DIMACS Technical Report 2006-02

The Paradoxical Nature of Locating Sensors in Paths and Cycles: The Case of 2-Identifying Codes

by

David L. Roberts[1]
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332 USA
`robertsd@cc.gatech.edu`

Fred S. Roberts[2]
DIMACS
Rutgers University
Piscataway, NJ 08854 USA
`froberts@dimacs.rutgers.edu`

[1]This research was performed while on appointment as a U.S. Department of Homeland Security (DHS) Fellow under the DHS Scholarship and Fellowship Program, a program administered by the Oak Ridge Institute for Science and Education (ORISE) for DHS through an interagency agreement with the U.S. Department of Energy (DOE). ORISE is managed by Oak Ridge Associated Universities under DOE contract number DE-AC05-00OR22750. All opinions expressed in this paper are the author's and do not necessarily reflect the policies and views of DHS, DOE, or ORISE.
[2]The author thanks the National Science Foundation for its support under grant EIA-0205116 to Rutgers University.

ABSTRACT

For a graph G and a set $D \subseteq V(G)$, define $N_r[x] = \{x_i \in V(G) : d(x, x_i) \leq r\}$ (where $d(x, y)$ is graph theoretic distance) and $D_r(x) = N_r[x] \cap D$. D is known as an r-identifying-code if for every vertex x, $D_r(x) \neq \emptyset$, and for every pair of vertices x and y, $x \neq y \Rightarrow D_r(x) \neq D_r(y)$. The various applications of these codes include attack sensor placement in networks and fault detection/localization in multiprocessor or distributed systems. In [2] and [16], partial results about the minimum size of D for r-identifying codes are given for paths and cycles and complete closed form solutions are presented for the case $r = 1$, based in part on [14]. We provide complete solutions for the case $r = 2$ as well as present our own solutions (verifying earlier results) to the $r = 1$ case. We use these closed form solutions to illustrate some surprisingly counterintuitive behavior that arises when the length of the path or cycle or the value of r varies.

1 Introduction

The problem of placing sensors or detectors in a network arises in many applications including homeland security, civil engineering, manufacturing, fault detection in distributed or multiprocessor systems, etc. There are several goals in sensor placement: Rapid and accurate detection of attacks, faults, or contamination of a network, minimizing the cost of sensors used, and identification of the location of an attack or fault or contamination. If we think of the network as represented by an undirected graph, the problem of sensor location to guarantee source identification has been formalized in several ways in the literature. This problem turns out to be NP-complete for general networks in its various formalizations and surprisingly complex for simple network topologies such as paths and cycles.

In [2], Bertrand, Charon, Hudry, and Lobstein study this problem for paths and cycles and provide a partial solution for detectors of varying strengths under two different formulations. The problem has been completely solved under both formulations for detectors that can detect attacks on neighbors. One purpose of this paper is to provide the complete solution under one of the formulations when detectors can detect attacks on vertices up to two steps away in the network.

In the process of solving this problem, we have found that the solution is sufficiently counterintuitive that it raises some very interesting paradoxes about sensor location. We were led to our interest in these issues by the paper [1], by Berger-Wolf, Hart, and Saia. This paper formalizes several sensor placement problems in networks represented by directed graphs. In particular, it formalizes the problem of attack detection and/or source identification with a fixed number of sensors and of detection and/or source identification within a given time limit. The authors argue that the complex goals of sensor placement require careful analysis in order to achieve the goals. The paradoxes we have uncovered underscore their point that sensor placement strategies require careful analysis by methods of computer science and mathematics since the results are sometimes counterintuitive and seemingly paradoxical.

To make our ideas precise, let $G = (V, E)$ be an undirected graph, D be a set of vertices in G at which we place detectors, and r be a positive integer. Let $N_r[x]$ be the set of all vertices in V to which there is a path of length at most r from x (so in particular $x \in N_r[x]$) and $D_r(x) = N_r[x] \cap D$. In certain literature, path length corresponds to the elapsed time before a detector is activated after an attack. In this sense, $D_r(x)$ is the set of all detectors at which an attack at x is detected in at most r time periods by some detector. We denote $D_1(x)$ by $D(x)$. We say that D is an ***r-dominating set*** in G if for every vertex x of G, $D_r(x) \neq \emptyset$, i.e., there is path of length $\leq r$ from x to some $y \in D$. (A 1-dominating set is of course a dominating set.) We say that D is an ***r-identifying code*** in G (***r-IC***) if it is an r-dominating set and if whenever $x \neq y$ are vertices, $D_r(x) \neq D_r(y)$. In an r-IC, the set of detectors activated by an attack provides a unique signature that allows us to determine where the attack took place. We shall seek the smallest d so that there is an r-IC of d vertices in G, if there is one. If so, we denote d by $M_r^I(G)$. If $r = 1$, we drop r in our terminology and speak of an identifying code or IC, and use $M^I(G)$ to mean $M_1^I(G)$. Identifying codes have been studied by many authors, starting with Karpovsky, Chakrabarty, and Levitin [19] and motivated by fault detection in multiprocessor networks. An r-identifying code enables a central controller to identify inoperable processors in a multiprocessor network. Some of the processors have "monitors" that report to the central controller if some processor within distance r is inoperable. If the monitors have been placed so that they define an r-identifying code, then the central controller can determine which processor is inoperable based on the reports by the monitors.

Note that not every graph has an r-identifying code. For instance, in the complete graph, every $D(x)$ is the set of all vertices. More generally, in any graph, if there are two vertices with the same closed neighborhood, there can be no 1-identifying code.

A closely related concept is defined as follows. We say that a set D of vertices in graph G is an $r-locating-dominating\ set$ or $r-LD\ set$ for short if for all $x \notin D$, $D_r(x) \neq \emptyset$ and for all $x, y \notin D, x \neq y$, we have $D_r(x) \neq D_r(y)$. The smallest d such that there is an r-LD set of size d is denoted by $M_r^{LD}(G)$. If $r = 1$, we speak of locating-dominating sets, LD sets and $M^{LD}(G)$. Locating-dominating sets were introduced (for $r = 1$) by Slater [23], motivated by nuclear power plant safety. $r-locating-dominating$ sets can also be used for fault detection in distributed systems. Note that in contrast to r-identifying codes, r-LD sets always exist, since the entire vertex set of a graph is an r-LD set.[1]

The literature about r-identifying codes and r-locating-dominating sets has become quite extensive. See [20] for a recent bibliography with over 100 entries. In this paper, we will limit ourselves to r-identifying codes.

One reason for our interest in paths and cycles is that paths are appropriate in applications like subway tunnels and cycles in applications like airport tram loops, to give two examples. (Note that while the trains or trams may only go one way, contaminants or pathogens can go either way through tunnels.) Many other interesting topologies have been investigated in the literature. The paper [19] studies r-identifying codes in specific topologies of interest in distributed computing, in particular binary cubes, nonbinary cubes, various meshes, and trees, and various other papers study r-identifying codes and r-LD sets for binary hypercubes, e.g., [3]. Papers [10, 12, 18] study square lattices while [8, 11, 19] study hexagonal and triangular grids. [4, 21, 23] study complete multipartite graphs and planar and outerplanar graphs.

As noted above, problems of finding optimal r-identifying sets or r-LD sets are difficult. Berger-Wolf, Hart, and Saia [1] show that for directed graphs the problem of minimizing the size of an r-identifying code is NP-complete. NP-completeness results for directed graphs for both r-LD sets and r-identifying codes were also obtained by Charon, Hudry, and Lobstein [6, 7] for both digraphs and undirected graphs, and for identifying codes by Cohen, Honkala, Lobstein, and Zemor [9] and for LD sets by Colbourn, Slater, and Stewart [13], both papers for undirected graphs.[2] By way of contrast, Slater [22] gives a linear algorithm for finding optimal LD sets in acyclic graphs, in particular trees, and Colbourn, Slater, and Stewart [13] give a linear time dynamic programming algorithm for finding optimal LD sets in series-parallel graphs.

In Section 2 we summarize the values of $M^I(G)$ for paths and cycles, that is in the case where detectors can only detect attacks one step away in a network. Section 3 finds the values

[1] A variant of an r-locating-dominating set is defined by Carson and Oellerman [5]. Let D be a set of vertices $\{v_1, v_2, \ldots, v_k\}$ in G and for each $x \notin S$, let $\vec{x}$ be the vector whose i^{th} entry is $\min\{r + 1, d(x, v_i)\}$ where $d(u, v)$ is the distance from u to v in the graph. Then we say that D is an ***r-reference-dominating set*** if for all $x \notin D, D_r(x) \neq \emptyset$ and for all $x, y \notin D, \vec{x} = \vec{y}$ iff $x = y$. If $r = 1$, a 1-reference-dominating set is the same as a 1-locating-dominating set.

[2] Carson and Oellermann [5] give similar NP-completeness results for reference-dominating sets.

$M_2^I(G)$ for paths and cycles, i.e., it analyzes the case where we have stronger detectors, ones that can detect attacks up to two steps away. In Section 4 we present some paradoxical results that follow from our theorems. Finally, Section 5 includes closing remarks.

2 Identifying Codes for Paths and Cycles

In this section, we present our own proofs of the known results for $M^I(P_n)$ and $M^I(C_n)$ where P_n is the path of n vertices and C_n is the cycle of n vertices. We label the vertices along the path or around the cycle in order as $x_1, x_2, \ldots, x_n$ and when we are dealing with a cycle, we also use addition and subtraction modulo n, so that, for example, x_{5n+4} means x_4. We shall determine $M^I(P_n)$ and $M^I(C_n)$ for every n. The results are simple, but serve as interesting contrasts to the results for 2-identifying codes given in Section 3.

Lemma 2.1. *Suppose the maximum degree of a vertex in graph G is 2, y_1, y_2, y_3, y_4 is a path in G, and D is an IC for G. Then it is not possible to have $y_1 \notin D$ and $y_4 \notin D$.*

Proof. If $y_1 \notin D, y_4 \notin D$, then $D(y_2) = D(y_3)$. □

Lemma 2.2. *Suppose the maximum degree of a vertex in graph G is 2 and D is an IC for G. Then in any induced path in G of four vertices, at least two are in D.*

Proof. Suppose y_1, y_2, y_3, y_4 is an induced path with at most one vertex in D. If on the path, only $y_1 \in D$ or only $y_4 \in D$ or no y_i is in D, then $D(y_3) = \emptyset$ or $D(y_2) = \emptyset$. The result follows by Lemma 2.1. □

Lemma 2.3. *If $n > 4$, D is an IC for a cycle C_n iff*

(1) there are no four consecutive vertices with the first and last not in D

(2) there are no three consecutive vertices none of which is in D.

Proof. Condition (2) is necessary and sufficient for the condition that $D(x) \neq \emptyset$ for all x. Necessity of condition (1) follows from Lemma 2.1. We shall now observe sufficiency of conditions (1) and (2) for the requirement that $D(x) \neq D(y), x \neq y$. Consider x_i and $x_j, i < j$, with the distance from x_i to x_j no larger in a clockwise direction around the cycle than in a counterclockwise direction. If $i + 1 \leq j \leq i + 3$, apply condition (1) to x_{i-1} and x_{i+2} to show that $D(x_i) \neq D(x_j)$. If $j > i + 3$, condition (2) implies that one of x_{i-1}, x_i, x_{i+1} is in D. □

The following theorem was proved for the case of even cycles in [2] and for odd cycles in [14]. It is also proved for odd cycles in [16]. Its analogue for LD sets was proved in [23].

Theorem 2.4. *For the cycle C_n:*

(1) $M^I(C_4) = 3, M^I(C_{2k}) = k, k \geq 3$;

(2) $M^I(C_5) = 3, M^I(C_{2k+1}) = k + 2, k \geq 3$.

Proof. Specific constructions demonstrate the upper bounds. For C_4, $\{x_2, x_3, x_4\}$ defines an IC of 3 vertices. For C_{2k}, $\{x_2, x_4, \ldots, x_{2k}\}$ is an IC of k vertices when $k \geq 3$. For C_5, $\{x_1, x_2, x_3\}$ gives a 3-vertex IC. For C_{2k+1}, $\{x_1, x_2, x_3, x_5, x_7, x_9, \ldots, x_{2k+1}\}$ gives a $(k+2)$-vertex IC if $k \geq 3$.

We now prove the lower bounds. In C_4, it is straightforward to show that 2-element sets D cannot be IC. Now consider $C_{2k}, k > 2$. If D is an IC, condition (1) of Lemma 2.3 implies that for every $i, x_i \in D$ or $x_{i+3} \in D$. (Here, as usual for cycles, addition is modulo n.) There are n such constraints and each x_i is in exactly two of them. Thus, if D has d detectors, at most $2d$ constraints are satisfied. It follows that D must have at least $\lceil \frac{n}{2} \rceil$ vertices. In particular, this shows that if $n = 2k$, any IC has at least k vertices, as desired.

We now assume that $n = 2k + 1$. By the argument above, at least $k + 1$ vertices are needed in an IC D. This proves the lower bound for C_5. Let $n > 5$. Let D be an IC. Since D has at least $k + 1$ vertices, there must be two consecutive vertices x_i, x_{i+1} in D. By condition (1) of Lemma 2.3, either $x_{i-1} \in D$ or $x_{i+2} \in D$, so we conclude that there are three consecutive vertices in D. Without loss of generality, assume $x_1, x_2, x_3 \in D$. By Lemma 2.2, each following consecutive set of four vertices has at least two vertices of D. It follows that if $2k + 1 = 4m + 3$, then D has at least $3 + 2m = k + 2$ vertices. Suppose $2k + 1 = 4m + 1$. If $x_4 \in D$ or $x_5 \in D$, then there are at least four vertices of D in the first five vertices and at least two in each subsequent four vertices, or at least $4 + 2(m - 1) = 2m + 2 = k + 2$ vertices of D. If $x_4 \notin D$ and $x_5 \notin D$, then by condition (1) of Lemma 2.3, $x_7 \in D, x_8 \in D$. Moreover, by condition (2) of Lemma 2.3, $x_6 \in D$. It follows that in the first nine vertices around the cycle, at least six are in D. In any subsequent four vertices, at least two are in D. Thus, the number of vertices of D is at least $6 + 2(m - 2) = 2m + 2 = k + 2$. □

We turn next to paths.

Lemma 2.5. *Consider a path P_n with vertices $x_1, x_2, \ldots, x_n$ in order and suppose D is an IC for P_n. Then x_3 and x_{n-2} are in D.*

Proof. If $x_3 \notin D$, then $D(x_1) = D(x_2)$ and similarly for x_{n-2}. □

Lemma 2.6. *If $n > 4$, D is an IC for a path P_n iff the following conditions hold:*

(1) there are no four consecutive vertices with the first and last not in D

(2) there are no three consecutive vertices none of which is in D

(3) $x_3, x_{n-2} \in D$

(4) x_1 or $x_2 \in D$ and x_n or $x_{n-1} \in D$.

Proof. Necessity of condition (1) follows by Lemma 2.1, of condition (2) since otherwise $D(x)$ for the middle vertex would be $\emptyset$, condition (3) by Lemma 2.5, and condition (4) because $D(x_1) \neq \emptyset, D(x_n) \neq \emptyset$. Sufficiency of the conditions follows by a proof analogous to that of Lemma 2.3. □

The following theorem was first proved in [2]. Its analogue for LD sets is due to [23].

Theorem 2.7. *For the path P_n:*

(1) $M^I(P_2)$ *is undefined,* $M^I(P_{2k}) = k+1, k \geq 2$*;*

(2) $M^I(P_{2k+1}) = k+1, k \geq 0$*.*

Proof. We know that $P_2 = K_2$ has no IC. The upper bounds are easy to establish. For P_{2k}, an IC of $k+1$ vertices is given by $\{x_2, x_3, x_4, x_6, x_8, x_{10}, ..., x_{2k}\}$. For P_{2k+1}, it is trivial that $\{x_1, x_3, x_5, \ldots, x_{2k+1}\}$ is an IC of $k+1$ detectors.

We now prove the lower bounds. Suppose $n = 2k = 4m$ and D is an IC for P_n with at most k detectors. By Lemma 2.2, every four consecutive vertices in the path has at least two vertices of D and so every four consecutive vertices includes exactly two elements of D. By conditions (1), (3) and (4) of Lemma 2.6, we note that $x_1 \in D$ or $x_4 \in D$, that $x_3 \in D$, and that $x_1 \in D$ or $x_2 \in D$. Thus, $x_1, x_3 \in D$ and $x_2, x_4 \notin D$. Then condition (1) of the lemma implies that $x_5 \in D$ and hence $x_6 \notin D$ since there can be no more than two vertices of D among x_3, x_4, x_5, x_6. Similarly, $x_7 \in D, x_8 \notin D$. Continuing this way, we conclude that $x_{n-2} \notin D$, which violates condition (3) of the lemma.

Suppose $n = 2k = 4m+2$ and D is an IC with at most k detectors. By Lemma 2.6, condition (4), either x_n or x_{n-1} is in D. It follows that each consecutive set of four vertices starting with the first four has exactly two vertices of D. But then by the construction above, $x_{n-2} \notin D$, contradicting condition (3) of the lemma.

Suppose $n = 2k+1 = 4m+1$ and D is an IC. If $x_n \in D$, then since in the first m sets of four consecutive vertices, each has at least two vertices of D, we conclude that D has at least $1 + 2m = k+1$ vertices. If $x_n \notin D$ and D has at most $k = 2m$ vertices, then by the construction above, $x_{n-1} = x_{4m}$ is not in D. But then condition (4) of Lemma 2.6 is violated.

Suppose $n = 2k+1 = 4m+3$ and D is any IC. Then by conditions (3) and (4) of Lemma 2.6, either x_n or x_{n-1} is in D and also x_{n-2} is in D, so there are at least two vertices of D in the last three. There are at least two vertices of D in each consecutive set of four vertices starting with the first four, so at least $2 + 2m = k+1$ vertices in D in all. □

3 2-Identifying Codes for Paths and Cycles

We observed in Section 2 that if two vertices in G have the same closed neighborhood, then there can be no IC. Recall that $N_2[x] = \{y : d(x,y) \leq 2\}$, where $d(x,y)$ is graph-theoretical distance. Then if there are two vertices with the same N_2, there can be no 2-IC. This shows that P_3, P_4, C_4, and C_5 have no 2-IC.

In the following, we assume that the vertices of P_n or C_n have been labeled consecutively as $x_1, x_2, \ldots, x_n$. When we are dealing with a cycle, we also use addition and subtraction modulo n as before.

Lemma 3.1. *Suppose graph G has maximum degree 2, $y_1, y_2, y_3, y_4, y_5, y_6$ is a path in G, and D is a 2-IC for G. Then it is not possible to have $y_1 \notin D$ and $y_6 \notin D$.*

Proof. If $y_1 \notin D$ and $y_6 \notin D$, then $D_2(y_3) = D_2(y_4)$. □

Lemma 3.2. *If $n > 6, D$ is a 2-IC for a cycle C_n iff*

(1) there are no six consecutive vertices with the first and last not in D

(2) there are no five consecutive vertices none of which is in D.

Proof. Condition (2) is necessary and sufficient for the condition that $D_2(x) \neq \emptyset$ for all x. Necessity of condition (1) follows from Lemma 3.1. We shall now observe sufficiency of conditions (1) and (2) for the condition $D_2(x) \neq D_2(y), x \neq y$. Consider x_i and $x_j, i < j$, with the distance from x_i to x_j no larger in a clockwise direction around the cycle than in a counterclockwise direction. To show that $D_2(x_i) \neq D_2(x_j)$ if $i+1 \leq j \leq i+5$, apply condition (1) to x_{i-2} and x_{i+3}. If $j > i+5$, apply condition (2) to $x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}$. □

Parts of the next theorem were known previously. In particular, (1) was proved by Bertrand, Charon, Hudry, and Lobstein [2], who show that the same result holds for arbitrary r. (1) is also proved by Gravier, Moncel, and Semri in [16]. Result (2)(c) follows from a more general result of [16], namely that if $2r+1$ divides n odd, then $M_r^I(C_n) = \frac{n+1}{2} + r$. For arbitrary r, [2] and [16] also provide lower bounds and [16] gives exact values for some special cases. [2] also gives results about optimal r-LD sets for cycles for arbitrary r.

Theorem 3.3. *For the cycle C_n:*

(1) $M_2^I(C_4)$ is undefined, $M_2^I(C_6) = 5, M_2^I(C_{2k}) = k$ for $k \geq 4$.

(2) $M_2^I(C_5)$ is undefined and if $k = 5p+q, q \in \{0,1,2,3,4\}$, then

- *(a) $M_2^I(C_{2k+1}) = k+2$ if $q = 0$ and $p > 0$;*
- *(b) $M_2^I(C_{2k+1}) = k+1$ if $q = 1$ and $p > 0$;*
- *(c) $M_2^I(C_{2k+1}) = k+3$ if $q = 2$ and $p > 0$;*
- *(d) $M_2^I(C_{2k+1}) = k+1$ if $q = 3$ and $p \geq 0$;*
- *(e) $M_2^I(C_{2k+1}) = k+2$ if $q = 4$ and $p > 0$, with $M_2^I(C_9) = 5$.*

Proof. We have already observed that C_4 has no 2-IC. For C_6, we note that including all but x_6 gives a 2-IC, so $M_2^I(C_6) \leq 5$. Suppose D is a 2-IC with at most four vertices. Assume without loss of generality that $x_1 \notin D$. Then using Lemma 3.2 in both the clockwise and counterclockwise direction implies that $x_2 \in D, x_6 \in D$. If $x_4 \notin D$, then $D_2(x_1) = D_2(x_4)$. Thus, either $x_3 \notin D$ or $x_5 \notin D$, without loss of generality the former. But then $D_2(x_4) = D_2(x_6)$. We conclude that $M_2^I(C_6) \geq 5$, so $M_2^I(C_6) = 5$.

Lemma 3.2 implies the constraint $x_i \in D \vee x_{i+5} \in D$ for $i = 1, 2, \ldots, n$.[3] There are n such constraints and each x_i is a term in exactly two of them. Thus, if D has d vertices, at most $2d$ such disjunctions are satisfied. It follows that D must have at least $\lceil \frac{n}{2} \rceil$ vertices. If $n = 2k$, then D must have at least k vertices, so $M_2^I(C_{2k}) \geq k$. If $n = 2k+1$, then similarly $M_2^I(C_{2k+1}) \geq k+1$. If $k \geq 4$, the set of x_i for i odd is a 2-IC for C_{2k}, so $M_2^I(C_{2k}) \leq k$, which completes the proof for C_{2k}.

We now turn to C_{2k+1}. We return to the constraints above and, for notational convenience, abbreviate x_j by j and abbreviate $x_i \in D$ or $x_j \in D$ by $i \vee j$. Choose $i \in \{1,2,3,4,5\}$. Consider the following stream of constraints, which we call ***constraint stream*** i:

$$i \vee i+1 \times 5, i+1 \times 5 \vee i+2 \times 5, \ldots, i+(g_i-1) \times 5 \vee i+g_i \times 5, i+g_i \times 5 \vee h_i,$$

[3]A similar condition for arbitrary r corresponds to the idea of a transversal in an auxiliary graph $C'_{(n,r)}$ in [16].

where

$$i + g_i \times 5 \leq 2k+1 < i + (g_i+1) \times 5 \equiv h_i(mod[2k+1]), h_i \in \{1,2,3,4,5\}$$

Suppose $k = 5p+q, q \in \{0,1,2,3,4\}$. If $k \neq 5p+2$, then $h_1 \neq 1$. Then constraint stream 1 leads into constraint stream h_1, which leads into constraint stream h_{h_1}, and so on until we hit every stream and end with the last $h_i = 1$. Putting the streams together in this order gives us the ***full constraint stream***. For example, if $k = 13$, constraint stream 3 is given by

$$3 \vee 8, 8 \vee 13, 13 \vee 18, 18 \vee 23, 23 \vee 1$$

and the full constraint stream is

$$1 \vee 6, 6 \vee 11, 11 \vee 16, 16 \vee 21, 21 \vee 26, 26 \vee 4, 4 \vee 9, 9 \vee 14, 14 \vee 19, 19 \vee 24, 24 \vee 2, 2 \vee 7, 7 \vee 12, 12 \vee 17, 17 \vee 22, 22 \vee 27,$$

$$27 \vee 5, 5 \vee 10, 10 \vee 15, 15 \vee 20, 20 \vee 25, 25 \vee 3, 3 \vee 8, 8 \vee 13, 13 \vee 18, 18 \vee 23, 23 \vee 1$$

Generalizing from $k = 13$, we note that if $k = 5p+3$, we have $2k+1 = 10p+7$, so $g_3 = 2p$ since $3 + 2p \times 5 \leq 10p+7 < 3 + (2p+1) \times 5$. If $k = 5p+2$, then $h_i = i$ and there is no full constraint stream. By way of contrast, if $k = 5p+4$, then $g_3 = 2p+1$.

We first prove (2)(b). We already know that $M_2^I(C_{2k+1}) \geq k+1$. We next show that we can find a 2-IC with $k+1$ vertices. The basic idea of the proof is that since the full stream has $2k+1$ constraints and each vertex is in exactly two constraints, then if we can satisfy all the constraints with $k+1$ vertices of D, there must be exactly one constraint where both vertices are in D and all other constraints have exactly one of their vertices in D. Which vertex in a constraint is in D is forced upon us by traversing the full stream starting from the constraint with both vertices in D. Without loss of generality this constraint is $1 \vee 1 + 1 \times 5$. Then we have the following:

- From constraint stream 1, include in D vertices 1 and $1 + z \times 5, z$ odd and 3 [a total of $p+2$ vertices since $g_1 = 2p$];
- Then, from constraint stream 3, additionally include in D vertices $3 + z \times 5, z \neq 0$ even [a total of p additional vertices since $g_3 = 2p$];
- Then, from constraint stream 5, additionally include in D vertices $5 + z \times 5, z$ odd [a total of p additional vertices since $g_5 = 2p-1$];
- Then, from constraint stream 2, additionally include in D vertices $2 + z \times 5, z$ odd, and 4 [a total of $p+1$ additional vertices since $g_2 = 2p$];
- Finally, from constraint stream 4, additionally include in D vertices $4 + z \times 5, z \neq 0$ even [a total of $p-1$ additional vertices since $g_4 = 2p-1$].

By construction, the set D satisfies all the $2k+1$ constraints and so condition (1) in Lemma 3.2 holds. In all, we have included in D

$$(p+2) + p + p + (p+1) + (p-1) = 5p+2 = k+1$$

vertices. It suffices to show that condition (2) in Lemma 3.2 holds. But this can be demonstrated as follows. Consider $1+5z, 2+5z, 3+5z, 4+5z, 5+5z, z \geq 1$. Note that if $z \geq 1$, $i+5z \in D$ iff $i+5(z+1) \notin D$. If we never have $j \in \{1,2,3,4,5\}$ such that $i+5 \in D$ iff $i \leq j$ or $i+5 \in D$ iff $i \geq j$, then condition (2) holds. In our case, we have $i+5 \in D$ iff $i = 1,2,5$. This completes the proof of (2)(b).

The proof of (2)(d) is analogous. The vertices of D are chosen as follows, starting with 1 and $1+1\times 5$:

- From stream 1: $1, 1+z\times 5, z$ odd [$p+2$ vertices];
- Then from stream 4: $4+z\times 5, z$ odd, plus 2 [$p+1$ additional vertices];
- Then from stream 2: $2+z\times 5, z \neq 0$ even, plus 5 [$p+1$ additional vertices];
- Then from stream 5: $5+z\times 5, z \neq 0$ even [p additional vertices];
- Finally from stream 3: $3+z\times 5, z$ odd [p additional vertices].

Again, D satisfies all the constraints in condition (1) of Lemma 3.2 and moreover the number of vertices in D is $5p+4 = k+1$. Condition (2) of Lemma 3.2 follows as above since $i+5 \in D$ iff $i = 1,4$.

We turn next to the proof of (2)(e). We assume that we can find a 2-IC D of $k+1$ vertices and shall reach a contradiction. As in the proof of (2)(b), once we take 1 and $1+1\times 5$ in D, the rest of the membership of D is forced upon us:

- From stream 1: 1 and $1+z\times 5, z$ odd [$p+2$ vertices];
- Then from stream 2: $2+z\times 5, z$ odd [$p+1$ additional vertices];
- Then from stream 3: $3+z\times 5, z$ odd [$p+1$ additional vertices];
- Then from stream 4: $4+z\times 5; z$ odd [$p+1$ additional vertices];
- Finally from stream 5: $5+z\times 5, z$ odd [p additional vertices].

This satisfies condition (1) of Lemma 3.2 and uses $5p+5 = k+1$ vertices. However, condition (2) of Lemma 3.2 is violated if $p > 0$, since then 11, 12, 13, 14, 15 are not in D. We conclude that $M_2^I(C_{2k+1}) \geq k+2$ if $k = 5p+4, p > 0$. If $p = 0$, the construction gives us $D = \{1,6,7,8,9\}$ and it is easy to show that this is a 2-IC of size $k+1$, so it is optimal. This shows that $M_2^I(C_9) = 5$.

To complete the proof for $k = 5p+4$, we construct a 2-IC with $k+2$ vertices. We do this by taking the following vertices for D:

- From stream 1: 1, $1+z\times 5, z$ odd [$p+2$ vertices];
- Then from stream 2: $2+z\times 5, z$ odd, and 3 [$p+2$ additional vertices];
- Then from stream 3: $3+z\times 5, z \neq 0$ even, and $3+(2p+1)\times 5$ [$p+1$ additional vertices];

- Then from stream 4: $4 + z \times 5, z$ odd [$p + 1$ additional vertices];
- Finally from stream 5: $5 + z \times 5, z$ odd [p additional vertices].

This satisfies the constraints of Condition (1) of Lemma 3.2 and uses $5p+6 = k+2$ vertices. Moreover, condition (2) of Lemma 3.2 holds since $i + 5 \in D$ iff $i = 1, 2, 4, 5$. This proves part (2)(e) of the Theorem.

The proof of (2)(a) is similar. Suppose $k = 5p, p > 0$, and there is a 2-IC D of $k + 1$ vertices. We take 1 and $1 + 1 \times 5$ in D and the rest of the membership of D is forced upon us. In particular, we get 7, 8, 9, 10, 11 not in D since $1 + z \times 5 \notin D$ for z even while for $i = 2, 3, 4, 5$, $i + z \times 5 \notin D$ for z odd. Thus, condition (2) of Lemma 3.2 is violated.

To complete the proof for $k = 5p, p > 0$, we construct a 2-IC with $k + 2$ vertices. We do this by taking the following vertices for D:

- From stream 1: 1, $1 + z \times 5, z$ odd, plus $1 + 2p \times 5$ [$p + 2$ vertices];
- Then from stream 5: $5 + z \times 5, z$ odd, plus vertex 4 [$p + 1$ additional vertices];
- Then from stream 4: $4 + z \times 5, z \neq 0$ even, plus vertex 3 [p additional vertices];
- Then from stream 3: $3 + z \times 5, z \neq 0$ even, plus vertex 2 [p additional vertices];
- Finally from stream 2: $2 + z \times 5, z \neq 0$ even [$p - 1$ additional vertices].

This satisfies the constraints of condition (1) of Lemma 3.2 and uses $5p+2 = k+2$ vertices. Moreover, condition (2) of Lemma 3.2 holds since $i + 5 \in D$ iff $i = 1, 5$. This proves part (2)(e) of the Theorem.

We now turn to Condition 2(c) and assume $k = 5p + 2$. In this case, each constraint stream begins and ends in i and the vertices in the different constraint streams are disjoint. Thus, to satisfy Condition (2) of Lemma 3.2, we need to satisfy all of the constraints in each stream i separately. Since each stream has $2p + 1$ constraints, we need at least $p + 1$ vertices from it to be put into D in order to satisfy all constraints. Thus, we need at least $5(p+1) = k+3$ vertices in D, which shows that $M_2^I(C_{2k+1}) \geq k+3$ in this case. It remains to prove that we can find a 2-IC D with $k+3$ vertices. We do this as follows: For $i = 1, 3, 5$, choose i and $i+z\times 5$ for z odd; for $i = 2, 4$, choose i and $i+z\times 5$ for z even. This satisfies all of the constraints and so condition (1) of Lemma 3.2 holds. It uses $5(p+1) = k+3$ vertices. Finally, condition (2) of Lemma 3.2 holds since $i + 5 \in D$ iff $i = 1, 3, 5$. This completes the proof of Theorem 3.3. □

We turn now to paths. The next lemma will help us to calculate $M_2^I(P_n)$.

Lemma 3.4. *Consider a path P_n with vertices $x_1, x_2, \ldots, x_n$ in order and suppose D is a 2-IC for P_n. Then:*

(1) $x_4 \in D$ and $x_{n-3} \in D$

(2) $x_5 \in D$ and $x_{n-4} \in D$

Proof. If $x_4 \notin D$, then $D_2(x_1) = D_2(x_2)$, and similarly for x_{n-3}. If $x_5 \notin D$, then $D_2(x_2) = D_2(x_3)$, and similarly for x_{n-4}. □

We now have:

Lemma 3.5. *If $n > 6, D$ is a 2-IC for a path P_n iff the following conditions hold:*

(1) there are no six consecutive vertices with the first and last not in D

(2) there are no five consecutive vertices none of which is in D

(3) $x_4, x_5, x_{n-3}, x_{n-4} \in D$

(4) x_1, x_2, or $x_3 \in D$ and x_n, x_{n-1}, or $x_{n-2} \in D$.

Proof. Necessity of condition (1) follows by Lemma 3.1, of condition (2) since otherwise $D_2(x) = \emptyset$ for the middle vertex x, of condition (3) by Lemma 3.4, and of condition (4) because $D_2(x_1) \neq \emptyset, D_2(x_n) \neq \emptyset$. Sufficiency follows by a proof analogous to that of Lemma 3.2. □

Lemma 3.5 allows us to proceed for a path P_n much as we did with cycles. Constraint streams are again the focus of our argument but since we have paths and not cycles, we modify the definition of constraint stream i to omit the last disjunction $i + g_i \times 5 \vee h_i$. We will consider the cases $n = 5p+q, q \in \{0, 1, 2, 3, 4\}$. For example, if $n = 5p+1$, the constraint stream 3 is given by

$$3 \vee 3 + 1 \times 5, 3 + 1 \times 5 \vee 3 + 2 \times 5, \ldots, 3 + (p-2) \times 5 \vee 3 + (p-1) \times 5$$

Since the constraint streams have disjoint sets of vertices, we are in a situation analogous to that of cycles in the case $k = 5p + 2$ where we considered satisfying the constraints in each stream separately.

We summarize the results in the following theorem. The result in case (2) for p even is proven in [2], where lower bounds are also given for M_r^I for arbitrary r that in fact match the exact results given in the theorem in cases (1), (2), and (3) for p even and cases (4) and (5) for p odd. [2] also gives results for r-LD sets for paths. The authors credit I. Honkala with obtaining the exact values for $M_r^{LD}(P_n)$ when $r = 2$.[4]

Theorem 3.6. *Let $n = 5p + q, q \in \{0, 1, 2, 3, 4\}$.*

(1) If $q = 0, p \geq 1$, then $M_2^I(P_n) = \frac{5p}{2} + 1$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 4$ if p is odd.

(2) If $q = 1, p \geq 1$ then $M_2^I(P_n) = \frac{5p}{2} + 1$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ if p is odd.

(3) If $q = 2, p \geq 1$, then $M_2^I(P_n) = \frac{5p}{2} + 2$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ if p is odd. Also, $M_2^I(P_2)$ is undefined.

[4]Let $M_r^{RD}(G)$ be the size of the smallest r-reference-dominating set in G (as defined in an earlier footnote). The exact values of $M_r^{RD}(P_n)$ are calculated in [5].

(4) If $q = 3, p \geq 1$, then $M_2^I(P_n) = \frac{5p}{2} + 3$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ if p is odd. Also, $M_2^I(P_3)$ is undefined.

(5) If $q = 4, p \geq 1$, then $M_2^I(P_n) = \frac{5p}{2} + 3$ if p is even, $M_2^I(P_n) = \frac{5(p-1)}{2} + 5$ if p is odd. Also, $M_2^I(P_4)$ is undefined.

Proof. As in the proof of Theorem 3.3, we use i as an abbreviation for vertex x_i.

(1) If $i \in \{1, 2, 3, 4, 5\}$, the constraint stream i is given as follows:

$$i \vee i+1\times 5, i+1\times 5 \vee i+2\times 5, \ldots, i+(p-2)\times 5 \vee i+(p-1)\times 5,$$

with $p-1$ constraints. By Lemma 3.5, condition (3), we know that $4, 5, 5p-3, 5p-4 \in D$. This tells us that the first constraint in streams 4 and 5 is satisfied and the last constraint in streams 1 and 2 by choosing detectors at these locations. To satisfy Condition (4) of Lemma 3.5, there are three possible cases: (1A) choose 3 and $5p-2$ from stream 3; (1B) choose 1 from stream 1 and $5p-2$ from stream 3, 2 from stream 2 and $5p-2$ from stream 3, 3 from stream 3 and $5p-1$ from stream 4, 3 from stream 3 and $5p$ from stream 5; (1C) choose 1 from stream 1 and $5p-1$ from stream 4, 1 from stream 1 and $5p$ from stream 5, 2 from stream 2 and $5p-1$ from stream 4, 2 from stream 2 and $5p$ from stream 5. (We lump these cases because counting number of detectors needed is the same in each of the situations in each case.)

Consider first case (1A). We need to satisfy all of the constraints in stream 1 and we have already placed $5p-4$ in the detector set, satisfying the last constraint. There are $p-2$ remaining constraints, none containing $5p-4$, so we need at least $\lceil \frac{p-2}{2} \rceil$ detectors to satisfy the remaining constraints. Turning to stream 2, since we have already placed $5p-3$ in the detector set, satisfying the last constraint, and since $5p-3$ does not appear in other constraints in this stream, the number of remaining constraints once again require at least $\lceil \frac{p-2}{2} \rceil$ detectors. The same number at least is required for streams 4 and 5. In stream 3, we need to use detectors 3 and $5p-2$, thus satisfying the first and last constraints, and at least $\lceil \frac{p-3}{2} \rceil$ detectors are needed to satisfy the remaining $p-3$ constraints, none of which contain either 3 or $5p-2$. It follows that we need at least

$$4 + 2 + \left\lceil \frac{p-3}{2} \right\rceil + 4 \times \left\lceil \frac{p-2}{2} \right\rceil$$

detectors in all. If p is even, this number is $\frac{5p}{2} + 1$, whereas if p is odd, the number is $\frac{5(p-1)}{2} + 5$.

We need to do a similar analysis in Case (1B). We consider the situation where we choose 1 from stream 1 and $5p-2$ from stream 3. After choosing 1, 4, 5, $5p-2, 5p-3, 5p-4$ we need at least $\lceil \frac{p-3}{2} \rceil$ detectors to satisfy the remaining $p-3$ constraints in stream 1, and, in each of the other streams, at least $\lceil \frac{p-2}{2} \rceil$ detectors to satisfy the remaining $p-2$ constraints. Thus, we need at least $\frac{5p}{2} + 1$ detectors if p is even and at least $\frac{5(p-1)}{2} + 5$ if p is odd.

Last, we consider Case (1C) and suppose we choose 1 from stream 1 and $5p-1$ from stream 4. After choosing 1, 4, 5, $5p-1, 5p-3, 5p-4$, to satisfy the remaining constraints, we need at least $\lceil\frac{p-3}{2}\rceil$ for stream 1, at least $\lceil\frac{p-2}{2}\rceil$ for stream 2, at least $\lceil\frac{p-1}{2}\rceil$ for stream 3, at least $\lceil\frac{p-3}{2}\rceil$ for stream 4, and at least $\lceil\frac{p-2}{2}\rceil$ for stream 5. Thus, the number of detectors needed in all is $\frac{5p}{2}+2$ if p is even and $\frac{5(p-1)}{2}+4$ if p is odd.

Finally, comparing the required minimum number of detectors in all three cases, we see that when p is even, the minimum is $\frac{5p}{2}+1$, which is achieved in both cases (1A) and (1B), and when p is odd, the minimum is $\frac{5(p-1)}{2}+4$, which is achieved in Case (1C) only. This establishes these values as lower bounds on $M_2^I(P_n)$ in the case $n=5p$.

We next establish these values as upper bounds. Consider first the case where p is even. We use either Case (1A) or Case (1B), which are the cases where the optimal lower bound is achieved. The lower bounds were calculated under the assumption that we minimize the number of detectors needed to satisfy all constraints in each stream. We now simply use algorithms similar to those used in the proof of Theorem 3.3 to choose detectors to satisfy each constraint stream and achieve the minimums. We use the first or last vertex in a stream as required and remove the corresponding first or last constraints. We then proceed left to right among remaining constraints, always choosing the largest vertex in the first unsatisfied constraint. It turns out that this method does not work for Case (1A), but does for Case (1B). Consider the situation where we use 1 from stream 1 and $5p-2$ from stream 3. In stream 1, we choose vertices 1 and $5p-4$ for the detector set D and then the algorithm adds vertices $1+z\times 5, z\neq 0$ even. In stream 2, we choose vertex $5p-3$ and the algorithm adds vertices $2+z\times 5, z$ odd. In stream 3, we choose vertex $5p-2$, as well as vertices $3+z\times 5, z$ odd. In stream 4 we choose vertices 4 and $4+z\times 5, z\neq 0$ even. Finally, in stream 5, we choose vertex 5 and vertices $5+z\times 5, z\neq 0$ even. We now use Lemma 3.5 to verify that this defines a 2-IC. Conditions (1), (3), and (4) of the lemma are satisfied by construction. Condition (2) follows as in the proof of Theorem 3.3 by observing that $i+5\in D$ iff $i=2,3$. This completes the proof in the case that p is even.

Next, suppose p is odd. Here, we use case (1C) and suppose we choose 1 from stream 1 and $5p-1$ from stream 4. In stream 1, we use 1 and $5p-4$, and the algorithm gives us in addition $1+z\times 5, z\neq 0$ even. In stream 2, we use $5p-3$ and the algorithm gives us in addition $2+z\times 5, z$ odd. In stream 3, the algorithm gives us $3+z\times 5, z$ odd. In stream 4, we use 4 and $5p-1$ and in addition $4+z\times 5, z\neq 0$ even. In stream 5, we use 5 and $5p$ and in addition $5+z\times 5, z\neq 0$ even. Condition (4) of Lemma 3.5 holds $i+5\in D$ iff $i=2,3$. This completes the proof of part (1) of the theorem.

The proofs of parts (2) through (5) are analogous and we leave the details to the reader. We simply include below the instructions for how to achieve the optimal detector assignment in each case.

(2) When p is even, the optimal detector assignment occurs when we use stream 1 to satisfy condition (4) of Lemma 3.5 for both front and back, i.e., we include vertices 1 and $1+5r$ in D. The optimal assignment is obtained by including 1, 4, 5, $1+5z, 4+5z, 5+5z$ for $z\neq 0$ even, and $2+5z, 3+5z$ for z odd. When p is odd, an optimal assignment

is obtained by using streams 2 and 5 to satisfy condition (4) of Lemma 3.5. The corresponding optimal assignment is obtained by including 2, 4, 5, $3+5(p-1) = 5p-2$, and also $2+5z, 4+5z, 5+5z$ for $z \neq 0$ even and $1+5z, 3+5z$ for z odd.

(3) When p is even, the optimal detector assignment occurs when we use stream 1 to satisfy condition (4) of Lemma 3.5 for both front and back, i.e., we include vertices 1 and $1+5p$ in D. The optimal assignment is obtained by including 1, 4, 5, $4+5(p-1) = 5p-1$ and also $1+5z, 4+5z, 5+5z$ for $z \neq 0$ even, and $2+5z, 3+5z$ for z odd. When p is odd, an optimal assignment is obtained by using streams 2 and 5 to satisfy condition (4) of Lemma 3.5. The corresponding optimal assignment is obtained by including 2, 3, 4, 5, and also $2+5z, 4+5z, 5+5z$ for $z \neq 0$ even and $1+5z, 3+5z$ for z odd.

(4) When p is even, the optimal detector assignment occurs when we use stream 1 to satisfy condition (4) of Lemma 3.5 for both front and back, i.e., we include vertices 1 and $1+5p$ in D. The optimal assignment is obtained by including 1, 4, 5, $4+5(p-1) = 5p-1, 5+5(p-1) = 5p$ and also $1+5z, 4+5z, 5+5z$ for $z \neq 0$ even, and $2+5z, 3+5z$ for z odd. When p is odd, an optimal assignment is obtained by using streams 1 and 2 to satisfy condition (4) of Lemma 3.5. The corresponding optimal assignment is obtained by including 1, 4, 5, and also $1+5z, 4+5z, 5+5z$ for $z \neq 0$ even and $2+5z, 3+5z$ for z odd.

(5) When p is even, the optimal detector assignment occurs when we use streams 1 and 4 to satisfy condition (4) of Lemma 3.5, i.e., we include vertices 1 and $4+5p$ in D. The optimal assignment is obtained by including 1, 4, 5, $5+5(p-1) = 5p$, and also $1+5z, 4+5z, 5+5z$ for $z \neq 0$ even, and $2+5z, 3+5z$ for z odd. When p is odd, an optimal assignment is obtained by using streams 2 and 3 to satisfy condition (4) of Lemma 3.5. The corresponding optimal assignment is obtained by including 2, 4, 5, and also $2+5z, 4+5z, 5+5z$ for $z \neq 0$ even and $1+5z, 3+5z$ for z odd. □

4 Paradoxes

Some values obtained from Theorems 2.4, 2.7, 3.3, and 3.6 are given in the Appendix. From the Appendix and the theorems, one can verify that the following paradoxical, counterintuitive results hold.

Paradox 1. *For $k > 2, M^I(C_{2k+1}) > M^I(C_{2k+2})$.*

This result is paradoxical since we take the same topology and make it longer but require fewer detectors. For instance, $M^I(C_7) > M^I(C_8)$. There are, of course, other examples in the literature where the parity (odd vs. even) creates similar paradoxes. The next paradox is a bit more unusual:

Paradox 2. *For $p \geq 1, M_2^I(C_{10p+5}) > M^I(C_{10p+5})$.*

This result shows that for the same topology, using detectors with greater range can actually require more detectors for perfect discrimination! For instance, $M_2^I(C_{15}) > M^I(C_{15})$. The next paradox gives a similar result for paths and shows that, for example, that $M_2^I(P_5) > M^I(P_5)$ and that $M_2^I(P_{13}) > M^I(P_{13})$.

Paradox 3. *For* $n = 5, 6, 7$ *and* $n = 10p+3, 10p+5, 10p+6, 10p+7, p \geq 1, M_2^I(P_n) > M^I(P_n)$.

Our final paradox also concerns cycles.

Paradox 4. *For* $n = 6, 11$ *and* $n = 10p + 5, 10p + 9, 10p + 11, p \geq 1, M_2^I(C_n) > M_2^I(C_{n+1})$.

Thus, for example, $M_2^I(C_{15}) > M_2^I(C_{16}), M_2^I 2(C_{19}) > M_2^I(C_{20}), M_2^I(C_{21}) > M_2^I(C_{22})$. In fact, we even have $M_2^I(C_{10p+5}) = M_2^I(C_{10p+6}) + 2, p \geq 1$. So, going to longer cycles can actually decrease the number of detectors needed by more than one.

Carson [4] has previously pointed out that increasing the range of detectors may lead us to require more detectors. He found a tree T so that $M_2^{LD}(T) = 6$ while $M_6^{LD}(T) > 6$.

5 Closing Remarks

The emphasis in this paper has been on determining exact values for $M^I(G)$ and $M_2^I(G)$ for given graphs, in particular paths and cycles. It should be noted that our results for paths and cycles are obtained by explicit constructions that yield simple algorithms that are linear in time in terms of the number of vertices. We have given results for 1-ICs and 2-ICs. It would be of interest to extend them to r-ICs for $r > 2$. It would also be of interest to completely describe $M_2^{LD}(G)$ for cycles. In [2], the exact value of $M_2^{LD}(G)$ is given for paths (the authors credit the result to I. Honkala) and it is given for cycles if $n = 6k, k \geq 1$. Other cases remain open.

In our formulation of the problem, we don't explicitly consider the time allocated for movement over an edge although it is implicitly considered to be equal for every edge. Considering the same problem for edges with weights representing possibly different times of movement over edges is also of interest. This problem is studied from an algorithmic point of view in [1].

Our formulation of the problem is only concerned with detecting attacks at single vertices. It would be useful to formulate and solve similar perfect detection discrimination problems if we allow attacks at multiple locations. This problem is studied by [15, 19] and elsewhere for identifying codes, but not for paths or cycles.

The problem with sensor failures allowed would also be of interest. Some preliminary results with one sensor failure can be found in [24] while some for multiple sensor malfunctions are in [17].

Finally, we have limited the discussion to detection problems on networks where both detectors and attacks can only take place at vertices. The problem is also of interest and can benefit from precise analysis of the type in this paper if we weaken these restrictions and allow attacks and/or detectors anywhere along an edge.

A Tables

Table 1: $M^I(C_n)$ **and** $M_2^I(C_n)$[5]

$\mathbf{n}$	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$\mathbf{M^I(C_n)}$	3	3	3	5	4	6	5	7	6	8	7	9	8	10	9	11	10	12	11	13
$\mathbf{M_2^I(C_n)}$	$\perp$	$\perp$	5	4	4	5	5	7	6	7	7	10	8	9	9	11	10	12	11	12

$\mathbf{n}$	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42
$\mathbf{M^I(C_n)}$	12	14	13	15	14	16	15	17	16	18	17	19	18	20	19	21	20	22	21
$\mathbf{M_2^I(C_n)}$	12	15	13	14	14	16	15	17	16	17	17	20	18	19	19	21	20	22	21

Table 2: $M^I(P_n)$ **and** $M_2^I(P_n)$**.**

$\mathbf{n}$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$\mathbf{M^I(P_n)}$	1	$\perp$	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10	11
$\mathbf{M_2^I(P_n)}$	$\perp$	$\perp$	$\perp$	$\perp$	4	5	5	5	5	6	6	7	8	8	9	10	10	10	10	11

$\mathbf{n}$	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
$\mathbf{M^I(P_n)}$	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18	19	19	20	20
$\mathbf{M_2^I(P_n)}$	11	12	13	13	14	15	15	15	15	16	16	17	18	18	19	20	20	20	20

References

[1] Berger-Wolf, T., Hart, W. E., and Saia, J. Discrete sensor placement problems in distribution networks. *Mathematical and Computer Modelling*. To appear.

[2] Bertrand, N., Charon, I., Hudry, O., and Lobstein, A. Identifying and locating-dominating codes on chains and cycles. *European Journal of Combinatorics 25* (2004), 969–987.

[3] Blass, U., Honkala, I., and Litsyn, S. Bounds on identifying codes. *Discrete Mathematics 241* (2001), 119–128.

[4] Carson, D. I. On generalized location-domination. In *Graph Theory, Combinatorics and Applications*, Y. Alavi and A. Schwenk, Eds. Wiley, New York, 1995, pp. 161–179.

[5] Carson, D. I., and Oellermann, O. R. A generalized reference-domination parameter. Unpublished manuscript, Department of Computer Science, University of Natal, 1995.

[6] Charon, I., Hudry, O., and Lobstein, A. Identifying and locating-dominating codes: NP-completeness results for directed graphs. *IEEE Transactions on Information Theory IT-48* (2002), 2192–2200.

[7] Charon, I., Hudry, O., and Lobstein, A. Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard. *Theoretical Computer Science 290* (2003), 2109–2120.

[5] $\perp$ means "undefined."

[8] Cohen, G., Gravier, S., Honkola, I., Lobstein, A., Mollard, M., Payan, C., and Zemor, G. Improved identifying codes for the grid. *Electronic Journal of Combinatorics 6* (1999). R19.

[9] Cohen, G., Honkala, I., Lobstein, A., and Zemor, G. On identifying codes. In *Codes and Association Schemes*, A. Barg and S. Litsyn, Eds. American Mathematical Society, Providence, RI, pp. 97–109. DIMACS Series Volume 56, 2001.

[10] Cohen, G., Honkala, I., Lobstein, A., and Zemor, G. New bounds for codes identifying vertices in graphs. *The Electronic Journal of Combinatorics 6* (1999). R14.

[11] Cohen, G., Honkala, I., Lobstein, A., and Zemor, G. Bounds for codes identifying vertices in the hexagonal grid. *SIAM Journal on Discrete Mathematics 13* (2000), 492–504.

[12] Cohen, G., Honkala, I., Lobstein, A., and Zemor, G. On codes identifying vertices in the two-dimensional square lattice with diagonals. *IEEE Transactions on Computers 50* (2001), 174–176.

[13] Colbourn, C. J., Slater, P. J., and Stewart, L. K. Locating-dominating sets in series-parallel networks. In *Proceedings of the 16th Annual Conference on Numerical Mathematics and Computing* (Winnipeg, Manitoba, 1986), pp. 135–162.

[14] Daniel, M. Codes identifiants. Rapport pour le DEA ROCO, Grenoble, June 2003.

[15] Gravier, S., and Moncel, J. Construction of codes identifying sets of vertices. *Electronic Journal of Combinatorics 12* (2005). R13.

[16] Gravier, S., Moncel, J., and Semri, A. Identifying codes of cycles. *European Journal of Combinatorics*. In press.

[17] Honkala, I., Laihonen, T., and Ranto, S. On locating-dominating codes in binary hamming spaces. *Discrete Mathematics and Theoretical Computer Science 6* (2004), 265–282.

[18] Honkala, I., and Lobstein, A. On the density of identifying codes in the square lattice. *Journal of Comb. Theory B 85* (2002), 297–306.

[19] Karpovsky, M. G., Chakrabarty, K., and Levitin, L. B. On a new class of codes for identifying vertices in graphs. *IEEE Transactions on Information Theory 44* (1998), 599–611.

[20] Lobstein, A. Codes identifiants et localisateurs-dominateurs dans les graphes: Une bibliographie. http://www.infres.enst.fr/ lobstein/bibLOCDOMetID.html, November 2005.

[21] Rall, D. F., and Slater, P. J. On location-domination numbers for certain classes of graphs. *Congr. Numer. 45* (1984), 97–106.

[22] Slater, P. J. Domination and location in acyclic graphs. *Networks 17* (1987), 55–64.

[23] Slater, P. J. Dominating and reference sets in a graph. *Journal of Math. Phys. Sci. 22* (1988), 445–455.

[24] Slater, P. J. Fault-tolerant locating-dominating sets. *Discrete Mathematics 249* (2002), 179–189.