Vertex Identifying Code in Infinite Hexagonal Grid

Gexin Yu
gyu@wm.edu

College of William and Mary
Definitions and Motivation

- **Goal**: put sensors in a network to detect which machine failed

- **Bad solution**: put a sensor on each node

- **Assumptions**:
  - Machines fail one at a time
  - Each sensor only sends one bit
  - A sensor at $v$ can see $v$ and its neighbors

- **Problem**: Find a subset $D \subset V(G)$ such that:
  - For all $v \in V(G)$, $N[v] \cap D \neq \emptyset$
  - For all $u, v \in V(G)$ if $u \neq v$ then $N[u] \cap D \neq N[v] \cap D$

- **Definition**: We call such a set $D$ a (vertex identifying) code.
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - Machines fail one at a time
  - Each sensor only sends one bit
  - A sensor at \( v \) can see \( v \) and its neighbors

- **Problem:** Find a subset \( D \subset V(G) \) s.t.
  - For all \( v \in V(G) \), \( N[v] \cap D \neq \emptyset \), and
  - \( \forall u, v \in V(G) \), if \( u \neq v \) then \( N[u] \cap D \neq N[v] \cap D \)

- **Definition:** We call such a set \( D \) a (vertex identifying) code.
Definitions and Motivation

- **Goal**: put sensors in a network to detect which machine failed

- **Bad solution**: put a sensor on each node

- **Assumptions**:
  - Machines fail one at a time
  - Each sensor only sends one bit
  - A sensor at $v$ can see $v$ and its neighbors

- Problem: Find a subset $D \subset V(G)$ s.t.

\[
\text{for all } v \in V(G), N[v] \cap D \neq \emptyset, \text{ and } \\
\forall u, v \in V(G) \text{ if } u \neq v \text{ then } N[u] \cap D \neq N[v] \cap D
\]

- Definition: We call such a set $D$ a (vertex identifying) code.
Definitions and Motivation

- **Goal**: put sensors in a network to detect which machine failed

- **Bad solution**: put a sensor on each node

- **Assumptions**:
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at \( v \) can see \( v \) and its neighbors

- **Problem**: Find a subset \( D \subseteq V(G) \) s.t.
  - \( \forall v \in V(G), N[v] \cap D \neq \emptyset \)
  - \( \forall u, v \in V(G) \), if \( u \neq v \) then \( N[u] \cap D \neq N[v] \cap D \)

- **Definition**: We call such a set \( D \) a (vertex identifying) code.

Gexin Yu  gyu@wm.edu

Vertex Identifying Code in Infinite Hexagonal Grid
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit

- Problem: Find a subset $D \subset V(G)$ s.t.

  - $\forall v \in V(G)$, $N[v] \cap D \neq \emptyset$
  - $\forall u, v \in V(G)$ if $u \neq v$ then $N[u] \cap D \neq N[v] \cap D$

- **Definition:** We call such a set $D$ a (vertex identifying) code.
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at $v$ can see $v$ and its neighbors

- Problem: Find a subset $D \subset V(G)$ s.t.
  - for all $v \in V(G)$, $N[v] \cap D \neq \emptyset$, and
  - $\forall u, v \in V(G)$ if $u \neq v$ then $N[u] \cap D \neq N[v] \cap D$

- Definition: We call such a set $D$ a (vertex identifying) code.
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at $v$ can see $v$ and its neighbors

- **Problem:** Find a subset $D \subset V(G)$ s.t.
 Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at $v$ can see $v$ and its neighbors

- **Problem:** Find a subset $D \subset V(G)$ s.t.
  - for all $v \in V(G)$, $N[v] \cap D \neq \emptyset$, and
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at \( v \) can see \( v \) and its neighbors

- **Problem:** Find a subset \( D \subset V(G) \) s.t.
  - for all \( v \in V(G) \), \( N[v] \cap D \neq \emptyset \), and
  - \( \forall u, v \in V(G) \) if \( u \neq v \) then \( N[u] \cap D \neq N[v] \cap D \)

Definition: We call such a set \( D \) a (vertex identifying) code.
Definitions and Motivation

- **Goal:** put sensors in a network to detect which machine failed

- **Bad solution:** put a sensor on each node

- **Assumptions:**
  - machines fail one at a time
  - each sensor only sends one bit
  - a sensor at $v$ can see $v$ and its neighbors

- **Problem:** Find a subset $D \subset V(G)$ s.t.
  - for all $v \in V(G)$, $N[v] \cap D \neq \emptyset$, and
  - $\forall u, v \in V(G)$ if $u \neq v$ then $N[u] \cap D \neq N[v] \cap D$

- **Definition:** We call such a set $D$ a (vertex identifying) code.
Examples: codes and non-codes

![Diagram showing vertex identifying code in an infinite hexagonal grid]

Observation: Every path $P_n$ with $n \geq 3$ has a code.

Gexin Yu  gyu@wm.edu  Vertex Identifying Code in Infinite Hexagonal Grid
Examples: codes and non-codes

1 2 3

NO!

1 2 3

NO!

1 2 3

NO!

Observation: Every path $P_n$ with $n \geq 3$ has a code.
Examples: codes and non-codes

\begin{itemize}
  \item Every path $P_n$ with $n \geq 3$ has a code.
\end{itemize}

Gexin Yu  gyu@wm.edu

Vertex Identifying Code in Infinite Hexagonal Grid
Examples: codes and non-codes

▶ Observation: Every path $P_n$ with $n \geq 3$ has a code.
Examples: codes and non-codes

- **Observation:** Every path $P_n$ with $n \geq 3$ has a code.
Not always possible

**Definition:** We call such a graph twin-free.

**New problem:** If $G$ is twin-free, find a smallest code.

We are most interested in infinite grids.
Obstacle: $N[u] = N[v]$, so for any $D$ we have $N[u] \cap D = N[v] \cap D$. 
Not always possible

- **Obstacle:** \( N[u] = N[v] \), so for any \( D \) we have \( N[u] \cap D = N[v] \cap D \).

- **Fact:** \( G \) has a code iff for all \( u \neq v \) we have \( N[u] \neq N[v] \).
Not always possible

Obstacle: \( N[u] = N[v] \), so for any \( D \) we have \( N[u] \cap D = N[v] \cap D \).

Fact: \( G \) has a code iff for all \( u \neq v \) we have \( N[u] \neq N[v] \).

Definition: We call such a graph twin-free.
Obstacle: \( N[u] = N[v] \), so for any \( D \) we have \( N[u] \cap D = N[v] \cap D \).

Fact: \( G \) has a code iff for all \( u \neq v \) we have \( N[u] \neq N[v] \).

Definition: We call such a graph twin-free.

New problem: If \( G \) is twin-free, find a smallest code.
Not always possible

- **Obstacle:** $N[u] = N[v]$, so for any $D$ we have $N[u] \cap D = N[v] \cap D$.

- **Fact:** $G$ has a code iff for all $u \neq v$ we have $N[u] \neq N[v]$.

- **Definition:** We call such a graph **twin-free**.

- **New problem:** If $G$ is twin-free, find a smallest code.

- **We are most interested in infinite grids.**
We consider infinite graphs with the following properties:

- Twin-free
- Locally finite (every vertex has finite degree)
- Vertex transitive (graph looks the same from each vertex)

Example: $V(G_{\mathbb{Z}}) = \mathbb{Z}$ and $uv \in E(G_{\mathbb{Z}})$ iff $|u - v| = 1$ (infinite path)

Definition: Rather than the smallest size code, we want the lowest density (fraction) code. We call this the density of $G$, $\tau(G)$. Question: What is $\tau(G_{\mathbb{Z}})$?
We consider infinite graphs with following properties:

- twin-free

Example: \( V(G) = \mathbb{Z} \) and \( uv \in E(G) \) iff \( |u - v| = 1 \) (infinite path).

Definition: Rather than the smallest size code, we want the lowest density (fraction) code.

We call this the density of \( G \), \( \tau(G) \).

Question: what is \( \tau(G) \)?
We consider infinite graphs with the following properties:
- twin-free
- locally finite (every vertex has finite degree)

Example: \( V(G_{\mathbb{Z}}) = \mathbb{Z} \) and \( uv \in E(G_{\mathbb{Z}}) \iff |u - v| = 1 \) (infinite path)

Definition: Rather than the smallest size code, we want the lowest density (fraction) code.
We call this the density of \( G \), \( \tau(G) \).

Question: What is \( \tau(G_{\mathbb{Z}}) \)?
We consider infinite graphs with following properties:

- twin-free
- locally finite (every vertex has finite degree)
- vertex transitive (graph looks the same from each vertex)
Infinite graphs

- We consider infinite graphs with following properties:
  - twin-free
  - locally finite (every vertex has finite degree)
  - vertex transitive (graph looks the same from each vertex)
- Ex. $V(G_Z) = \mathbb{Z}$ and $uv \in E(G_Z)$ iff $|u - v| = 1$ (infinite path)
We consider infinite graphs with following properties:
  - twin-free
  - locally finite (every vertex has finite degree)
  - vertex transitive (graph looks the same from each vertex)

Ex. \( V(G_Z) = \mathbb{Z} \) and \( uv \in E(G_Z) \) iff \( |u - v| = 1 \) (infinite path)
Infinite graphs

- We consider infinite graphs with following properties:
  - twin-free
  - locally finite (every vertex has finite degree)
  - vertex transitive (graph looks the same from each vertex)
- Ex. \( V(G_Z) = \mathbb{Z} \) and \( uv \in E(G_Z) \) iff \( |u - v| = 1 \) (infinite path)

- **Definition:** Rather than the smallest size code, we want the lowest density (fraction) code.
Infinite graphs

We consider infinite graphs with following properties:
- twin-free
- locally finite (every vertex has finite degree)
- vertex transitive (graph looks the same from each vertex)

Ex. $V(G_{\mathbb{Z}}) = \mathbb{Z}$ and $uv \in E(G_{\mathbb{Z}})$ iff $|u - v| = 1$ (infinite path)

Definition: Rather than the smallest size code, we want the lowest density (fraction) code.

We call this the density of $G$, $\tau(G)$. 
Infinite graphs

- We consider infinite graphs with following properties:
  - twin-free
  - locally finite (every vertex has finite degree)
  - vertex transitive (graph looks the same from each vertex)

- Ex. \( V(G_\mathbb{Z}) = \mathbb{Z} \) and \( uv \in E(G_\mathbb{Z}) \) iff \( |u - v| = 1 \) (infinite path)

- Definition: Rather than the smallest size code, we want the lowest density (fraction) code.
- We call this the density of \( G \), \( \tau(G) \).
- Question: what is \( \tau(G_\mathbb{Z}) \)?
Density of Square and Triangular Grids

- **Triangular Grid:** Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau = \frac{1}{4}$.
Density of Square and Triangular Grids

- **Triangular Grid:** Karpovskiy-Chakrabarty-Levitin (1998) showed that $\tau = \frac{1}{4}$.

- **Square Grid:** Cohen-Hongala-Lobstein-Zémor (2000) showed that $\tau \leq \frac{7}{20}$; and Ben-Haim-Litsyn (2005) showed that $\tau \geq \frac{7}{20}$. 
Cohen-Hongala-Lobstein-Zémor (2000) had the following constructions with density $\frac{3}{7}$:
Cohen-Hongala-Lobstein-Zémor (2000) had the following constructions with density $\frac{3}{7}$:
Cohen-Hongala-Lobstein-Zémor (2000) had the following constructions with density $\frac{3}{7}$:
Cohen-Hongala-Lobstein-Zémor (2000) had the following constructions with density $\frac{3}{7}$:
Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau \geq \frac{2}{5}$.

Cohen-Hongala-Lobstein-Zémor (2000) showed that $\tau \geq \frac{16}{39} \approx 0.4102$.

They took a finite portion of the grid, proved a lower bound for the (finite) graph, and then extended that to infinite grid.

We (with Dan Cranston, 2009) used a cake-sharing idea and proved:

Theorem: $\tau \geq \frac{12}{29} \approx 0.41379$.
Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau \geq \frac{2}{5}$.

Cohen-Hongala-Lobstein-Zémor (2000) showed that

$$\tau \geq \frac{16}{39} \approx 0.4102...$$
Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau \geq \frac{2}{5}$.

Cohen-Hongala-Lobstein-Zémor (2000) showed that

$$\tau \geq \frac{16}{39} \approx 0.4102...$$

They took a finite portion of the grid, proved a lower bound for the (finite) graph, and then extended that to infinite grid.
Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau \geq \frac{2}{5}$.

Cohen-Hongala-Lobstein-Zémor (2000) showed that

$$\tau \geq \frac{16}{39} \approx 0.4102...$$

They took a finite portion of the grid, proved a lower bound for the (finite) graph, and then extended that to infinite grid.

We (with Dan Cranston, 2009) used a cake-sharing idea and proved
Karpovsky-Chakrabarty-Levitin (1998) showed that $\tau \geq \frac{2}{5}$.

Cohen-Hongala-Lobstein-Zémor (2000) showed that

$$\tau \geq \frac{16}{39} \approx 0.4102...$$

They took a finite portion of the grid, proved a lower bound for the (finite) graph, and then extended that to infinite grid.

We (with Dan Cranston, 2009) used a cake-sharing idea and proved

**Theorem:** $\tau \geq \frac{12}{29} \approx 0.41379...$
Forget infinite for now
Forget infinite for now

Suppose that $D$ is a code for $G$. Put a cake at each $v \in D$ and redistribute so that each $u \in V(G)$ get at least $t$ cake ($0 < t < 1$).
Proving a Lower Bound (sketch)

▶ Forget infinite for now

▶ Suppose that $D$ is a code for $G$. Put a cake at each $v \in D$ and redistribute so that each $u \in V(G)$ get at least $t$ cake ($0 < t < 1$).

▶ Then $|D| \geq t|V(G)|$, or $\tau(G) = \frac{|D|}{|V(G)|} \geq t$.
Proving a Lower Bound (sketch)

- Forget infinite for now

- Suppose that $D$ is a code for $G$. Put a cake at each $v \in D$ and redistribute so that each $u \in V(G)$ get at least $t$ cake ($0 < t < 1$).

  - Then $|D| \geq t|V(G)|$, or $\tau(G) = \frac{|D|}{|V(G)|} \geq t$.

- The same idea works for infinite graphs.
Proving a Lower Bound (sketch)

- Forget infinite for now

- Suppose that $D$ is a code for $G$. Put a cake at each $v \in D$ and redistribute so that each $u \in V(G)$ get at least $t$ cake ($0 < t < 1$).

- Then $|D| \geq t|V(G)|$, or $\tau(G) = \frac{|D|}{|V(G)|} \geq t$.

- The same idea works for infinite graphs.

- **Key**: how should we share the cake?
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $2\frac{k}{5}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $2\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

- $v \not\in D$:
  - $k = 2\frac{k}{5} = 2\frac{2}{5}$.

- $v$ in a $1$-cluster:
  - $1 - 3(2\frac{k}{5}) = 2\frac{1}{5}$.

- $v$ in a $3+$-cluster:
  - $v$ has $3$ neighbors in cluster: $1 - 0 = 1$.
  - $v$ has $2$ neighbors in cluster: $1 - 1(2\frac{k}{5}) = 3\frac{3}{5}$.
  - $v$ has $1$ neighbor in cluster: $1 - 2\frac{k}{5} - 1 = 2\frac{4}{5}$. 

Gexin Yu  gyu@wm.edu
Vertex Identifying Code in Infinite Hexagonal Grid
Theorem: For the hex grid, \( \tau \geq \frac{2}{5} \).

Proof. each vertex in \( D \) gives \( \frac{2}{5k} \) cake to each neighbor not in \( D \) that has \( k \) neighbors in \( D \).
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

1. $v \not\in D$: $\frac{2}{5k} = \frac{2}{5}$.
2. $v$ in a 1-cluster: $1 - 3(\frac{2}{5}) = \frac{2}{5}$.
3. $v$ in a 3+-cluster:
   - $v$ has 3 neighbors in cluster: $1 - 0 = 1$
   - $v$ has 2 neighbors in cluster: $1 - 1(\frac{2}{5}) = \frac{3}{5}$
   - $v$ has 1 neighbor in cluster: $1 - 2(\frac{2}{5}) - \frac{1}{5} = \frac{2}{5}$

Vertex Identifying Code in Infinite Hexagonal Grid
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

- $v \notin D$: $k \frac{2}{5k} = \frac{2}{5}$.
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

- $v \notin D$: $k \cdot \frac{2}{5k} = \frac{2}{5}$.
- $v$ in a 1-cluster: $1 - 3\left(\frac{2}{5(2)}\right) = \frac{2}{5}$. 
Proving a Lower Bound (real proof)

- **Theorem:** For the hex grid, \( \tau \geq \frac{2}{5} \).

**Proof.** each vertex in \( D \) gives \( \frac{2}{5k} \) cake to each neighbor not in \( D \) that has \( k \) neighbors in \( D \).

We must show that each vertex \( v \) has at last \( \frac{2}{5} \) of a cake.

We consider cases, based on what size cluster contains \( v \):

- \( v \not\in D \): \( k \frac{2}{5k} = \frac{2}{5} \).
- \( v \) in a 1-cluster: \( 1 - 3 \left( \frac{2}{5(2)} \right) = \frac{2}{5} \).
- \( v \) in a 3\(^+\)-cluster:
**Theorem:** For the hex grid, \( \tau \geq \frac{2}{5} \).

**Proof.** Each vertex in \( D \) gives \( \frac{2}{5k} \) cake to each neighbor not in \( D \) that has \( k \) neighbors in \( D \).

We must show that each vertex \( v \) has at least \( \frac{2}{5} \) of a cake.

We consider cases, based on what size cluster contains \( v \):

- \( v \notin D \): \( k \frac{2}{5k} = \frac{2}{5} \).
- \( v \) in a 1-cluster: \( 1 - 3(\frac{2}{5(2)}) = \frac{2}{5} \).
- \( v \) in a \( 3^{+} \)-cluster:
  - \( v \) has 3 neighbors in cluster: \( 1 - 0 = 1 \).
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

- $v \notin D$: $k \frac{2}{5k} = \frac{2}{5}$.
- $v$ in a 1-cluster: $1 - 3 \left( \frac{2}{5(2)} \right) = \frac{2}{5}$.
- $v$ in a $3^+$-cluster:
  - $v$ has 3 neighbors in cluster: $1 - 0 = 1$
  - $v$ has 2 neighbors in cluster: $1 - 1 \left( \frac{2}{5} \right) = \frac{3}{5}$
Theorem: For the hex grid, $\tau \geq \frac{2}{5}$.

Proof. Each vertex in $D$ gives $\frac{2}{5k}$ cake to each neighbor not in $D$ that has $k$ neighbors in $D$.

We must show that each vertex $v$ has at least $\frac{2}{5}$ of a cake.

We consider cases, based on what size cluster contains $v$:

- $v \not\in D$: $k \cdot \frac{2}{5k} = \frac{2}{5}$.
- $v$ in a 1-cluster: $1 - 3(\frac{2}{5(2)}) = \frac{2}{5}$.
- $v$ in a $3^+$-cluster:
  - $v$ has 3 neighbors in cluster: $1 - 0 = 1$
  - $v$ has 2 neighbors in cluster: $1 - 1(\frac{2}{5}) = \frac{3}{5}$
  - $v$ has 1 neighbor in cluster: $1 - \frac{2}{5} - \frac{1}{5} = \frac{2}{5}$
Research Problems

- We now know that for infinite hexagon grid, \( \frac{12}{29} \leq \tau \leq \frac{3}{7} \). What’s the exact value of \( \tau \)?

- How about 3-dimensional grids?

- It is NP-hard to find the minimum ID-code for a given graph, even a given connected planar graph with maximum degree 4 and girth at least \( k \geq 3 \). Can we find any good bounds for such graphs?
We now know that for infinite hexagon grid, \( \frac{12}{29} \leq \tau \leq \frac{3}{7} \). What’s the exact value of \( \tau \)?

How about 3-dimensional grids?
We now know that for infinite hexagon grid, $\frac{12}{29} \leq \tau \leq \frac{3}{7}$. What's the exact value of $\tau$?

How about 3-dimensional grids?

It is NP-hard to find the minimum ID-code for a given graph, even a given connected planar graph with maximum degree 4 and girth at least $k \geq 3$.

Can we find any good bounds for such graphs?
Questions?