e$ is, because at most two of the edges in $F$ share the same zero pattern and exactly one of them is in $H$.

So, we can assume that the faulty edge is none of the edges in $F$. But this means that $e \in D_j$ if and only if the $j$-th bit in the binary representation of $k - 1$ is 0. Using all the closed walks $D_j$ we can find all the bits in the binary representation of $k - 1$. Hence we know where the zeros are in $e$ and where $\star$ is; the remaining coordinates are ones. \hfill \square

**Example 1** Consider the case $n = 5$.

First we construct the closed walk $C_2, C_3, C_4$ and $C_5$. Consider, for instance, the closed walk $C_5$. It goes through all the edges between vertices $\star\star\star\star\star\star\star\star\star\star\star\star\star\star\star\star\star\star$. These vertices and the edges between them form an Eulerian subgraph, because all the vertices have an even degree, and we take $C_5$ to be the Eulerian cycle $00000, 10000, 11000, 01000, 00000, 00100, 10100, 10000, 10010, 11010, 11000, 11100, 11010, 10110, 11110, 11100, 01100, 01110, 11110, 11010, 10100, 00110, 00100, 00100, 01000, 01010, 01100, 01100, 000000$.

The cycle $F$ is $00000, 01000, 01100, 00100, 01110, 01100, 00000, 00010, 01010, 01001, 00011, 01011, 01111, 00110, 01101, 01001, 00001$.

In the cycle $F$, the edges between $01000$ and $01100$ and between $01100$ and $00100$ have the same zero pattern, and there are three other similar pairs. To construct $H$, we take every second of each pair, i.e., the edges between $01100$ and $00100$, the one between $01110$ and $010101$, the one between $01111$ and $00111$, and finally the one between $01101$ and $01001$. The relevant graph now has eight
vertices with odd degree and is no longer Eulerian. Nevertheless, it is very easy to construct a closed walk containing all its edges (and no others).

We still need the closed walks \( D_1, D_2 \) and \( D_3 \). For instance, \( D_1 = \{5\} \). Starting from 00000, \( D_1 \) first goes through all the edges between the vertices which have 0 in the last coordinate (in the case we can use \( C_5 \), if we like), and then it moves from 00000 to 00001 along \( F \), i.e., via the points 01000, 01100, 00100, 00110, 01110, 01010. Then starting from 00001 it goes through all the edges between vertices which have 1 in the last coordinate (we can just change the last bits in \( C_5 \) to 1’s). Finally, it returns to 00001 and moves back 00000 along \( F \).

3 The case \( G = \mathbb{Z}_p^2 \) with respect to the Lee metric

In this section \( G \) is the set \( \mathbb{Z}_p^2 \) endowed with the Lee metric, i.e., two vertices \((x_1, x_2, \ldots, x_n)\) and \((y_1, y_2, \ldots, y_k)\) are adjacent if and only if \( x_j - y_j = \pm 1 \) for a unique \( j \) and \( x_i = y_i \) for all \( i \neq j \). We denote \( V(G) \), \( V^*(G) \) and \( E^*(G) \) by \( V(p, n) \), \( V^*(p, n) \) and \( E^*(p, n) \).

The case \( n = 1 \) is not difficult. In this case only the functions \( V^*(p, 1) \) and \( E^*(p, 1) \) are applicable.

**Theorem 4** \( V^*(p, 1) = E^*(p, 1) = \lceil p/2 \rceil \) for all \( p \geq 5 \).

**Proof** It is natural to denote the vertices by 1, 2, \ldots, \( p \). Then \( j \) is adjacent to \( j - 1 \) and \( j + 1 \) (modulo \( p \)).

For each of the \( p \) pairs \((1, 2), (2, 3), \ldots, (p-1, p), (p, 1)\) there has to be a closed walk separating the points, i.e., a closed walk containing exactly one of them. However, the set of vertices contained in any closed walk (or any connected subgraph) is of the form \( \{i, i+1, \ldots, j\} \) (again modulo \( p \)). Consequently this closed walk only separates the elements in two of our pairs, namely \( (i-1, i) \) and \((j, j+1)\). Hence \( V^*(p, 1) \geq p/2 \) and we have the lower bound.

If \( p \geq 6 \) is even, then we can use the walks \((1, 2, 3), (3, 4, 5), (5, 6, 7), \ldots, (p-1, p, 1)\). If \( p \geq 5 \) is odd, then we can take the walks \((1, 2, 3), (3, 4, 5), (5, 6, 7), \ldots, (p-2, p-1, p), (p, 1)\) instead.

Clearly, the problem for the edges is the same as the one for the vertices, and hence \( E^*(p, 1) = V^*(p, 1) \).

Consider the case \( n = 2 \). The graph is now a \( p \times p \) torus (cf. Figure 1; we have only drawn the vertices). For each \( i \), denote by \((1, i), (2, i), \ldots, (p, i)\) the vertices on the \( i \)-th horizontal row from the bottom. We operate on the coordinates modulo \( p \). Each vertex \((i, j)\) is adjacent to the four vertices \((i-1, j), (i+1, j), (i, j-1)\) and \((i, j+1)\). For instance, \((1, p)\) and \((1, 1)\) are adjacent.

**Theorem 5** \( 2 \log_2 p + 1 \leq V(p, 2) \leq 2 \lceil \log_2 (p+1) \rceil + 1 \) for all \( p \geq 4 \).
Proof In the construction we refer to three special cycles $D$, $E$ and $F$. The cycle $D$ begins $(1, p), (2, p), (2, p-1), \ldots$, and moves alternately one step to the right and one step down until it circles back to $(1, p)$. The cycle $E$ is obtained from $D$ by shifting the cycle down by one step, and the cycle $F$ from $E$ by shifting down by one step. Throughout the proof we illustrate the construction in the case $p = 12$; for the cycle $D$ and $F$, see Figure 1.

As the first step, take $k = \lceil \log_2 p \rceil$, and let $A$ be the $k \times p$ matrix whose $j$-th column is the binary representation of $j - 1$.

For $p = 12$, we have

$$A = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1
\end{pmatrix}. $$

From each row $A_i$ of $A$ we form a cycle $B_i$ as follows. Each row $A_i$ begins with 0. Our cycle always starts from the vertex $(1, 1) \in D$, and moves to $(1, p)$ and then to $(2, p)$. The cycle is built using the following rules:

Assume that we currently lie in $(k, p - k + 2) \in D$.

- If the $k$-th bit of $A_i$ is 0, then we take one step down and one step to the right to $(k + 1, p - k + 1) \in D$. 

Figure 1: The cycles $D$ and $F$. 


Figure 2: The cycle $B_3$.

- If the $k$-th bit of $A_i$ is 1, then we move up along the $k$-th column and circle round to the point $(k, p - k + 1)$ and take one step to the right to $(k + 1, p - k + 1) \in D$.

The cycle $B_3$ is given in Figure 2 when $p = 12$.

If $p$ is not a power of two, then there is no all 1 column in $A$, which implies that a vertex $v$ belongs to all the cycles $B_1, B_2, \ldots, B_k$ if and only if $v \in D$. If $p$ is a power of two, then we also take $D$ itself to our collection of cycles.

The idea is that apart from a belt $D$ consisting of two diagonals, we can say that a vertex $v = (x, y)$ belongs to $B_k$ if and only if the $k$-th bit in the binary representation of $x - 1$ is 1. In other words, if we know that $v \notin D$, then we can determine $x$ using the $B$-cycles.

In a similar way, we can construct $k$ more cycles $C_1, C_2, \ldots, C_k$ for determining the $y$-coordinate of $v$. For all $i = 1, 2, \ldots, k$, the cycle $C_i$ starts from $(1, p - 2) \in F$ and is built using the following rules:

Assume that we currently lie in $(k, p - k - 1) \in F$.

- If the $(p - k - 1)$-st bit of $A_i$ is 0, we move one step to the right and one down, to the vertex $(k + 1, p - k - 2)$.
- If the $(p - k - 1)$-st bit of $A_i$ is 1, we move to the left along the
(p - k - 1)-st row and circle round until we reach the vertex 
(k + 1, p - k - 1), and move one step down to (k + 1, p - k - 2).

The cycle $C_2$ is illustrated in Figure 3.

Again, if $p$ is not a power of two, a vertex $v$ belongs to all the cycles $C_1, \ldots, C_k$, if and only if $v \in \mathcal{F}$. If $p$ is a power of two, we also take $\mathcal{F}$ in our collection of cycles.

We claim that the cycles $B_1, B_2, \ldots, B_k, C_1, C_2, \ldots, C_k$ and $\mathcal{E}$, together with $\mathcal{D}$ and $\mathcal{F}$ if $p$ is a power of two, have the required property. Clearly, the number of cycles is $2 \lceil \log_2 (p + 1) \rceil + 1$.

First of all, all vertices are in at least one of our cycles. The identification of the empty set is therefore clear. It suffices to show that we can identify an unknown vertex $v = (x, y)$ based on the information, which of our cycles it belongs to.

We first decide whether or not $v \in \mathcal{D}$, and similarly whether or not $v \in \mathcal{F}$. If neither, then the $B$-cycles tell us the $x$-coordinate and the $C$-cycles the $y$-coordinate of $v$, and we are done. Assume that $v \in \mathcal{D}$. Since $\mathcal{D}$ and $\mathcal{F}$ have an empty intersection, we can use the $C$-cycles to determine the $y$-coordinate of $v$. There are only two vertices in $\mathcal{D}$ with a given $y$-coordinate, and exactly one of them belongs to $\mathcal{E}$, so we can identify $v$. In the same way, if $v \in \mathcal{F}$, then the $B$-cycles tell us the $x$-coordinate of $v$. Again the cycle $\mathcal{E}$ tells us, which one of
the two remaining vertices in $F$ with the same $x$-coordinate $v$ is.

If we want to do the identification using paths, i.e., walks not containing a vertex more than once, we can just delete an arbitrary edge from each of the cycles constructed in the previous theorem.

To identify the edges, we again consider the easier problem of using closed walks.

**Theorem 6** $|2\log_2 p| + 2 \leq E^*(p, 2) \leq 2\log_2 p + 2$, $p \geq 3$.

**Proof** The number of edges is $2p^2$, and the lower bound is clear.

Denote by $X_i$ ($Y_i$) the vertical (resp. horizontal) cycles consisting of all the edges between the vertices $(i, *)$ (resp. $(*, i)$).

Let $k = \lceil \log_2 p \rceil$.

As the first two closed walks, take $D = Y_1$ and $E = Y_1 \cup X_1 \cup X_2 \cup \ldots \cup X_p$.

Form the $k$ closed walks $B_1, B_2, \ldots, B_k$ using the rows of the matrix $A$ in the proof of Theorem 5 as follows. Let $B_i$ be the union of $X_j \cup Y_j$ over all indices $j$ such that the $j$-th entry in $A_i$ equals 1.

Together, the $k + 2$ closed walks chosen so far already contain all the edges, so the identification of the empty set is clear. We try to identify an unknown edge $e$. 

Figure 4: The closed walk $D_3$. 

10
To complete the construction, we use the rows $A_i$ to construct $k$ more closed walks $D_i$ as follows. For all $j$ such that the $j$-th entry in $A_i$ is 1 we take the cycle $(j, 1), (j, 2), (j + 1, 2), (j + 1, 3), (j + 2, 3), \ldots, (j, 1)$ as a part of $D_i$. To connect these cycles, we then take $Y_2$ (or any fixed $Y$-cycle), and reverse the status of all edges in $Y_2$; we remove all the chosen edges that belong to $Y_2$ and take as edges of $D_i$ all the edges of $Y_2$ not previously chosen. The resulting graph is clearly connected (but not a cycle in general). The closed walk $D_3$ for $p = 12$ is drawn in Figure 4.

Using $D$ and $E$, we can first tell if $e$ is a vertical or a horizontal edge. Moreover, if it is a vertical edge, we can use the closed walks $B_1, B_2, \ldots, B_k$ and determine the index $i$ for which $e \in X_i$. Similarly, if $e$ is a horizontal edge, we can find the index $i$ for which $e \in Y_i$.

Then using the closed walks $D_1, D_2, \ldots, D_k$, we can find the exact location of the edge $e$.

\begin{corollary}
If $p \geq 3$ is a power of two, then $E^*(p, 2) = 2\log_2 p + 2$.
\end{corollary}

Acknowledgment: The first author would like to thank Petri Rosendahl and Tero Laihonen for useful comments.

References


