ZERO FORCING PROCESSES ON PROPER INTERVAL GRAPHS AND TWISTED HYPERCUBES

by

Peter Collier

Submitted in partial fulfillment of the requirements
for the degree of Master of Science

at

Dalhousie University
Halifax, Nova Scotia
April 2023

Contents

List of Tables

List of Figures

Abstract

Zero forcing is a graph infection process where a colour change rule is applied iteratively to a graph and an initial set of vertices, $S \subseteq V(G)$. If S results in the entire graph becoming infected, we call this set a zero forcing set. The size of the smallest zero forcing set for a graph, G, is called the zero forcing number of G. We study subgraphs of proper interval graphs to determine how the removal of edges affects the zero forcing number of these graphs. We, then, compare the zero forcing number of twisted hypercubes to that of the same size hypercube, and determine that twisted hypercubes have smaller zero forcing number. Finally, we turn our attention to probabilistic zero forcing, a variant on zero forcing, and show that there are graphs who become forced faster when initiating the process from vertices that are outside the center of the graph.

Acknowledgements

I would like to thank my supervisor, Dr. Jeannette Janssen, for introducing me to the concepts discussed within. This would not have been possible without your support and guidance. I would also like to thank Dr. Jason Brown and Dr. Nancy Clarke for reading this thesis.

Chapter 1

Introduction

Recently, the world has become especially interested in the study of how diseases and infections spread. We are able to model this spread mathematically by using Graph Theory. The particular type of graph infection I will be studying is known as Zero Forcing.

In zero forcing, an initial set of nodes in a network are infected. These infected nodes can spread the infection to neighbouring nodes under certain conditions. If an initial set of nodes are able to infect the entire network, then the initial set is called a zero forcing set.

Zero forcing had previously been used to bound certain algebraic properties of graphs, but was first studied in its own right in a 2007 paper by the AIM Minimum Rank-Special Graphs Work Group [1]. As others became interested in this new graph parameter [14, 8, 13], various new applications arose, including to inverse eigenvalue problems [18], PMU placement problems [5], and quantum control problems [6].

With these applications, more information on zero forcing became of interest. This gave rise to the zero forcing polynomial in [4]. This is a graph polynomial whose coefficients are related to the number of zero forcing sets of the given graph. Relationships to other well-known graph parameters have also been studied, such as the chromatic number in [21].

There also came the introduction of variations on, what is now known as, classical zero forcing. Among these are k-forcing [2], and Leaky Forcing [9]. The variation that I will be studying is Probabilistic Zero Forcing, introduced by Kang and Yi in their short paper [17]. The appeal of this model of zero forcing is that it can more closely simulate a potential real-world model of infection spread along a network.

In this thesis, we will begin by exploring the classical zero forcing properties of a particular family of graphs, called proper interval graphs, and subgraphs of this family. After fully characterizing these subgraphs, we will extend these results to random subgraphs and determine the zero forcing properties of these random subgraphs. Then, in Chapter 3, we will develop some zero forcing results on a variation of an extensively studied family of graphs, the hypercube.

In the context of probabilistic zero forcing, the question of how long it takes to infect a graph is the main focus. In Chapter 4, we will consider the problem of which vertices minimize the expected time it takes for a graph to become fully infected for a particular family of proper interval graphs.

Now, we will begin by introducing the basic concepts in graph theory, zero forcing, and probability that will be required for this thesis, as well as some previous results.

1.1 Graph Theory

A *graph*, G, is the ordered pair, (V, E), where V is the set of vertices, and E is the set of undirected edges. If a graph has no multiple edges or loops, then it is called a *simple graph*, otherwise it is a *multigraph*. Unless otherwise stated, the order of any graph is n, $|V| = n$.

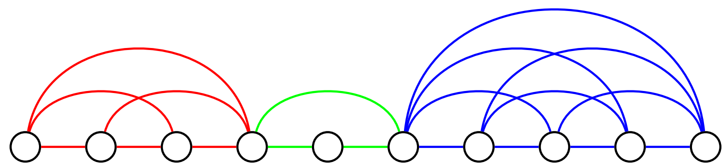
If two vertices, u and v, are connected by an edge, we say that u is *adjacent* to v, or that u and v are neighbours, denoted $u \sim v$. The set of neighbours of a vertex, v, is called the *open neighbourhood* of v, $N(v)$. The set $N(v) \cup \{v\}$ is called the *closed neighbourhood* of v, denoted $N[v]$. The size of the open neighbourhood of a vertex, v is called its *degree*, $\deg(v)$.

If there is a path from a vertex to any other vertex in G, then we say that G is *connected*. For the purposes of this thesis, when I say graph, I refer to an undirected, simple, connected graph.

For a graph, G, the minimum degree of G, $\delta(G)$, is the smallest degree of any of the vertices of G.

A *clique* in a graph is a set of pairwise adjacent vertices A *clique cover* of a graph, G, is a set of cliques (not necessarily disjoint) such that every edge is contained in a clique. The smallest size of a clique cover for G is called the *clique cover number* of G, denoted $cc(G)$.

Example 1. Consider the following graph.



This graph has clique cover number 3, with a minimal clique cover indicated by the edges in red, green, and blue.

For an edge, e, in a graph G, denote $G - e = (V, E \setminus \{e\})$ as the graph representing the deletion of the edge e from G.

One family of graphs of particular interest are the *proper interval graphs.*

Definition 1. A graph is called an *interval graph* if each of its vertices can be associated with an interval on the real line in such a way that two vertices are adjacent if and only if the associated intervals have a nonempty intersection.

Definition 2. A graph is called a *proper interval graph* if it is an interval graph, and no interval is contained within another. Label the vertices from 1 to n, in the order of the start of the intervals that correspond to each vertex, from left to right. We will refer to this labeling as the standard labeling. See Figure 1.1.

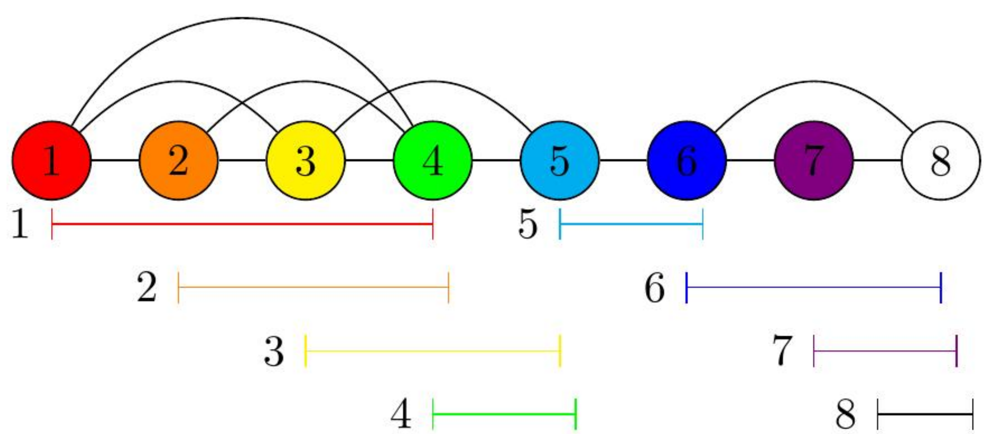


Figure 1.1: An example of a proper interval graph, with coloured intervals corresponding to the same coloured vertex

Proper interval graphs naturally arise in the context of graph infection as they can be viewed as a simple model of groups of people, say households, clubs, or any group that shares a space. Each clique corresponds to a club, and any intersection between the cliques will represent members of multiple clubs. While this family of graphs has its limitations of accurately modeling the complexity of such networks, it is an excellent place to start.

Another family of graph that will arise are the q-trees.

Definition 3. A *q-tree*, T_n^q, is defined by the following recursive construction:

- K_q, the complete graph on q vertices, is a q-tree.
- A q-tree on $n > q$ vertices is constructed by beginning with a q-tree on $n-1$ vertices, T_{n-1}^q, adding a vertex, and attaching the new vertex to all vertices in a copy of K_q in T_{n-1}^q.

Specifically, I will work with the subset of q-trees that I call q-paths.

Definition 4. A *q-path* on n vertices, P_n^q, is a q-tree where the recursive construction has the following restriction:

- K_q is a q-path.
- A q-path on $n > q$ vertices is constructed by beginning with a q-path on $n-1$ vertices, P_{n-1}^q, adding a vertex, and attaching the new vertex to the copy of K_q in P_{n-1}^q containing the vertex $n-1$.

These q-paths are a special case of the proper interval graphs, where each interval, i, intersects with an interval $j \neq i$ whenever $|j-i| \leq q$.

I will also discuss known properties of the hypercube graphs, and some results in extending these properties to a family of graphs known as twisted hypercubes, defined below:

Definition 5. A *hypercube of dimension k*, Q_k, has vertex set $\{0,1\}^k$, with vertices adjacent when they differ in exactly one coordinate. See Figure 1.2 for a 3-dimensional hypercube.

Definition 6. The unique twisted hypercube of dimension 0 consists of a single vertex. For $k \geq 1$, a *twisted hypercube of dimension k* is obtained from two twisted hypercubes of dimension $k-1$ by adding a matching joining the vertex sets of the two smaller graphs.

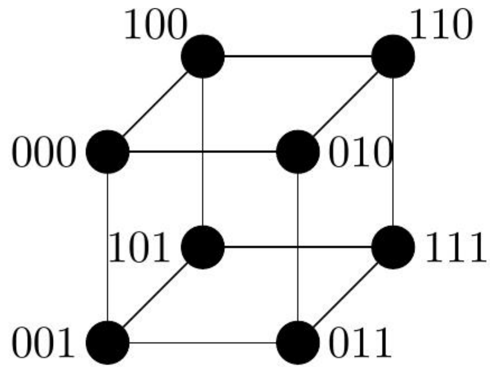


Figure 1.2: The 3-dimensional hypercube graph, Q_3.

1.2 Zero Forcing

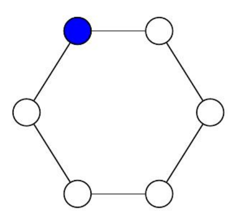
Definition 7. Given a graph, G, where each vertex is coloured either white or blue, and an initial set of blue vertices, $S \subseteq V(G)$, we define *zero forcing* as the graph infection process in which we iteratively apply the following colour change rule:

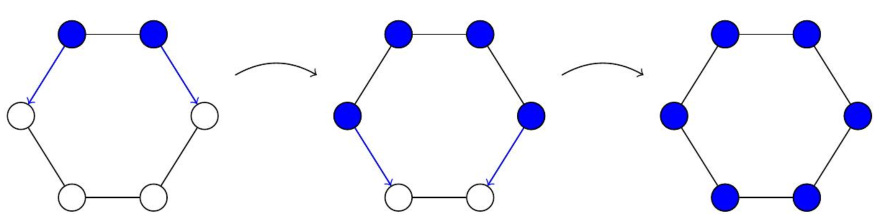
If a blue vertex has exactly one white neighbour, then this neighbour changes to blue.

The *derived set* is the set of blue vertices after performing all possible forces. If the derived set is the entire vertex set, then we call S a *zero forcing set*.

The size of the smallest zero forcing set is called the *zero forcing number* of G, and is denoted $Z(G)$. If a zero forcing set has size $Z(G)$, then it is called an *optimal zero forcing set*.

Example 2. Consider the cycle on 6 vertices.

i) 

ii) 

In (i), we try to force the graph with a single vertex. As every choice of initial blue vertex will have two white neighbours, no forces are possible, and there are no zero forcing sets of size 1.

In (ii), we choose two vertices that are adjacent as the initial set. Each of these vertices has exactly one white neighbour, and can therefore force this neighbour. Each newly forced vertex has exactly one white neighbour, and therefore forces it. After these forces, all vertices are blue and our initial vertex set is a zero forcing set. This also shows that the cycle has zero forcing number 2.

Note that we do not allow one vertex to be forced by multiple neighbours. If a vertex can be forced by multiple neighbours, then we choose one to perform the force. Also, we can see that each vertex can only force at most one other vertex.

If vertex u is blue with one white neighbour, v, then u *forces* v in the zero forcing process. This will be denoted $u \rightarrow v$.

Zero forcing was first used as a tool in studying the minimum rank problem of graphs, introduced in [20]. The minimum rank problem is as follows:

Given an $n \times n$ real symmetric matrix $A = [a_{u,v}]$, we may define an undirected graph $G(A)$ on n vertices $1, 2, \ldots, n$, by including the edge joining vertex u to vertex v in the edge set, if and only if $a_{u,v} \neq 0$. (We always ignore loops, $a_{v,v}$). Then, given a graph, G, an *adjacency matrix of* G is a matrix, A, such that $G(A) = G$.

For a graph, G, define $\mathcal{S}(G) = \{A | G(A) = G\}$ to be the set of all adjacency matrices of G with entries over $\mathbb{R}$. The minimum rank of G is $mr(G) = \{\text{rank}(A) | A \in \mathcal{S}(G)\}$. The problem is to determine the minimum rank of any graph.

The zero forcing number of a graph is used as a bound for the minimum rank of a matrix, or more precisely, a bound on the maximum nullity, which is similarly defined, $M(G) = \{\text{null}(A) | A \in \mathcal{S}(G)\}$.

Theorem 1.1. *[1] For any graph, G,*

$$M(G) \leq Z(G).$$

This bound was known for some time before the AIM Minimum Rank-Special Graphs Work Group formalized the notion of the zero forcing number in [1], and began studying this new parameter in its own right. They were able to flip the script and use maximum nullity results to determine zero forcing properties of well known graph families.

Theorem 1.1, along with the following results, provide useful bounds on the zero forcing number of graphs that I will use regularly throughout this thesis:

Observation 1. For a graph, G,

$$Z(G) \geq \delta(G).$$

This is straightforward to see, as for a force to occur, a vertex must have its entire closed neighbourhood coloured blue except for one vertex. The smallest such set in a graph G has exactly $\delta(G)$ vertices.

When considering subgraphs, we need to understand how zero forcing properties react to deletion of edges. Note the following definition of a forcing chain from [10]:

Definition 8. Consider a zero forcing set of a graph, G. Construct the derived set, writing all of the forces as directed edges. Then, the graph induced by these directed edges is acyclic, and consists of vertices with at most one in-edge, and one out-edge. Therefore, the graph induced by the directed edges is a disjoint collection of directed paths. These paths are called *forcing chains*. A maximal forcing chain is a forcing chain that is not a proper subsequence of another zero forcing chain.

Notice that collections of forcing chains need not be unique. If a vertex can be forced by more than one neighbour, then choosing to force with one or the other neighbour results in two different collections of forcing chains.

The zero forcing number of a graph can be thought of as the size of a minimal collection of forcing chains. Furthermore, we can compute the number of forces, or the number of edges, in a minimal collection of forcing chains as

$$|E(\mathcal{F})| = n - Z(G),$$

which, in turn, rearranges to give another expression for the zero forcing number

$$Z(G) = n - |E(\mathcal{F})|.$$

Note that in a clique, at most one edge can be a forcing edge. Therefore, every edge in $\mathcal{F}$ is in one clique. This gives that $|E(\mathcal{F})| \leq cc(G)$. Therefore,

$$\begin{aligned} Z(G) &= n - |E(\mathcal{F})| \\ &\geq n - cc(G). \end{aligned}$$

Observation 2. [1] For a graph, G, of order n,

$$Z(G) \geq n - cc(G).$$

The following lemma proves very useful in studying the zero forcing sets of subgraphs.

Lemma 1.2. *[10] Let G be a graph, and S a zero forcing set of G with collection of forcing chains $\mathcal{F}$. If an edge $e \notin \mathcal{F}$, then S is also a zero forcing set of the graph $G - e$.*

Proof. Let S be a zero forcing set of G with collection of forcing chains $\mathcal{F}$. Assume that an edge, e, is not in $\mathcal{F}$. Then, consider the graph $G - e$ with initial set of blue vertices S. Since $e \notin \mathcal{F}$, there is not a force along the edge e in this list of forces in G. Therefore, the same forces as in $\mathcal{F}$ are still possible in $G - e$, as no vertex can gain a white neighbour by the removal of an edge. □

This shows that there is at least a case when the zero forcing number will not increase upon deletion of an edge. Unfortunately, in general, there is not a monotonic relationship between the zero forcing number of a graph and the zero forcing number of its subgraphs.

Theorem 1.3. *[10] Let G be a graph with zero forcing number $Z(G)$. For an edge, e, define $z_e(G) = Z(G) - Z(G - e)$ to be the edge spread of e in G. Then,*

$$-1 \leq z_e(G) \leq 1$$

This, however, is good enough to prove quite useful in determining the zero forcing number of subgraphs. In particular, if one can demonstrate a zero forcing set of size $Z(G) - 1$ for any subgraph of G, then it immediately follows that the zero forcing number of the subgraph is exactly one less than the original graph.

1.3 Probabilistic Zero Forcing

Probabilistic Zero Forcing was introduced by Kang in 2012 [17] as an extension of regular zero forcing that reduces to classical zero forcing in a special case. As mentioned above, this model of graph infection is a more believable model of short term infection spread across a network, while maintaining some of the convenience that comes with classical zero forcing.

First, I will need to outline some of the basic probabilistic concepts required for this thesis. See [11], or any introductory probability textbook for the following definitions and theorems.

Let the triple $(\Omega, \mathcal{F}, P)$ be a probability space where Ω is the sample space, $\mathcal{F}$ is an event space, and P is a probability measure. Note that for the purposes of this thesis, all event spaces will be countable, and we take $\mathcal{F} = 2^{\Omega}$.

The expectation of a random variable X, $\mathbb{E}(X)$, will be defined as

$$\mathbb{E}(X) = \sum_{i=-\infty}^{\infty} iP(X = i).$$

Definition 9. Law of Total Probability

For the discrete case of The Law of Total Probability:

Let the events $\{B_k\}_{k\in\mathbf{N}}$ be a countably infinite or finite partition of a sample space, where each event, B_k is in $\mathcal{F}$. If A is an event in the same probability space, then

$$P(A) = \sum_k P(A \cap B_k),$$

equivalently,

$$P(A) = \sum_k P(A|B_k)P(B_k).$$

For example, if the partition consists of only two elements,

$$\{B_k\} = \{B_1, B_2\},$$

in the same probability space, then the probability of the event A can be written as

$$P(A) = P(B_1)P(A|B_1) + (1 - P(B_1))P(A|B_2).$$

Some other probabilistic concepts I will use in this thesis:

Theorem 1.4. *Markov's Inequality*

Let $(\Omega, \mathcal{F}, P)$ be a probability space, X a random variable on this space, and $a \in \mathbb{Z}^{\star}$. Then

$$P(X \geq a) \leq \frac{\mathbb{E}(X)}{a}.$$

Theorem 1.5. *Chebyshev's Inequality*

Let X be a random variable with variance $\mathrm{Var}(X)$*. Then for all positive real numbers a,*

$$P(|X - \mathbb{E}(X)| \geq a) \leq \frac{\mathrm{Var}(X)}{a^2}.$$

Definition 10. Martingale

A sequence of random variables $\{X_i\}$ is a *Martingale* if

$$\mathbb{E}(|X_i|) < \infty,$$
$$\mathbb{E}(X_i | X_1, \ldots, X_{i-1}) = X_{i-1}.$$

A sequence of random variables $\{X_i\}$ is a *submartingale* if

$$\mathbb{E}(|X_i|) < \infty,$$
$$\mathbb{E}(X_i | X_1, \ldots, X_{i-1}) \geq X_{i-1}.$$

Definition 11. Martingale with respect to another sequence

Let $\{X_i\}$ be a sequence of random variables. A sequence of random variables $\{Y_i\}$ is said to be a *Martingale with respect to* $\{X_i\}$ if

$$\mathbb{E}(|Y_i|) < \infty,$$
$$\mathbb{E}(Y_i | X_1, \ldots, X_{i-1}) = Y_{i-1}.$$

Definition 12. Stopping Time

Let $\{X_i\}$ be a Martingale. A *Stopping Time* with respect to $\{X_i\}$ is a random variable $T \in \{1, 2, \ldots\} \cup \{\infty\}$, such that the event $\{T \leq n\}$ can be determined from $X_1, \ldots, X_n$.

Theorem 1.6. *Stopping Time Theorem*

If $\{Y_t\}$ is a Martingale with respect to $\{X_t\}$ such that Y_t is uniformly integrable, and T is a stopping time with respect to $\{X_t\}$, such that $P(T < \infty) = 1$, then,

$$\mathbb{E}(Y_T) = Y_0.$$

Similarly, for Y_t a submartingale,

$$\mathbb{E}(Y_T) \geq Y_0.$$

Definition 13. Markov Chain

Consider a discrete-time stochastic process represented by a sequence of random variables, $\{X_t\}_{t\geq 0}$, which take values in a countable set S. The process $\{X_t\}$ is a *Markov chain* if, for all $t \geq 0$ and all $s, x_0, x_1, ..., x_t \in S$,

$$P(X_{t+1} = s | X_0 = x_0, X_1 = x_1, ..., X_t = x_t) = P(X_{t+1} = s | X_t = x_t).$$

Definition 14. Hitting Time

Let $\{X_t\}$ be a Markov chain with finite state space, S. For any $u, v \in S$, the *hitting time*, $h_{u,v}$, is the expected number of steps it takes to reach state v when starting at state u. Precisely,

$$h_{u,v} = \sum_{t=1}^{\infty} t \cdot P\left(\{X_t = v\} \cap \bigcap_{i=1}^{t-1} \{X_i \neq v\} | X_0 = u\right).$$

Theorem 1.7. *Maximum of two Random Variables*

Let X and Y be two independent, discrete random variables that take values on $\{0, 1, 2, \dots\}$. Then,

$$\mathbb{E}(\max\{X, Y\}) = \sum_{x=0}^{\infty} P(X = x) \left(x + \sum_{y=x+1}^{\infty} P(Y \geq y)\right).$$

Proof. Let X, Y be independent, discrete random variables. Let $Z = \max\{X, Y\}$. Then,

$$\begin{aligned}
\mathbb{E}(Z) &= \sum_{z=0}^{\infty} P(\max\{X, Y\} \geq z) \\
&= \sum_{z=0}^{\infty} \sum_{x=0}^{\infty} P(X = x) P(\max\{x, Y\} \geq z | X = x) \\
&= \sum_{x=0}^{\infty} P(X = x) \left(\sum_{z=0}^{\infty} P(\max\{x, Y\} \geq z)\right), \text{by independence} \\
&= \sum_{x=0}^{\infty} P(X = x) \left(x + \sum_{z=x+1}^{\infty} P(Y \geq z)\right).
\end{aligned}$$

□

The preceding probabilistic concepts have proven vital in the study of probabilistic zero forcing, which I can now define.

Definition 15. [17] For a graph, G, consider a set of blue vertices, S. With respect to S, define the probabilistic colour change rule as follows:

Let $P(u \to v)$ be the probability that the event $u \to v$ occurs. Then

$$P(u \to v) = \begin{cases} \frac{|N[u] \cap S|}{|N(u)|}, & \text{if } u \in S \text{ and } v \in N(u) \cap S^c \\ 0 & \text{otherwise,} \end{cases}$$

where u forces each of its white neighbours independently.

So the probability that u forces one of its neighbours depends on the number of blue neighbours of u. As you can see, when u has exactly one white neighbour, v, the probability that $u \to v$ is

$$P(u \to v) = \frac{deg(u) - 1 + 1}{deg(u)} = 1,$$

and classical zero forcing falls out of this definition as a special case of probabilistic zero forcing.

There will be occasions where I am only interested in whether a certain vertex becomes forced, and will not care which of its neighbours actually performed the force. In this case, I will consider the event that v is forced, denoted $\to v$. This is defined as

$$\{\to v\} = \bigcup_{u \in N(v)} \{u \to v\}.$$

There will also be instances where I am interested in whether a vertex does not become forced. The event that u does not force v is exactly $\{u \to v\}^C$, which I will denote $\{u \nrightarrow v\}$, and $P(u \nrightarrow v) = 1 - P(u \to v)$. Similarly, I will define the event that a vertex v is not forced by any of its neighbours as $\nrightarrow v$, where

$$\begin{aligned} \{\nrightarrow v\} &= \{\to v\}^C \\ &= \left(\bigcup_{u \in N(v)} \{u \to v\} \right)^C \\ &= \bigcap_{u \in N(v)} \{u \nrightarrow v\}. \end{aligned}$$

As each potential force in the probabilistic colour change rule is independent of the others, this gives a convenient way to compute whether or not a vertex is forced after a particular application of the probabilistic colour change rule:

$$\begin{aligned} P(\nrightarrow v) &= P\left(\bigcap_{u\in N(v)} u \nrightarrow v\right) \\ &= \prod_{u\in N(v)} P(u \nrightarrow v), \end{aligned}$$

$$\begin{aligned} P(\rightarrow v) &= 1 - P(\nrightarrow v) \\ &= 1 - \prod_{u\in N(v)} P(u \nrightarrow v). \end{aligned}$$

Notice that for any connected graph G, given any set of initially blue vertices, G will become entirely forced eventually with probability 1. So, every subset of vertices will be a probabilistic zero forcing set. This means that rather than studying whether a graph will become completely forced, we will determine how quickly a graph can be forced.

Definition 16. [12] The *probabilistic propagation time* of a set, S, of vertices of a connected graph, G, $pt_{pzf}(G, S)$, is a random variable that reflects the time (number of iterations of the probabilistic colour change rule) at which the last white vertex turns blue when applying a probabilistic zero forcing process starting with the set S blue. For a graph G and a set S of vertices, the expected propagation time of S for G is the expected value of the propagation time of S. i.e.,

$$ept(G, S) = \mathbb{E}[pt_{pzf}(G, S)].$$

The expected propagation time of a connected graph G is the minimum of the expected propagation time of S for G over all one vertex sets S. i.e.,

$$ept(G) = \min_{v\in V(G)}\{ept(G, \{v\})\}.$$

Bounds on the expected propagation time have been determined for various families of graphs. I will state the most useful bound for the purposes of this thesis as its own result, and the others will be listed in Table 1.1.

Theorem 1.8. *[12] For the star on $n+1$ vertices,*

$$ept(K_{1,n}) = \Theta(log\ n).$$

Graph, G	$ept(G)$
C_n[12]	$\begin{cases} \frac{n}{2}+\frac{1}{3} & \text{if } n \text{ is even} \\ \frac{n}{2}+\frac{1}{2} & \text{if } n \text{ is odd} \end{cases}$
P_n[12]	$\begin{cases} \frac{n}{2}+\frac{2}{3} & \text{if } n \text{ is even} \\ \frac{n}{2}+\frac{1}{2} & \text{if } n \text{ is odd} \end{cases}$
K_n[7]	$\Theta(\log\log n)$
A spider graph with k legs[12]	$rad(G)+O(\log k)$
$K_{m,n}$ for any $m,n \in \mathbb{Z}_+$[7]	$O(\log(m+n))$
$K_{c,n}$ for any $n \in \mathbb{Z}_+$ and fixed c[7]	$\Theta(\log n)$
Any connected graph, G[19]	$\frac{n}{2}+O(\log n)$
$G_{m\times n}$, the $m \times n$ grid graph[15]	$(\frac{1}{2}-o(1))(m+n) \leq ept(G_{m\times n}) \leq (4+o(1))(m+n)$
Any d-regular graph, $d \geq 2$[15]	$O(n\frac{\log d}{d})$
Q_n[15]	$O(n\log n)$

Table 1.1: The expected propagation times of various families of graphs

Chapter 2

Zero Forcing on Proper Interval Graphs

Proper interval graphs have a structure resembling intersecting cliques in a linear arrangement. Before I begin, it will be useful to have the following definition:

Definition 17. Let G be a proper interval graph where vertices are labeled according to the standard labeling in Definition 2, and $e \in E(G)$ an edge in G. Then, e is an *outer edge* of G if e is in a maximal clique of order $a+1$, and $e = (r, r+a)$. Otherwise, e is an *inner edge* of G.

In other words, the outer edges of a proper interval graph are the edges that have no other edges above them. We will say that one edge, $e = (i, j)$, covers another edge, $e' = (i', j')$, if $i \leq i'$ and $j \geq j'$. So, an outer edge covers all edges in the maximal clique that contains it.

For example, in Figure 1.1, the edges $(1, 4), (3, 5), (5, 6)$, and $(6, 8)$ are all of the outer edges. The inner edges are those that have at least one other edge completely covering them. We can consider the clique cover number of proper interval graphs with respect to this definition.

Lemma 2.1. *Let G be a proper interval graph. The clique cover number of G is equal to the number of outer edges in G.*

Proof. Let G be a proper interval graph. Notice that each outer edges defines a clique. As every edge is in a clique defined by an outer edge, the outer edges define a particular clique cover. So

$$cc(G) \leq \text{ the number of outer edges.}$$

Also, each clique contains at most one outer edge, because outer edges are incident to the first and last vertex of the clique. Therefore,

$$cc(G) \geq \text{ the number of outer edges.}$$

$\square$

And so, the clique cover number of a proper interval graph is simply the number of outer edges. From this perspective, the clique cover number, $cc(G)$, is the number of maximal cliques along the interval. Notice that every clique intersects with at least one other clique in at least one vertex. Only the first clique has its first vertex in only one clique, and only the last clique has its last vertex in only one clique.

From this, the first result is as follows:

Theorem 2.2. *[16] Let G be a proper interval graph. Then,*

$$Z(G) = n - cc(G).$$

I will include a proof of this result because it demonstrates the particular construction of a zero forcing set that I will refer to for the results in this chapter. See Figure 2.1 for an example.

Proof. The lower bound follows from Observation 2 in Chapter 1.2. This lower bound holds for all graphs. Therefore, it suffices to demonstrate a particular zero forcing set on $n - cc(G)$ vertices.

Consider an optimal clique covering, $\{K_1, \ldots, K_r\}$, of G, as defined by the outer edges of G, , where the cliques are ordered by their smallest vertices. Consider the set S of all vertices except those that are the larger indexed vertex of an outer edge.

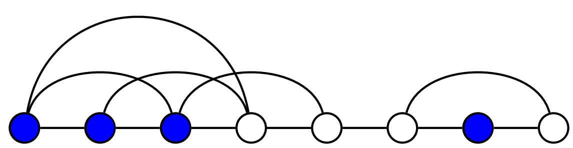


Figure 2.1: A construction of a zero forcing set for Theorem 2.2

This is a zero forcing set of order $n - cc(G)$. We will demonstrate this by strong induction on the cliques in the clique cover, K_i. That is, we will show inductively that all vertices in cliques $K_1, \ldots, K_r$ are forced.

Base Case: First clique, K_1: Vertex 1 is adjacent to only the other vertices in the first clique, by construction. All of these are blue except for the final vertex in this clique. Therefore, 1 forces this final vertex.

Induction Hypothesis: Fix $i \geq 2$. Suppose that all vertices in cliques up to K_i are forced. Now consider the clique K_{i+1}.

The first vertex of K_{i+1}, v, is necessarily blue because it is a member of a previous clique. Note that all neighbours of v are either less than v or in K_{i+1}. Consider a neighbour, u, of v. If u is in one of the cliques, $K_1, \ldots, K_i$, then u is blue by the induction hypothesis. Otherwise, $u \in K_{i+1}$. If u is the final vertex of K_{i+1}, then u will be white, by the construction of the set. Let u be any other vertex in K_{i+1}. Then, u is either contained in another previous maximal clique, and is therefore blue by the induction hypothesis, or u is not in a previous clique, and is blue by the construction of S. So, all vertices less than v are blue, and all but the final vertex in K_{i+1} are blue, so v forces the final vertex in K_{i+1}.

This proves that the constructed set is a zero forcing set.

□

Recall from Definition 8 that the list of forces corresponds to a collection of forcing chains, $\mathcal{F}$. As each of the forces for this zero forcing set occurs along an outer edge, this collection of forcing chains consists of one primary directed path along the outer edges, and $n - cc(G) - 1$ isolated vertices.

The construction for the zero forcing set in Theorem 2.2 will be the standard set that I consider for the results in the rest of this chapter. Notice that the forcing chains for this zero forcing set are not unique. There could be more than one vertex that can force a neighbour at each step of the process. For the purposes of the following chapter, the particular collection of forcing chains we will refer to is the one described in Theorem 2.2, $\mathcal{F}$, which is the directed path along the outer edges.

A *reversal* of a zero forcing set is the set of final vertices of the maximal zero forcing chains of a chronological list of forces. In [3], Barioli et al. prove that the reversal of any zero forcing set is another zero forcing set. This means that forcing the graph in the other direction also constitutes a minimal zero forcing set, and if we reverse the ordering of the vertices, the reversal will have the same structure as described in Theorem 2.2.

2.1 q-paths

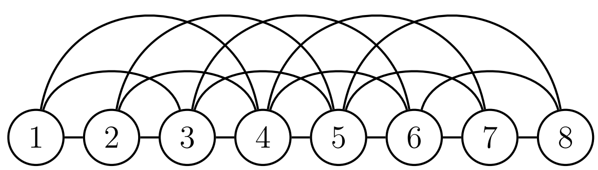
One particular family of proper interval graphs that I will discuss in this section are q-paths. Recall from Definition 4:

Definition 4. A *q-path* on n vertices, P_n^q, is a q-tree where the recursive construction has the following restrictions

- K_q is a q-path.
- A q-path on $n > q$ vertices is constructed by beginning with a q-path on $n-1$ vertices, P_{n-1}^q, adding a vertex, and attaching the new vertex to the copy of K_q in P_{n-1}^q containing the vertex $n-1$.

Another characterization of q-paths is that P_n^q is a proper interval graph of order n with $e = (i, j) \in E(P_n^q)$ iff $|i - j| \leq q$.

Figure 2.2: A 3-path on 8 vertices, P_8^3



The zero forcing number of these graphs follows from Theorem 2.2.

Corollary 2.3. *Let P_n^q be a q-path on $n \geq q+1$ vertices. Then*

$$Z(P_n^q) = q.$$

Proof. By Theorem 2.2, as P_n^q is a proper interval graph, it has zero forcing number

$$Z(P_n^q) = n - cc(P_n^q).$$

We saw that the $cc(P_n^q)$ is equal to the number of outer edges in P_n^q. Notice that, in P_n^q, every vertex after vertex q is the second vertex of an outer edge. So there are $n-q$ outer edges, and therefore the clique cover number of P_n^q is $n-q$. This gives

$$\begin{aligned} Z(P_n^q) &= n - cc(P_n^q) \\ &= n - (n-q) \\ &= q. \end{aligned}$$

$\square$

In general, subgraphs of proper interval graphs are not necessarily proper interval graphs. When $q = 1$, P_n^q is just a path, and subgraphs of paths with the same vertex set are disconnected paths. The zero forcing number of these subgraphs are just the number of connected components, and are not of interest. So, I will consider the case when $q \geq 2$. Looking at particular subgraphs, as well as random subgraphs, leads to the following result.

Theorem 2.4. *Let P_n^q be a q-path on $n \geq q+2$ vertices. Let $|E(P_n^q)| = m$. If an edge, e, of P_n^q is chosen uniformly at random, then the expected zero forcing number of the graph $P_n^q - e$ is,*

$$\begin{aligned}\mathbb{E}(Z(P_n^q - e)) &= Z(P_n^q) + \frac{cc(P_n^q) - 2}{m} \\ &= q + \frac{n-q-2}{m}.\end{aligned}$$

The proof of this result first requires determining the effect of removing edges from P_n^q on the zero forcing number. These are Lemmas 2.5-2.7. For these results, P_n^q is a q-path of degree $n \geq q+2$.

Recall the definition of edge spread, $z_e(G)$, for an edge, e, in a given graph, G, from Theorem 1.3, namely

$$z_e(G) = Z(G) - Z(G-e).$$

Lemma 2.5. *If e is an inner edge of P_n^q, then*

$$z_e(P_n^q) = 0.$$

Proof. Consider the zero forcing set of P_n^q described in Theorem 2.2, with collection of forcing chains $\mathcal{F}$. Edge e is not in $\mathcal{F}$, so the same set is still a zero forcing set in $P_n^q - e$ by Lemma 1.2. Therefore,

$$Z(P_n^q - e) \leq Z(P_n^q).$$

- If $e \neq (1, i)$ or (j, n), then $\delta(P_n^q - e) = q$. Therefore,

  $$Z(P_n^q) = q \leq Z(P_n^q - e),$$

  by Observation 1. So, in this case we find that

  $$Z(P_n^q - e) = Z(P_n^q).$$

- If e is incident to either 1 or n, (say 1, by symmetry), then

$$\begin{aligned} q - 1 &= Z(P_n^q) - 1 \\ &\leq Z(P_n^q - e). \end{aligned}$$

We will show, by contradiction, that no zero forcing set of size $q-1$ exists. Let $e = (1, k)$, for some $2 \leq k \leq q$. Suppose S is a zero forcing set of $P_n^q - e$ of size $q-1$. As 1 is the only vertex of degree $q-1$, S must consist of 1 and $q-2$ of its neighbours. Then, 1 has neighbours $2, \ldots, k-1, k+1, \ldots q+1$, all of which are blue, except for some neighbour, j, so 1 can force j. Every vertex in the first clique, other than 1, is adjacent to k, as well as at least one other white vertex $> q+1$, since P_n^q is a q-path. As these are all of the blue vertices in the graph, no further forces are possible. So S must not have been a zero forcing set to begin with, and no such set exists. Therefore,

$$\begin{aligned} Z(P_n^q - e) &\geq q \\ &= Z(P_n^q). \end{aligned}$$

So we find that

$$Z(P_n^q - e) = Z(P_n^q)$$

whenever e is an inner edge. □

When an outer edge, $e = (i, j)$, is removed from a proper interval graph, the resulting graph is still a proper interval graph. This is because removing an edge is equivalent to shortening one interval so it intersects with exactly one less interval. If we choose to shorten the interval corresponding to vertex i, the left endpoint of the interval remains in place, while the right endpoint moves to the left until it no longer intersects with the interval corresponding to vertex j. As the left endpoint has not moved, no interval to the right of i can contain i, and since i was an outer edge, no interval to the left of i will intersect with interval j, meaning that no interval to the left of i can contain i. Therefore, removing the outer edge e from a proper interval graph results in another proper interval graph. See Figure 2.3 for a diagram of this procedure.

Therefore, we can determine the effect that removing an outer edge has on the clique cover number of a q-tree in the following way:

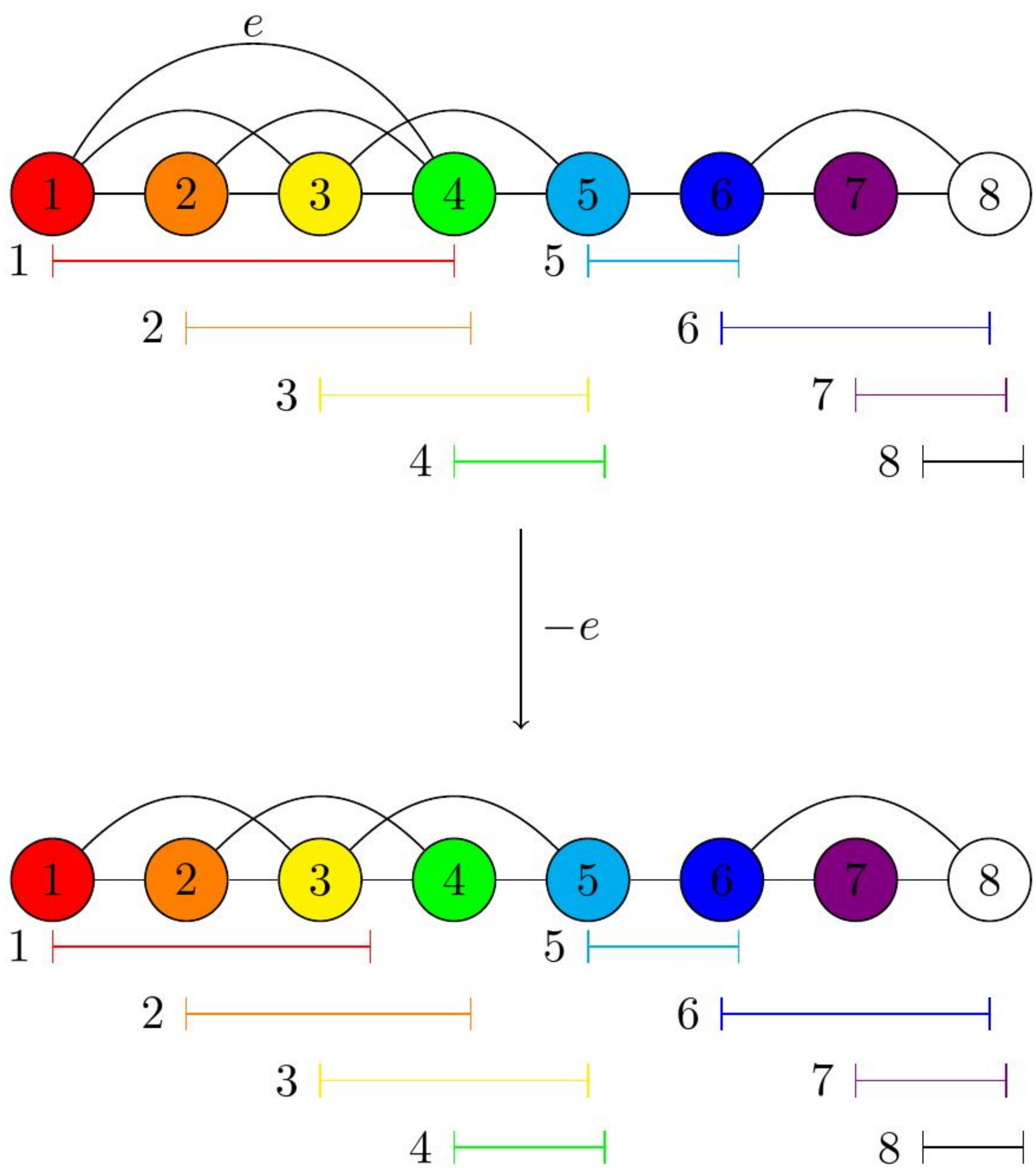


Figure 2.3: An example showing how removing an outer edge of a proper interval graph results in another proper interval graph

Lemma 2.6. *Let e be an outer edge. If e is incident to 1 or n, then*

$$cc(P_n^q - e) = cc(P_n^q).$$

Otherwise,

$$cc(P_n^q - e) = cc(P_n^q) - 1.$$

Proof. First, assume that e is incident to vertex 1. The argument for e incident to n is exactly the same.

Removing e from P_n^q removes an outer edge, but once e is removed, the edge $e' = (1, q)$ now has no other edges completely covering it. Therefore e' is now an outer edge. Any other edge in $P_n^q - e$ is either already an outer edge, or will have been covered by another edge in P_n^q, and will therefore still be covered in $P_n^q - e$. This gives us

$$cc(P_n^q - e) = cc(P_n^q),$$

because P_n^q and $P_n^q - e$ have the same number of outer edges.

Now, assume that e is not incident to either 1 or n. So $e = (i, i+q)$. Removing e from P_n^q still removes an outer edge, but in this case, any edge covered by e is also covered by $(i-1, i+q-1)$ or $(i+1, i+q+1)$. Note that $i+q+1 \leq n$ because e is not incident to n.

Consider any edge, e', that is covered by e in P_n^q. Then the endpoints of e' must be between i and $i+q$, because it was covered by e. If the left endpoint of e' is i, then the right endpoint must be less than q, but then e' will also be covered by the edge $(i-1, i+q-1)$.

Similarly, if the right endpoint of e' is $i+q$, then e' will also be covered by $(i+1, i+q+1)$. Any smaller edge covered by e will, then, also be covered by these edges. Therefore, every edge covered by e is covered by at least one other edge, and so removing e from P_n^q will not result in additional outer edges in $P_n^q - e$.

The only other case to consider is for edges not previously covered by e, but these edges will either already be outer edges, or will still be covered by the same outer edge as in P_n^q. This gives that the outer edges of $P_n^q - e$ are exactly the outer edges of P_n^q, minus e.

So, removing e from P_n^q only removes an outer edge, and no new outer edge takes its place. This means that

$$cc(P_n^q - e) = cc(P_n^q) - 1$$

$\square$

Understanding, now, how removing outer edges affects a q-tree, we can determine the effect of removing these edges on the zero forcing number of the graphs.

Lemma 2.7. *Let e be an outer edge. If e is incident to 1 or n, then*

$$z_e(P_n^q) = 0.$$

Otherwise,

$$z_e(P_n^q) = -1.$$

Proof. In either case, the resulting graph, $P_n^k - e$, is a proper interval graph. So we just need to look at the clique cover number of $P_n^q - e$.

- If $e = (1, q)$ or $(n-q, n)$, then $cc(P_n^q - e) = cc(P_n^q)$ by Lemma 2.6. Therefore,

$$\begin{aligned} Z(P_n^q - e) &= n - cc(P_n^q - e) \\ &= n - cc(P_n^q) \\ &= Z(P_n^q). \end{aligned}$$

- If $e \neq (1, k)$ or $(n-k, n)$, then $cc(P_n^q - e) = cc(P_n^q) - 1$ by Lemma 2.6. Therefore,

$$\begin{aligned} Z(P_n^q - e) &= n - cc(P_n^q - e) \\ &= n - cc(P_n^q) + 1 \\ &= Z(P_n^q) + 1. \end{aligned}$$

$\square$

This concludes the required lemmas describing how the zero forcing number is affected by removing any edges from a q-tree. We are now ready for the proof of Theorem 2.4.

Proof of Theorem 2.4: From Lemmas 2.5 and 2.7, we know that the zero forcing number increases when an edge, e, is removed from P_n^q, and e is an outer edge, not of the first or last clique. Otherwise, removing e does not change the zero forcing number.

We also know that there are exactly $cc(P_n^q)$ outer edges in P_n^q. So, when choosing an edge uniformly at random, the probability that the chosen edge increases the zero forcing number is $\frac{cc(P_n^q)-2)}{m}$. Define the set $W \subset E$ to be the set of outer edges in E that are not incident to the vertices 1 or n.

Define the random variable $X_{W,e}$ to be

$$X_{W,e} = \begin{cases} 1, & \text{if } e \in W \\ 0, & \text{if } e \notin W \end{cases}$$

so, $X_{W,e}$ is the increase in zero forcing number when removing e from P_n^q. When the edge is chosen uniformly a random, we find that

$$\mathbb{E}(X_{W,e}) = \frac{cc(P_n^q) - 2}{m}.$$

So the zero forcing number would be expected to increase by $\frac{cc(P_n^q)-2}{m}$, or the expected zero forcing number of $P_n^q - e$ is

$$\begin{aligned} Z(P_n^q - e) &= Z(P_n^q) + \frac{cc(P_n^q) - 2}{m} \\ &= q + \frac{n - q - 2}{m}. \end{aligned}$$

□

2.2 Edge-Disjoint Proper Interval Graphs

Consider the family of proper interval graphs where none of the cliques share an edge. We will call this family of graphs Edge-Disjoint Proper Interval Graphs. See Figure 2.2 for an example with a minimal zero forcing set.

If a proper interval graph has a 2-clique as a maximal clique, then the associated edge is a cut-edge, and therefore removing that edge disconnects the graph. In the case of proper interval graphs, the resulting graph consists of two connected components, both of which are proper interval graphs. This is because of the path-like structure of the proper interval graphs. No vertex to the left of the maximal 2-clique, u, is

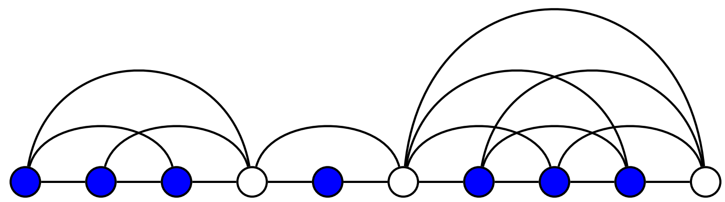


Figure 2.4: An edge-disjoint proper interval graph. A minimal zero forcing set demonstrated.

adjacent to any vertex to the right of the maximal 2-clique, and so every path from u to the right of the maximal 2-clique passes through the maximal 2-clique. As these are trivial to study in the context of zero forcing, we will exclude the possibility of having maximal 2-cliques in the following proper interval graphs.

The edge spread of this family of proper interval graphs behaves much more uniformly than that of the q-paths in Section 2.1.

Theorem 2.8. *For an edge-disjoint proper interval graph, G, with no maximal 2-cliques,*

$$z_e(G) = 1 \;\forall e \in E(G).$$

Proof. Let G be an edge-disjoint proper interval graph. We will show that for any choice of edge, e, there is a smaller zero forcing set of $G - e$ than a minimum zero forcing set of G. Consider the zero forcing set, S, of G described by Theorem 2.2, with collection of forcing chains $\mathcal{F}$.

First, notice that when $e = (i, j)$ is an outer edge, then $G - e$ is a proper interval graph, as described above. Recall that the clique cover number of a proper interval graph is the number of outer edges. When e is removed, two new outer edges appear in its place. This means that removing e replaces one outer edge in G with two in $G - e$, and so $cc(G - e) = cc(G) + 1$. Therefore,

$$\begin{aligned} Z(G-e) &= n - cc(G-e) \\ &= n - cc(G) - 1 \\ &= Z(G) - 1. \end{aligned}$$

This gives us that

$$z_e(G) = 1.$$

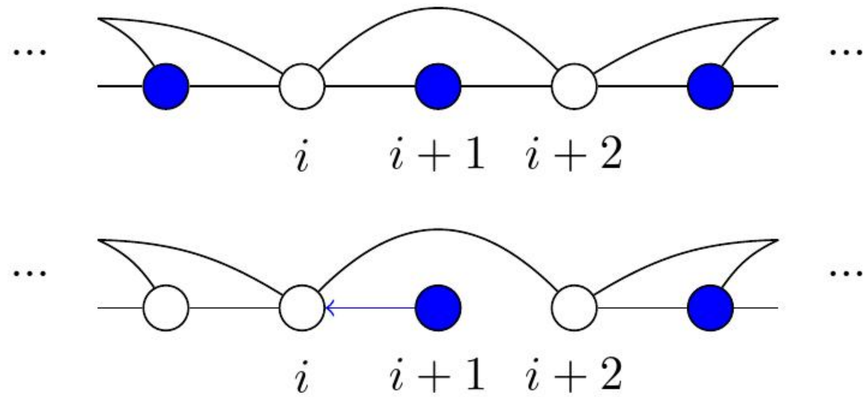


Figure 2.5: Demonstrating how to adjust the initial set of vertices if an inner edge is removed from a 3-clique.

Now suppose that e is an inner edge of G.

If the first (or last, by symmetry) clique of G is of size 3 and $e = (2, 3)$ (or $e = (n-2, n-1)$ in the symmetric case), then $G - e$ is a proper interval graph with two 2-cliques in the place of the 3-clique in G. This means that $cc(G-e) = cc(G)+1$, and therefore

$$Z(G - e) = Z(G) - 1,$$

as above. So again, we see that

$$z_e(G) = 1.$$

Let e be any other inner edge. Suppose e is in a clique of size 3, $\{i, i+1, i+2\}$. WLOG, assume that $e = (i+1, i+2)$. (If it isn't, reverse the ordering of the vertices and it will be.). Recall that S is the forcing set of G described by Theorem 2.2 with collection of forcing chains $\mathcal{F}$. Since there are no maximal 2-cliques, there is a unique maximal clique containing $i-1$, call this clique C. Clique C also contains i, so $i-1$ is not the final vertex in C. Therefore, by the construction of S, $i-1 \in S$.

Let $S' = S - \{i-1\}$, as in Figure 2.5. We will show that S' is a zero forcing set for $G - e$. Let w be the first vertex in the C. The sequence of forces are the same as those of $\mathcal{F}$ until w is forced. Then $i+1$ forces i, as it only has one neighbour. After i is forced, w only has one white neighbour, being $i-1$, so $w \to i-1$. Then the only white neighbour of i remaining is $i+2$, so $i \to i+2$, and the remaining vertices will be forced as in the proof of Theorem 2.2. Therefore,

$$Z(G-e) \leq Z(G) - 1,$$

when e is in a 3-clique. From Theorem 1.3, we know that the zero forcing number of any subgraph formed by the removal of one edge can differ from the zero forcing number of the original graph by at most 1. With this, we see that,

$$Z(G-e) = Z(G) - 1.$$

Now, suppose that $e = (u, v)$, $u < v$, is an inner edge in a clique, C, of size $k \geq 4$. By reversing the ordering of G, if necessary, we can ensure that e is not incident with the last vertex in C. Therefore, we can assume that $v \in S$, where S is the forcing set of G described by Theorem 2.2 with collection of forcing chains $\mathcal{F}$. Now consider $S' = S - \{v\}$. Let the first vertex of C be w.

The sequence of forces of S' are the same as those in $\mathcal{F}$, until w is forced. See Figure 2.6 for an example of the following cases.

If $u = w$, then it is adjacent to vertices in the previous clique, which are all coloured blue, and all vertices in C except for v. So its only white neighbour is the last vertex in C, $u+k-1$. Therefore, $u \to u+k-1$ along the top edge of the clique. Once $u+k-1$ is forced, any other vertex in the clique, aside from u, will have v as its only white neighbour, and will therefore force v.

If $u \neq w$, then u is adjacent to all vertices in C, aside from v, so its only white neighbour is the final vertex of the clique. Therefore u forces this final vertex. Then, w is adjacent to the vertices in the previous clique, which are all coloured blue, and all vertices in the clique containing e, so v is its only white neighbour. So $w \to v$.

Once v is forced, the entire clique is forced, and the remaining vertices will be forced as in the proof of Theorem 2.2. So, $S - \{v\}$ is a zero forcing set of $G - e$ of size $Z(G) - 1$. Therefore,

$$Z(G-e) = Z(G) - 1$$

and so, we have proven that

$$z_e(G) = 1, \forall e \in E(G)$$

□

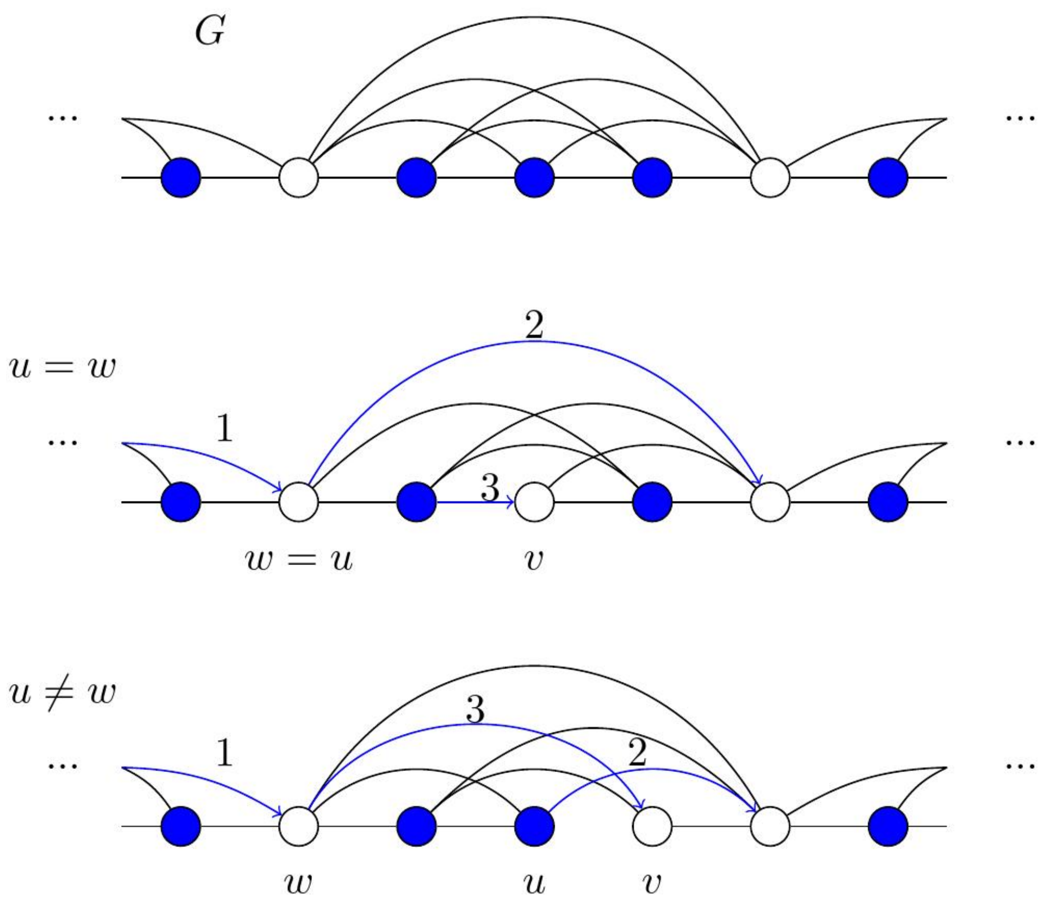


Figure 2.6: Examples of adjusted initial sets of vertices if an inner edge is removed from a k-clique, for $k = 5$. The first is when $u = w$, the second is when $u \neq w$. In both cases, arrows are drawn on edges where forces occur, and labeled in the order of chronological forces. Once these forces are complete, the forcing chain continues as in $\mathcal{F}$.

Chapter 3

Twisted Hypercubes

One of the first families of graphs whose zero forcing number was studied was the hypercube. Hypercubes have a recursive construction that relates to certain zero forcing results very nicely.

Definition 18. A *hypercube of dimension* k, Q_k, has vertex set $\{0,1\}^k$, with vertices adjacent when they differ in exactly one coordinate.

Or, the constructive definition:

Let Q_0 be a single vertex. For $k \geq 1$, Q_k is formed by taking two copies of Q_{k-1} and adding a matching joining the corresponding vertices in the two copies. This is equivalent to taking the Cartesian product of Q_{k-1} and K_2 to form Q_k.

So the hypercube has a convenient construction that only uses a Cartesian product of a lower dimensional hypercube and a copy of K_2. In [1], the AIM Minimum Rank-Special Graphs Work Group proves the following result about the relationship between the Cartesian product of two graphs, and the zero forcing number of their product:

Theorem 3.1. *[1] Zero Forcing Number of the Cartesian Product.*

Let G and H be two non-empty graphs. Then

$$Z(G\square H) \leq \min\{Z(G)|H|, |G|Z(H)\}$$

Sketch of Proof: Consider a minimal zero forcing set, S, for the graph G, with forcing chain $\mathcal{F}$. If we take the same set in each of the $|H|$ copies of G in $G\square H$, then each of the vertices in each copy of G are only adjacent to their corresponding vertex in every other copy of G. This means that no vertex in the initial set in $G\square H$ is adjacent to any extra white vertices, and so the forcing chain in each copy of G will successfully follow the forces in $\mathcal{F}$.

□

This result, along with the fact that Q_1, the one-dimensional hypercube, or K_2, has zero forcing number $Z(Q_1) = Z(K_2) = 1$, gives that the zero forcing number of any hypercube is bounded above by

$$Z(Q_k) \leq 2^{k-1}$$

Recall from Chapter 1.2, that the original purpose of the zero forcing number as a graph parameter was to act as an upper bound for the maximum nullity of the graph. The AIM Minimum Rank-Special Graphs Work Group had the insight to use, instead, the maximum nullity of particular families of graphs as a lower bound for the zero forcing number. Thus, began the study of the zero forcing number of graphs in their own right. In their original paper [1], they show that the maximum nullity of a k-dimensional hypercube is also at least 2^{k-1}, concluding that the zero forcing number of hypercubes is, in fact, equal to 2^{k-1}:

Theorem 3.2. *[1] Maximum Nullity of the Hypercube*
If Q_k is a k-dimensional hypercube, then

$$M(Q_k) \geq 2^{k-1}$$

Sketch of Proof: The proof given in [1] uses the following recursive definitions of sequences of block matrices.

$$H_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, L_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

For $k \geq 2$,

$$H_k = \begin{bmatrix} L_{k-1} & I \\ I & L_{k-1} \end{bmatrix}, L_k = \frac{1}{\sqrt{2}} \begin{bmatrix} L_{k-1} & I \\ I & -L_{k-1} \end{bmatrix}.$$

So H_k is an adjacency matrix for Q_k, and

$$\begin{bmatrix} I & 0 \\ -L_{k-1} & I \end{bmatrix} H_k = \begin{bmatrix} L_{k-1} & I \\ 0 & 0 \end{bmatrix}. \tag{3.1}$$

As the left matrix in equation 3.1 is an invertible matrix, it acts on H_k as performing standard row operations. Because the result is a matrix with rank 2^{k-1}, this shows that the minimum rank of H_k is at most 2^{k-1}. This gives us a bound on the maximum nullity of the graph,

$$\begin{aligned} M(Q_k) &= |Q_k| - m(Q_k) \\ &\geq 2^k - 2^{k-1} \\ &= 2^{k-1}. \end{aligned}$$

□

It follows, from Theorem 1.1 and the above upper bound on the zero forcing number of the hypercube, that

$$2^{k-1} \leq M(Q_k) \leq Z(Q_k) \leq 2^{k-1}.$$

So,

$$Z(Q_k) = 2^{n-1}.$$

Another family of graphs closely related to the hypercube is the twisted hypercube family. These graphs are defined as follows:

Definition 19. The unique twisted hypercube of dimension 0 consists of a single vertex. For $k \geq 1$, a *twisted hypercube of dimension* k is obtained from two twisted hypercubes of dimension $k-1$ by adding a matching joining the vertex sets of the two smaller graphs.

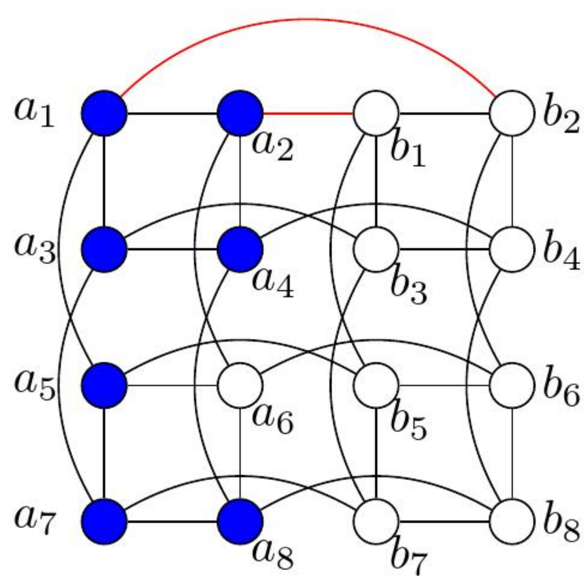
Definition 20. The randomly twisted hypercube of dimension 0, $\hat{Q}_0$, consists of a single vertex. For $k \geq 1$, the *randomly twisted hypercube of dimension* k, $\hat{Q}_k$, is formed from two disjoint, independently generated, random $(k-1)$-dimensional twisted hypercubes, $\hat{Q}_{k-1}$ and $\hat{Q}'_{k-1}$, by adding a random matching joining their vertex sets.

So hypercubes are actually a special case of the twisted hypercube. This leads to the natural follow-up question to Theorem 3.2:

Do twisted hypercubes of dimension k also have zero forcing number 2^{k-1}? The answer to the above question is no. For any twisted hypercube of dimension k, $\hat{Q}_k$, it is easy to see that the zero forcing number cannot be more than 2^{k-1}. If $\hat{Q}_k$ is formed by matching the vertices of $\hat{Q}_{k-1}$ to those of $\hat{Q}'_{k-1}$, then taking the initial set to be $\hat{Q}_{k-1}$ is a zero forcing set. This is because each vertex in $\hat{Q}_{k-1}$ will be adjacent to exactly one white vertex, namely the vertex in $\hat{Q}'_{k-1}$ to which it was matched. As each vertex in $\hat{Q}'_{k-1}$ is matched to a vertex in $\hat{Q}_{k-1}$, each vertex will be forced by one of the vertices in the initial set.

So no twisted hypercube can have zero forcing number greater than the standard hypercube, but for $k > 3$, it is possible to find, using a brute force computer algorithm, examples of twisted hypercubes of dimension k with zero forcing number less than 2^{k-1}. See Appendix B for this algorithm.

Example 3. Consider the following twisted hypercube of dimension 4 with a single twist, indicated in red. This twisted hypercube consists of two standard 3-dimensional twisted hypercubes with vertices $\{a_1, \ldots, a_8\}$ and $\{b_1 \ldots, b_8\}$. The matching between these two copies of Q_3 is a_1 to b_2, a_2 to b_1, and a_i to b_i for $3 \leq i \leq 8$.



This twisted hypercube has a zero forcing set of size $2^3 - 1 = 7$, consisting of the initial set indicated, $S = \{a_1, a_2, a_3, a_4, a_5, a_7, a_8\}$. A forcing chain for this zero forcing set is

$$\begin{aligned}\mathcal{F} = \{&a_1 \to b_2,\\ &a_3 \to b_3,\\ &a_4 \to b_4 \to b_8,\\ &a_7 \to b_7 \to b_5,\\ &a_8 \to a_6 \to b_6,\\ &a_2 \to b_1\}.\end{aligned}$$

From this example and Theorem 3.1, we can see that there must be twisted hypercubes of dimension $k \geq 4$ with zero forcing number at most $7 \cdot 2^{k-4} = 2^{k-1}(1 - 2^{-3})$, which leads to the first result on the zero forcing number of twisted hypercubes.

Theorem 3.3. *There exist infinite families of twisted hypercubes with zero forcing number*

$$\begin{aligned}Z(\hat{Q}_k) &\leq 2^{k-1}(1 - 2^{-3})\\ &= Z(Q_k)(1 - 2^{-3}),\end{aligned}$$

for $k \geq 4$.

Proof. We will prove this by induction on the dimension of the twisted hypercubes.

Base Case: Consider the twisted hypercube outlined in Example 3, call it $\hat{Q}_4$. Recall that $Z(Q_k) = 2^{k-1}$. This twisted hypercube has zero forcing number

$$\begin{aligned}Z(\hat{Q}_4) &= 7\\ &= 2^{4-1} - 1\\ &= 2^3(1 - 2^{-3})\\ &= Z(Q_4)(1 - 2^{-3}).\end{aligned}$$

Induction Step: Let $\hat{Q}_k$ be a k-dimensional twisted hypercube with zero forcing number less than Q_k,

$$Z(\hat{Q}_k) = 2^{k-1}(1 - 2^{-3}).$$

Now consider a twisted hypercube of dimension $k+1$, $\hat{Q}_{k+1} = \hat{Q}_k \square K_2$.

From Theorem 3.1, we see that

$$\begin{aligned} Z(\hat{Q}_{k+1}) &= Z(\hat{Q}_k \square K_2) \\ &\leq Z(\hat{Q}_k)|K_2| \\ &= 2^{k-1}(1 - 2^{-3}) \cdot 2 \\ &= 2^k(1 - 2^{-3}) \\ &= Z(Q_k)(1 - 2^{-3}). \end{aligned}$$

So, taking the Cartesian product of this $\hat{Q}_4$ with K_2 results in a twisted hypercube with zero forcing number smaller than that of Q_5, and taking the Cartesian product of the resulting graphs with K_2 results in larger and larger twisted hypercubes with smaller zero forcing number than the corresponding hypercubes. □

Now we can ask whether these families of twisted hypercubes have the smallest zero forcing number among all twisted hypercubes? Again, the answer to this question is no. Using the same algorithm as that used to find the 4-dimensional twisted hypercube with zero forcing number 7, we was able to find, for twisted hypercubes of dimension 5 and 6, zero forcing sets of size 13 and 25 respectively. See Appendix A for examples. Notice that 13 is double the minimum zero forcing number of a twisted hypercube of dimension 4, minus 1. We can write this as

$$\begin{aligned} Z(\hat{Q}_5) &\leq 2(2^3(1 - 2^{-3})) - 1 \\ &= 2^4 - 2 - 1 \\ &= 2^4(1 - 2^{-3} - 2^{-4}). \end{aligned}$$

Using the same method as Theorem 3.3, we can then find an infinite family of twisted hypercubes of dimension $k \geq 4$ with zero forcing number

$$Z(\hat{Q}_k) = 2^{k-1}(1 - 2^{-3} - 2^{-4}).$$

Similarly, 25 is double 13 minus 1, which we can express as

$$\begin{aligned} Z(\hat{Q}_6) &\leq 2(2^4(1 - 2^{-3} - 2^{-4})) - 1 \\ &= 2^5 - 2^2 - 2 - 1 \\ &= 2^5(1 - 2^{-3} - 2^{-4} - 2^{-5}), \end{aligned}$$

which results in an infinite family of twisted hypercubes of dimension $k \geq 5$ such that

$$Z(\hat{Q}_k) = 2^{k-1}(1 - 2^{-3} - 2^{-4} - 2^{-5}).$$

This short sequence seems to fit a pattern of doubling the zero forcing number and subtracting one, whenever the dimension is increased after $k = 3$. We believe this pattern will continue, and pose the following conjecture:

Conjecture 1. There exists a family of twisted hypercubes of dimension $k \geq 4$, $\hat{Q}_k$, such that

$$Z(\hat{Q}_k) \leq 2^{k-1}(1 - 2^{-3} - 2^{-4} - \cdots - 2^{-(k-1)}).$$

As well as these examples with small zero forcing numbers, I have found examples of twisted hypercubes of dimension 4 with zero forcing number 8, twisted hypercubes of dimension 5 with zero forcing number 14, and 15, and twisted hypercubes of dimension 6 with zero forcing number varying between the currently observed minimum, 25, and the value of the 6 dimensional hypercube, 32. So, being a twisted hypercube will not guarantee that the zero forcing number of the graph will be the minimum possible, and in fact, it appears as though every value for the zero forcing number between the minimum and 2^{k-1} is attainable for some twisted hypercube of dimension k.

For $k \leq 3$, however, the zero forcing number of any twisted hypercube matches the zero forcing number of the untwisted hypercube. This can be proven exhaustively as there are relatively few hypercubes of dimension less than 4.

Theorem 3.4. *For $k \leq 3$,*

$$Z(\hat{Q}_k) = Z(Q_k).$$

Proof. For $k = 1$ and $k = 2$ there is exactly one twisted hypercube, up to isomorphism. These being,

$$Q_1 \cong K_2,$$
$$Q_2 \cong C_4,$$

and we know their zero forcing numbers, from [1], are

$$Z(K_2) = 1,$$
$$Z(C_4) = 2.$$

Since these graphs are the same, up to isomorphism, as Q_1 and Q_2 with the same zero forcing number, the only graphs to check are the different copies of $\hat{Q}_3$.

There are a number of twists possible between two copies of a four cycle, but many of these are the same up to isomorphism. To represent these different twists, I will use standard permutation notation.

Let Q_2 and Q_2' be two copies of the 2-dimensional hypercube with vertex sets $\{1, 2, 3, 4\}$ and $\{1', 2', 3', 4'\}$ respectively. The matching that matches vertex 1 in the first copy of Q_2 to vertex $2'$ in the second copy of Q_2, matches 2 to $3'$, 3 to $4'$, and 4 to $1'$ would be

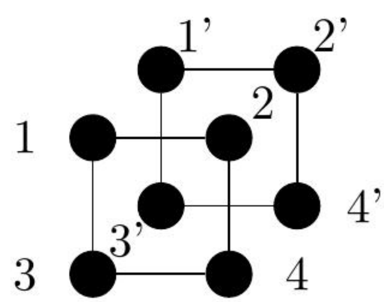
$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}.$$

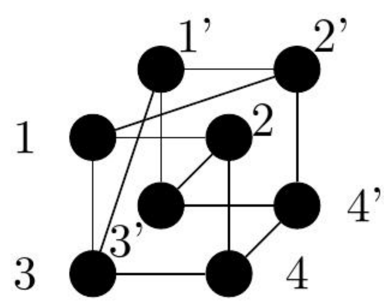
Because both of these graphs are cycles, the permutations of the vertices will result in many graphs isomorphic to each other. WLOG, assume that vertex 4 matches to $4'$, so there will be $3! = 6$ different permutations to study.

The first permutations I will consider are the following:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 1 & 4 \end{pmatrix}.$$

The first is the identity, and results in a Q_3, so can be ignored. The 3-cycle results in a graph not isomorphic to Q_3, but a brute force check confirms that there are no zero forcing sets of size < 4.

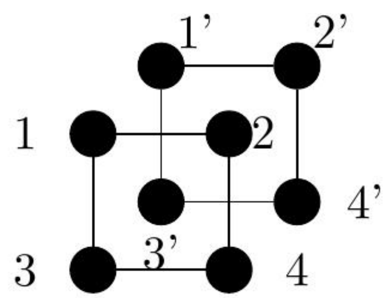


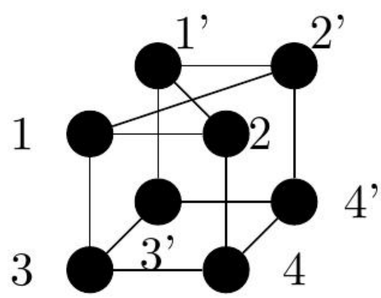


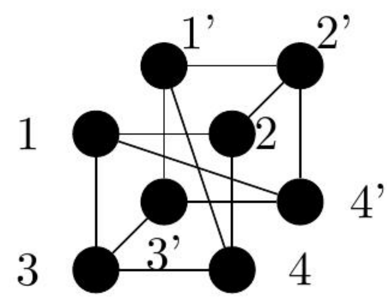
The next set of permutations are the single twists:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 3 & 1 \end{pmatrix}.$$

These permutations also do not result in a graph isomorphic to Q_3, the first results in a graph which is isomorphic to the above 3-cycle generated graph, so has already been checked. Again, a brute force check shows that no initial set of three vertices is a zero forcing set for the second permutation.



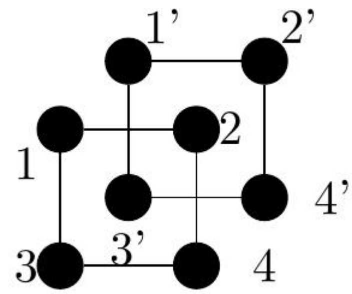


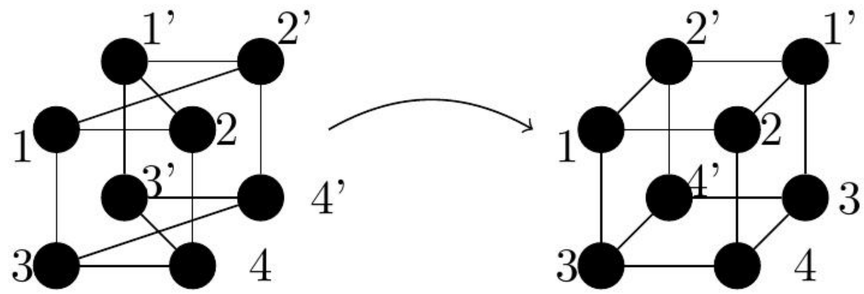


The final permutations to consider are the double twists:

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 1 \end{pmatrix}.$$

The first of these permutations also results in graphs isomorphic to Q_3, and so has zero forcing number 4. The second results in a graph isomorphic to the second single twist, and so has already been checked.





So, of all unique matchings joining the two copies of Q_2, only two resulted in unique 3-dimensional twisted hypercubes, which have no zero forcing sets of size less than 4. Therefore, the zero forcing number of any 3-dimensional twisted hypercube, $\hat{Q}_3$, is 4. □

Therefore,

$$Z(\hat{Q}_3) = 4$$

for all possible twisted hypercubes of dimension 3. However, the more immediately useful result from Theorem 3.4 is that there are only 3 possible 3-dimensional twisted hypercubes. Denote these twisted hypercubes as $\{Q_3, \hat{Q}_3, \hat{Q}_3'\}$, where Q_3 is the 3-dimensional hypercube. This gives us a way to, more efficiently, exhaustively check all possible hypercubes of dimension 4.

There are six ways to choose which of the 3 copies of $\hat{Q}_3$ to use to create our $\hat{Q}_4$, since we can decide to use the same copy twice, and order doesn't matter. So, choosing option 1 and option 3 is the same as choosing option 3 and option 1. This gives the possible pairings of twisted hypercubes as

$$\begin{array}{lll} Q_3 \to Q_3 & \hat{Q}_3 \to \hat{Q}_3 & \hat{Q}_3' \to \hat{Q}_3' \\ Q_3 \to \hat{Q}_3 & \hat{Q}_3 \to \hat{Q}_3' & \\ Q_3 \to \hat{Q}_3' & & \end{array}$$

For each of these six possible pairs, there are 8! possible ways to match the vertices of one $\hat{Q}_3$ to the other. This gives a total count of 4-dimensional twisted hypercubes we can generate to be

$$6 \cdot 8! = 241920.$$

Many of these twisted hypercubes are isomorphic. I attempted to determine the number of non-isomorphic twisted hypercubes, but the program to determine the set of unique twisted hypercubes did not finish after a week of checking.

While this is still too large a set to realistically analyse by hand, it is within the grasp of brute force computer computation to sort out. I have found that of the 241920 possible twisted hypercubes of dimension 4, 234480 have zero forcing number 7, and the remaining 7440 4-dimensional twisted hypercubes have zero forcing number 8. This gives us that 96.9% of of these twisted hypercubes have zero forcing number 7, and the remaining 3.1% have zero forcing number 8. Interestingly, this means that there are thousands of generated examples of 4-dimensional twisted hypercubes with zero forcing number 8, many of which are not isomorphic to the standard hypercube. Therefore, being a twisted hypercube is not sufficient to have zero forcing number less than that of the standard hypercube of the same size.

While this method of brute force computation is viable for these small examples of twisted hypercubes, it is already pushing the limits of what can be efficiently done. Once we step up to 5 dimensions, we've already reached the point of being way too vast to compute in a lifetime. There are

$$\sum_{i=1}^{241920} i = 29,262,764,160$$

different ways to choose a pair of 4-dimensional twisted hypercubes, by the same reasoning as above. For each pair, there are 16! different matchings between the two graphs. Considering that it took my computer roughly a day to generate all possible $\hat{Q}_4$'s and find each zero forcing number, trying to use the same method for $\hat{Q}_5$ is not realistic. So, more general techniques will be required in order to analyse higher dimensional twisted hypercubes.

As indicated in the proof of Theorem 3.2, the adjacency matrices of hypercubes have a nice structure and can be used to study their algebraic properties. I hoped to use this same property to find similar results for the zero forcing number of twisted hypercubes as the zero forcing number of hypercubes. As with hypercubes, twisted hypercubes have a nice block matrix form for their adjacency matrices, which can be defined recursively.

Let

$$T_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Assuming we are given T_{k-1} and T'_{k-1} define

$$T_k = \begin{bmatrix} T_{k-1} & P_{k-1} \\ P_{k-1}^T & T'_{k-1} \end{bmatrix},$$

where P_k is any permutation matrix of size $2^k \times 2^k$. Note that T_1 is an adjacency matrix for $\hat{Q}_1$, and so T_k is an adjacency matrix for $\hat{Q}_k$. My initial idea was to try to emulate the proof of Theorem 3.2, and use the fact that the zero forcing number is an upper bound for the maximum nullity, in reverse.

The Special Graphs Work Group uses the adjacency matrix of hypercubes to find a lower bound for the maximum nullity, and therefore a lower bound for the zero forcing number. In their case, the derived lower bound matched the previously known upper bound and gave equality. Immediately, we can see that for twisted hypercubes, a general equality statement is impossible. This is because there are different possible twisted hypercubes with the same dimension that have different zero forcing numbers. For $k > 3$, there will, at best, be a range of potential zero forcing numbers for the twisted hypercubes of a given dimension, k.

To bound the maximum nullity, the Special Graphs Work Group instead computed an upper bound for the minimum rank of the adjacency matrices, and used

$$M(G) = n - mr(G) \leq n - (rank(A)),$$

where A is an adjacency matrix for G.

Their proof hinged on the fact that the permutation used in the construction of hypercubes is the identity permutation, as well as the fact that the inverse of one of the adjacency matrices of Q_k, specifically L_k, is itself. This allowed them to compute the rank of the constructed matrices to be 2^{k-1}. In other words, they row-reduced the matrix with a specified set of operations that converted half of the rows to be zeroes.

In the case of twisted hypercubes, one would need to solve similar equations. The first difference is that the permutations describing the matchings between twisted hypercubes need not be the identity. The second is that the two graphs being matched are not necessarily the same. Finally, the row reduction algorithm cannot result in a matrix with rank 2^{k-1}, as we know that there exist twisted hypercubes with zero forcing number less than 2^{k-1}.

Firstly, we need to construct adjacency matrices for twisted hypercubes of dimension k, T_k. Then, we need to row reduce this matrix to show it has nullity equal to the smallest zero forcing number for twisted hypercubes of dimension k.

The construction of T_k will presumably follow the same recursive style as Theorem 3.2, being

$$T_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

$$T_k = c \begin{bmatrix} T_{k-1} & P_{k-1} \\ P_{k-1}^T & T_{k-1} \end{bmatrix}.$$

where $c \in \mathbb{R}$ is a scalar, and P_{k-1} is a permutation matrix of size 2^{k-1}.

Once constructed, the rank of T_k must be determined. I was unable to find a construction which guaranteed a rank lower than $2^k - 2$. This is not nearly low enough, as for $k = 4$, I have found zero forcing sets of size 7 for particular copies of $\hat{Q}_4$. This bounds the maximum nullity of T_4 as

$$M(T_4) \leq Z(\hat{Q}_4) = 7$$

From the maximum nullity, we can compute the minimum rank of the adjacency matrix

$$\begin{aligned} mr(T_4) &= n - M(T_4) \\ &\geq 2^4 - 7 \\ &= 9 \end{aligned}$$

Similarly, for $k = 5$ and $k = 6$, we need to demonstrate adjacency matrices with rank 19 and 39 respectively.

To this end, I pose the problem

Problem 1. Minimize the rank of the adjacency matrix of the twisted hypercube. For small values of k, show the following:

$$rank(T_4) = 9$$
$$rank(T_5) = 19$$
$$rank(T_6) = 39.$$

Chapter 4

Probabilistic Zero Forcing

Probabilistic Zero Forcing, introduced in [17], is a generalization of Zero Forcing where blue vertices can force their neighbours without the requirement that only one of their neighbours are white. Recall Definition 15, the probabilistic color change rule:

Definition 15: For a graph, G, consider a set of blue vertices, S. With respect to S, define the probabilistic colour change rule as follows:

Let $P(u \to v)$ be the probability that the event $u \to v$ occurs. Then

$$P(u \to v) = \begin{cases} \frac{|N[u] \cap S|}{|N(u)|}, & \text{if } u \in S \text{ and } v \in N(u) \cap S^c \\ 0 & \text{otherwise} \end{cases}$$

where u forces each of its white neighbours independently.

From this definition, we can see that if a blue vertex, u, has only one white neighbour, v, then $P(u \to v) = 1$, and the process reduces down to classical zero forcing. Also, for any connected graph, having a single initial vertex in S is sufficient for the graph to become fully forced in a finite number of iterations, with probability 1. This gives that the probabilistic zero forcing number of any graph is the number of connected components, and is not an interesting parameter to study. Rather, we can look at how long it takes to fully force a given graph.

This leads to the primary topic of study within probabilistic zero forcing, the expected propagation time. Recall Definition 16:

Definition 16: The expected propagation time of a connected graph G is the minimum of the expected propagation time from an initial vertex,

$$ept(G) = \min_{v \in V(G)} \{ept(G, \{v\})\}.$$

In this chapter, we will refer to one iteration of applying the probabilistic colour change rule to all blue vertices and their white neighbours as one turn. Then the expected propagation time of a graph is the expected number of turns it takes to colour all vertices blue.

Recall from Table 1.1 that the expected propagation time of a path is

$$ept(P_n) = \begin{cases} \frac{n}{2} + \frac{2}{3} & \text{if } n \text{ is even} \\ \frac{n}{2} + \frac{1}{2} & \text{if } n \text{ is odd.} \end{cases}$$

This result comes from taking the initial vertex to be the center of the graph, and once any force occurs, the process becomes a deterministic zero forcing process. Similarly, for the star graph in Theorem 1.8, the expected propagation time comes from beginning with the central vertex and forcing all of the leaves. For nearly all of the families of graphs that have been studied, the upper bounds on the expected propagation time are computed by taking the starting vertex in the center of the graph. While this is a natural candidate for expected propagation time, as these vertices will have the shortest paths to force the furthest vertices, is this always the best choice for any graph?

I will begin by outlining some simple experiments that I have run on a particular family of graphs, q-paths. Recall from Definition 4 that q-paths are a special type of q-tree where each new copy of K_q is attached to the next-most recently added K_q.

I will give evidence to support the theory that particular length q-paths may be faster to propagate when starting the probabilistic zero forcing process from outside the center of the graph. I will supplement this with theoretical computations to attempt to explain the relationship.

4.1 Experiments

When propagating a probabilistic zero forcing process, the first thing that must occur is the forcing of the entire closed neighbourhood of the initial vertex. This is because once the neighbourhood of the initial vertex is forced, then this set of blue vertices is a classical zero forcing set. Once our graph contains a classical zero forcing set, at least one vertex will be forced each turn with probability 1, and the limit on the number of turns remaining is the number of remaining white vertices. The worst case for this is when none of the neighbours of the initial vertex are adjacent. For this reason, we use the star bound, Theorem 1.8, to bound the expected propagation time to force this set. This results in the first forces of our process taking time dependent on the degree of the initial vertex.

The reason that q-paths are a viable choice of graph for which a starting vertex outside of the center is because the degrees of the vertices in the center of the graph are, generally, much higher than others. For example, for any q-tree with more than $2q$ vertices, the degree of the central vertex is $2q$, whereas the degree of the end vertices are only q. This is because each new vertex, v, added to the q-path is immediately connected to q other vertices. Then, the next q vertices added to the q-path will also be connected to v, giving it a degree of $2q$.

So, in theory, it will take much longer for a central vertex to force its entire neighbourhood at the beginning of the zero forcing process, rather than an end vertex. This could be significant because once this initial neighbourhood is forced, this will be a classical zero forcing set for P_n^q. Therefore, each turn will include at least one deterministic zero force, or a probabilistic force with probability 1. This gives a hard limit on the number of remaining turns in the propagation, being the number of remaining white vertices. So, if we can get to this state of the propagation process faster, then we may be able to force the entire graph faster.

Therefore, we are looking for q-paths that have central vertices with high enough degree that they are significantly slower to force their neighbourhoods, but are short enough so that the end vertices can force the opposite end of the graph before the central vertices can force their neighbourhoods.

For the purposes of these experiments, I will consider q-paths of length at least $q+2$, because q-paths of length q and $q+1$ are complete graphs, and all vertices are, therefore, central.

For each of the following values of q and n, I have run the probabilistic zero forcing process on the graph 10 000 times for each starting vertex and computed the average number of turns required to force the graph. See Appendix B for the associated code.

n	$ept(P_n^q, [1])$	$ept(P_n^q, [\lceil \frac{n}{2} \rceil])$	$e_{\text{diff}} = ept(P_n^q, [1]) - ept(P_n^q, [\lceil \frac{n}{2} \rceil])$
$q = 2$			
4	2.5569	2.5481	0.0088
5	3.0358	2.9501	0.0857

6	3.6832	3.312	0.3712
$q = 3$			
5	2.8594	2.8683	-0.0089
6	3.1791	3.1385	0.0406
7	3.5049	3.3741	0.1308
$q = 4$			
6	3.0869	3.0918	-0.0049
7	3.3356	3.2967	0.0389
8	3.5401	3.4925	0.0476
9	3.7612	3.6357	0.1255
$q = 5$			
7	3.2654	3.2704	-0.005
8	3.4281	3.4277	0.0004
9	3.5924	3.5551	0.0373
10	3.7462	3.6838	0.0624
11	3.9297	3.7968	0.1329
$q = 6$			
8	3.3823	3.3972	-0.0149
9	3.5321	3.5135	0.0186
10	3.661	3.6293	0.0317
11	3.779	3.7313	0.0477
12	3.8825	3.8507	0.0318
13	4.0542	3.9149	0.1393
$q = 7$			
9	3.5053	3.4944	0.0109
10	3.6081	3.6317	-0.0236
11	3.7224	3.7012	0.0212
12	3.8211	3.7772	0.0439
13	3.8997	3.855	0.0447
14	4.0285	3.9552	0.0733
15	4.1345	4.0026	0.1319

$q = 8$			
10	3.5732	3.6013	-0.0281
11	3.6729	3.6837	-0.0108
12	3.7665	3.7606	0.0059
13	3.8529	3.8288	0.0241
14	3.9453	3.8981	0.0472
15	4.0213	3.9538	0.0675
16	4.1205	4.0271	0.0934
$q = 9$			
11	3.6504	3.6613	-0.0109
12	3.7319	3.7127	0.192
13	3.8106	3.8069	0.0037
14	3.8784	3.865	0.0134
15	3.9568	3.9399	0.0169
16	4.041	3.9994	0.0416
17	4.1008	4.0724	0.0284
18	4.1857	4.1355	0.0502
19	4.279	4.1659	0.1131
$q = 10$			
12	3.726	3.7223	0.0037
13	3.7853	3.7813	0.004
14	3.8451	3.869	-0.0239
15	3.8981	3.9087	-0.0106
16	3.9976	3.9579	0.0397
17	4.0542	4.0195	0.0347
18	4.1117	4.0683	0.0434
19	4.1729	4.1101	0.0628
20	4.2384	4.1722	0.0662
21	4.3191	4.218	0.1011

As e_{diff} is the difference between the expected propagation times with initial vertex at the end of the graph, and the center of the graph, negative values would indicate that the end vertex had the faster propagation, on average. From this table, it would appear that, beginning at $q = 3$, we can find examples of graphs where e_{diff} is negative. This generally occurs when $n = q + 2$, the first q-path that is not a clique. For larger values of q, there are more examples of graphs where this occurs. As for $q = 7$, when $n = q + 2$, we see that e_{diff} is positive, and when $n = q + 3$ we get a negative e_{diff} value. Similarly, for $q = 10$, we get positive values of e_{diff} for $n = 12$ and 13, before getting negative values for both $n = 14$ and $n = 15$. In the following sections, we will develop bounds on the expected propagation times of q-trees according to different initial vertices. This will give a better idea of when we would expect negative e_{diff} values.

4.2 2-paths

For $q = 2$, I will first look at the expected propagation time of P_n^2 from vertex 1, the end vertex of the 2-path. This will be more straightforward to analyse because all forces will happen in the same direction. From the table in the previous section, we expect to see that, for 2-trees, beginning in the center of the graph will always be faster, on average.

Theorem 4.1. *Let P_n^2 be a 2-tree of order n. Then*

$$ept(P_n^2, \{1\}) = \frac{4}{7}n(1 + o(1)).$$

Proof. Let Z_t be the number of blue vertices in P_n^2 after turn t, and S_t the set of blue vertices after turn t. So $Z_t = |S_t|$. When $S_0 = \{1\}$, before the first turn,

$$P(1 \to 2) = P(1 \to 3) = \frac{1}{2}.$$

So, the first turn will consist of the events $\{1 \to 2\}, \{1 \to 3\}, \{1 \to 2 \cap 1 \to 3\}$, or the event that no forcing occurs, all with equal probability, $\frac{1}{4}$. If either $\{1 \to 2\}$ or $\{1 \to 2 \cap 1 \to 3\}$ occur on turn 1, then we will say that the graph is in State A, see Figure 4.1. Let the event that P_n^q is in State A be A. In general, we will say that the graph is in State A after turn t if $Z_t > 1$, and all vertices to the left of the rightmost vertex, v, are blue. See Figure 4.1 for an example.

Suppose the graph is in State A and the rightmost blue vertex is v. Then, vertex $v + 1$ will be forced on the next turn with probability 1 by vertex $v - 1$. The only

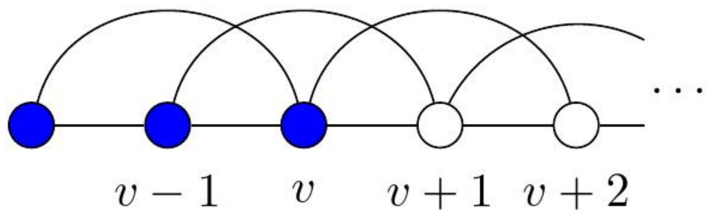


Figure 4.1: A 2-tree in State A

other vertex that can be forced on this turn is $v+2$, and this vertex can only be forced by v. As v has two blue neighbours and two white neighbours, $v_i \to v+2$ with probability

$$\begin{aligned} P(v \to v+2) &= \frac{|N[v] \cap S_t|}{|N(v)|} \\ &= \frac{3}{4}. \end{aligned}$$

Define the random variable Y as follows:

$$Y_t = \begin{cases} 1, & \text{if } v_i \to v+2 \\ 0, & \text{otherwise.} \end{cases}$$

Since Y_t is a Bernoulli random variable, $\mathbb{E}(Y_t) = \frac{3}{4}$.

We will assume, first, that the q-path is infinite.

Note that if the graph is in State A for turn t, $Z_t = Z_{t-1} + 1 + Y_t$. This gives us that $\mathbb{E}(Z_t|A, Z_{t-1}) = Z_{t-1} + \frac{7}{4}$. In other words, the expected number of vertices forced each turn when the graph is in State A is $\frac{7}{4}$. Also notice, when the graph is in State A, the only possibility after each turn is for the graph to remain in State A. So, we know the expected number of forces that will occur each turn once the graph is in State A, and therefore, we can use stopping times to determine when all of the vertices will be forced.

Define the random variable $W_t = Z_t - \frac{7}{4}t$. Then $W_0 = Z_0 = 1$, and

$$\begin{aligned} \mathbb{E}(W_{t+1}|Z_t) &= \mathbb{E}(Z_{t+1}|Z_t) - \frac{7}{4}(t+1) \\ &= Z_t - \frac{7}{4} + \frac{7}{4}(t+1) \\ &= Z_t + \frac{7}{4}t \\ &= W_t. \end{aligned}$$

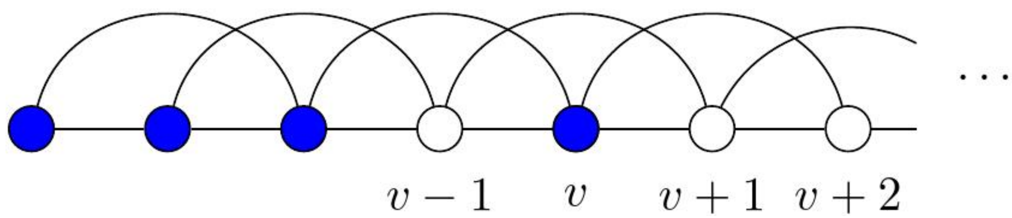


Figure 4.2: A 2-tree in State B

So, $\{W_t\}$ is a martingale with respect to $\{Z_t\}$. We will say that this process ends once $Z_t \geq n$, so we can define T to be the first time that $Z_T \geq n$. This is a stopping time for this process. Also, note that we can force at most two vertices per turn when the graph is in State A, so $Z_T = n$ or $n+1$. By the Stopping Time Theorem, Theorem 1.6, we find that

$$\begin{aligned}\mathbb{E}(W_T) &= W_0 \\ &= 1,\end{aligned}$$

and because we know that $W_T = Z_T - \frac{7}{4}T$, we can compute

$$\begin{aligned}\mathbb{E}(Z_T) &= \mathbb{E}(W_T) + \frac{7}{4}\mathbb{E}(T) \\ &= 1 + \frac{7}{4}\mathbb{E}(T).\end{aligned}$$

Therefore, we find

$$\frac{4}{7}n - \frac{4}{7} \leq \mathbb{E}(T) \leq \frac{4}{7}n.$$

This tells us that if the graph begins in State A, we would expect the graph to become fully forced in at most $\frac{4}{7}n$ turns.

If, on turn 1, the event $\{1 \to 3\}$ occurs, then I will say that the graph is in State B. Let the event that P_n^q is in State B be B. In general, I will say a graph is in State B after turn t if $|Z_t| > 1$, and all vertices to the left of v are blue, except for vertex $v-1$. Notice that since these are all possible sequences of forces on turn 1, and the rightmost vertex only has two white neighbours to its right, the graph is always in either State A or State B after the initial force. See Figure 4.2 for an example.

Now, suppose the graph is in State B. Then $v-1$ will be forced with probability 1 by vertex $v-2$ on this turn. Aside from this, v can force $v+1$ and $v+2$, both with probability $\frac{1}{2}$. These are all vertices that can be forced this turn. Define the random variables $Y_{t,1}$ and $Y_{t,2}$ as follows:

$$Y_{t,1} = \begin{cases} 1, & \text{if } v \to v+1 \\ 0, & \text{otherwise} \end{cases}, \qquad Y_{t,2} = \begin{cases} 1, & \text{if } v \to v+2 \\ 0, & \text{otherwise} \end{cases}$$

These are both Bernoulli random variables, so $\mathbb{E}(Y_{t,1}) = \mathbb{E}(Y_{t,2}) = \frac{1}{2}$. When the graph is in State B, $Z_t = Z_{t-1} + 1 + Y_1 + Y_2$. Then $\mathbb{E}(Z_t | B, Z_{t-1}) = Z_{t-1} + 2$, or the expected number of vertices forced each turn while the graph is in State B is 2. While in State B, the only way to remain in State B after a turn is when $v \to v+2$ and $v \nrightarrow v+1$. As these events are independent, the probability that this occurs is

$$P(Y_1^c \cap Y_2) = P(Y_1^c)P(Y_2) = \frac{1}{4}.$$

So there is a $\frac{1}{4}$ chance to remain in State B if the graph in State B. Otherwise, the graph will transition to State A. If the graph remains in State B after an application of the probabilistic color change rule, then exactly one sequence of forces must have occurred, $\{v-2 \to v-1 \cup v \to v+2\}$. Therefore, we know that if the graph is in State B before turn t, and remains in State B after turn t, then

$$Z_{t+1} = Z_t + 2,$$

or, exactly two vertices are forced if the graph remains in State B. Furthermore, if after turn 1, the graph is in State B, then $Z_1 = 2$, and if the graph is still in State B after turn t, then $Z_t = 2t$.

From the above transition probabilities, we can set up a simple Markov chain to describe the behaviour of the forcing set, as shown in Figure 4.3. Let $\{X_t\}$ be the sequence that says what state the graph is in at time t. Let $\{1\}$ be the event that only vertex 1 is in S_t. Then the transition matrix, P, for this Markov chain is given by

$$P = \begin{array}{c} \\ \{1\} \\ B \\ A \end{array} \!\!\begin{array}{c} \begin{array}{ccc} \{1\} & B & A \end{array} \\ \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \\ 0 & \frac{1}{4} & \frac{3}{4} \\ 0 & 0 & 1 \end{bmatrix} \end{array}.$$

As A is clearly the absorbing state for this Markov chain, we can compute the expected number of turns to transition to State A if we assume this is an infinite

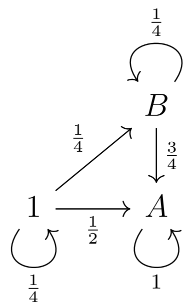


Figure 4.3: The Markov chain of forcing set states of P_n^2

process. We can use the hitting times from Definition 14 to compute the expected number of turns to transition from $\{1\}$ to A. Let $h_{i,j}$ be the hitting time from state i to state j. Then

$$\begin{aligned} h_{B,A} &= \sum_{i=1}^{\infty} iP(\{X_{t+i} = A\} \cap \bigcap_{j=0}^{i-1} \{X_{t+j} = B\} | X_t = B) \\ &= \sum_{i=1}^{\infty} i \left(\frac{3}{4}\right)\left(\frac{1}{4}\right)^{i-1} \\ &= 3\sum_{i=1}^{\infty} \frac{i}{4^i} \\ &= \frac{4}{3}, \end{aligned}$$

$$h_{1,A} = \frac{1}{2}(1) + \frac{1}{4}(1 + h_{1,A}) + \frac{1}{4}(1 + h_{B,A}).$$

Grouping like terms gives

$$\frac{3}{4}h_{1,A} = \frac{1}{2} + \frac{1}{4} + \frac{7}{12},$$

which solves to give

$$h_{1,A} = \frac{16}{9}.$$

So the expected time to hit State A when starting from $S_0 = \{1\}$ is $\frac{16}{9}$ turns.

Now, we separate the expected propagation time into two parts. Recall from above that T is the first time the sequence $Z_t \geq n$. Define T_1 to be the first time that the graph transitions into State A, or vertex n becomes blue.

Let U be the event that $Z_{T_1} \leq n$. So, U is the event that the graph transitions into State A before vertex $n-1$ turns blue. Then U^C is the event that the process finishes without transitioning to State A, namely, the graph is always in State B. If U^C occurs, we immediately know a few things:

- First, the total time to force the graph will be one more than the time to transition to State A. Once n is forced, there will need to be one more force, which happens with probability 1. So,

  $$T \leq T_1 + 1,$$

  since the graph does not transition to State A.

- It follows that

  $$\begin{aligned} \frac{n}{2} - 1 \leq \mathbb{E}(T|U^C) &= \mathbb{E}(T_1|U^C) + 1 \\ &\leq h_{1,B}|U^C + \frac{n-1}{2} + 1 \\ &\leq n \end{aligned}$$

  because we know that two vertices are forced each turn if the graph remains in State B, and $n-1$ vertices are blue at time T_1.

- Finally,

  $$\begin{aligned} P(U^C) &\leq \frac{1}{2}\left(\frac{1}{2}\right)^{n/2-1} \\ &= \left(\frac{1}{2}\right)^{n/2} \end{aligned}$$

  because we could be in either State $\{1\}$ or B, and the minimum probability of entering State A is $\frac{1}{2}$. Also,

  $$\begin{aligned} P(U^C) &\geq \frac{1}{2}\left(\frac{1}{4}\right)^{n/2-1} \\ &\geq \left(\frac{1}{4}\right)^{n/2} \end{aligned}$$

  because the minimum probability of not entering State A is $\frac{1}{4}$. This gives that

  $$\left(\frac{1}{2}\right)^{n} \leq P(U^C) \leq \left(\frac{1}{2}\right)^{n/2}.$$

From the Law of Total Probability (Definition 9), we can see that

$$\mathbb{E}(T_1) = P(U)\mathbb{E}(T_1|U) + P(U^C)\mathbb{E}(T_1|U^C).$$

Rearranging this equation, we can solve for $\mathbb{E}(T_1|U)$:

$$\begin{aligned}
\mathbb{E}(T_1|U) &= \frac{1}{P(U)}\left(\mathbb{E}(T_1) - P(U^C)\mathbb{E}(T_1|U^C)\right) \\
&\geq \frac{16}{9} - \left(\frac{1}{2}\right)^{n/2} n \\
&= \frac{16}{9}(1 - o(1)).
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}(T_1|U) &= \frac{1}{P(U)}\left(\mathbb{E}(T_1) - P(U^C)\mathbb{E}(T_1|U^C)\right) \\
&\leq \frac{1}{1 - \left(\frac{1}{2}\right)^{n/2}}\left(\frac{16}{9} - \left(\frac{1}{2}\right)^{n} n\right) \\
&= \frac{16}{9}(1 - o(1)).
\end{aligned}$$

This gives

$$\mathbb{E}(T_1|U) = \frac{16}{9}(1 - o(1)).$$

Next, we can determine the expected remaining time to force the graph once it is in State A, because we know how many vertices are forced each turn before reaching State A. Let T_2 the time it takes for the process to finish once in State A. Then $T = T_1 + T_2 < n - 1$.

As mentioned above,

$$0 \leq \mathbb{E}(T_2|U^C) \leq 1$$

because P_n^2 never achieves State A. If we do achieve State A, then by the stopping time argument above, we know the upper bound on the number of turns remaining to force the graph. It is $\frac{4}{7}$ times the number of white vertices remaining,

$$\mathbb{E}(T_2|U) = \frac{4}{7}\left(n - 2\mathbb{E}(T_1|U)\right).$$

This gives a total expected propagation time of

$$\begin{aligned}
\mathbb{E}(T|U) &= \mathbb{E}(T_1|U) + \mathbb{E}(T_2|U) \\
&= \mathbb{E}(T_1|U) + \frac{4}{7}(n - 2\mathbb{E}(T_1|U)) \\
&= \frac{4}{7}n - \frac{1}{7}\mathbb{E}(T_1|U) \\
&= \frac{4}{7}n - \frac{1}{7}\left(\frac{16}{9}(1 - o(1))\right) \\
&= \frac{4}{7}n(1 - o(1)).
\end{aligned}$$

Then

$$\begin{aligned}
\mathbb{E}(T) &= P(U)\mathbb{E}(T|U) + P(U^C)\mathbb{E}(T|U^C) \\
&\leq \frac{4}{7}n(1 - o(1)) + \left(\frac{1}{2}\right)^{n/2}(n+1) \\
&= \frac{4}{7}n(1 + o(1)),
\end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}(T) &= P(U)\mathbb{E}(T|U) + P(U^C)\mathbb{E}(T|U^C) \\
&\geq \frac{4}{7}n(1 - o(1)) + \left(\frac{1}{2}\right)^{n}\left(\frac{n}{2} - 1\right) \\
&= \frac{4}{7}n(1 + o(1)),
\end{aligned}$$

giving that $\mathbb{E}(T) = \frac{4}{7}n(1 + o(1))$

□

Now I will consider the case when the probabilistic zero forcing process begins in the center of the graph. The major difference between the initial vertex being in the center and at the end is that there are really two separate probabilistic zero forcing processes occurring simultaneously. The set of blue vertices will be expanding towards both ends of the q-path at the same time. The following lemma will, therefore, be of use.

Lemma 4.2. *Let v be a vertex in P_n^q, and $S_t \subset V(P_n^q)$ the subset of blue vertices after turn t. If $N[v] \in S_t$, then no vertex greater than v can force a vertex less than v.*

Proof. If v is in the first maximal clique of P_n^q, then $N[v]$ being in S_t implies that there are no vertices less than v remaining to be forced, and the result holds trivially. A symmetric argument holds for v in the last maximal clique.

So, assume v is not adjacent to 1 or n. We will argue for the vertices greater than v, and the symmetric argument will hold for vertices less than v. The smallest vertex greater than v that could possibly be forced is $v+q+1$, because $v+q \in N[v]$, which is assumed to be blue. Any vertex $u < v$ cannot be in the neighbourhood of $v+q+1$, because

$$\begin{aligned} |(v+q+1)-u| &= v+q+1-u \\ &> v+q+1-v \\ &= q+1. \end{aligned}$$

So no vertex smaller than v can force any vertex greater than v once its closed neighbourhood is completely forced. □

This lemma tells us that once the neighbourhood of the initial vertex is forced, then the graph is effectively split into two halves, with respect to probabilistic zero forcing. We can, then, consider the subgraphs induced by $\{1, \ldots, v\} \subset V(P_n^q)$, and $\{v, \ldots, n\}$ as two separate q-paths after $N[v]$ is all blue. Therefore, when the initial vertex in the forcing set is in the center, a similar computation can be used to determine the expected propagation time of $S_0 = \{\lceil \frac{n}{2} \rceil\}$ on P_n^2.

Theorem 4.3. *Let P_n^2 be a 2-tree of order n. Then*

$$ept(P_n^2, \{\left\lceil \frac{n}{2} \right\rceil\}) \leq \frac{2}{7}n(1+o(1))$$

Proof. Let Z_t be the number of blue vertices in P_n^2 after turn t and S_t the set of blue vertices after turn t. Let v be the vertex $\lceil \frac{n}{2} \rceil$. By Lemma 4.2, once $N[v]$ is entirely forced, the vertices on either side of v cannot interact with each other. We will refer to the subgraph induced by $\{1, \ldots, v\}$ as the left half of P_n^q, and the subgraph induced by $\{v, \ldots, n\}$ as the right half of P_n^q.

This naturally splits the probabilistic zero forcing process into two phases: the first is forcing $N[v]$, the second is forcing the remaining vertices. By Theorem 1.8, this first phase takes constant time, because the degree of v is fixed. If we then assume that $N[v]$ is forced in constant time, then the worst case is when no other vertices outside of $N[v]$ are forced. While we could define specific states that the graph could be in, note that in Theorem 4.1, the lowest expected number of forces came from State A, which is the equivalent state that both halves of P_n^q are in under this assumption. So assuming that P_n^2 is in State A will give an upper bound on the expected propagation time for the graph.

Define $T_{r,1}$ to be the time it takes to force $N[v]$, and $T_{r,2}$ to be the time to force the remaining vertices in the right half of P_n^2 once $N[v]$ is forced. Let T_r be the first time that all vertices in the right half of P_n^2 are forced. Then $T_r = T_{r,1} + T_{r,2}$. Define the analogous random variables for the left half, $T_l = T_{l,1} + T_{l,2}$. Recall from Theorem 4.1 that the stopping time for a 2-path once in State A is $\frac{4}{7}$ time the number of vertices remaining. Assume $N[v]$ is forced, and no other vertices are forced. Then each half of P_n^2 can be considered to be a 2-path on at most $\frac{n}{2}+1$ vertices where the probabilistic zero forcing process began with an end vertex, and is in State A. As 3 vertices from these subgraphs are blue, the expected time remaining to force each half of P_n^2 is at most

$$\begin{aligned}\mathbb{E}(T_{r,2}) &\leq \frac{4}{7}\left((\frac{n}{2}+1)-3\right)\\ &= \frac{2}{7}(n-4).\end{aligned}$$

Then, the total expected time to force the right half of the graph is

$$\begin{aligned}\mathbb{E}(T_r) &= \mathbb{E}(T_{r,1}) + \mathbb{E}(T_{r,2})\\ &\leq O(1) + \frac{2}{7}(n-4)\\ &= \frac{2}{7}n(1+o(1)).\end{aligned}$$

The symmetric argument shows that the expected time to force the left half of P_n^2 will also be $\frac{2}{7}n(1+o(1))$. This gives that the time it takes to force the graph P_n^2 is the time it takes for both halves of the graph to be forced. This is equal to the number of turns it takes the slower of the two halves to become forced.

Notice that T_r can take values from $\frac{n}{4}$ if all possible vertices are forced each turn, to $\frac{n}{2}$ if only the deterministic vertices are forced each turn. The former happens with probability $(\frac{3}{4})^{n/4}$, and the latter with probability $(\frac{1}{4})^{n/2}$. Notice that if the process ends after $\frac{n}{4}$ turns, then every turn consisted of two forces, and $2 \cdot \frac{n}{4} = \frac{n}{2}$ vertices were forced. If the process ends after $\frac{n}{4}+1$ turns, there were necessarily $\frac{n}{4}-1$ turns with two forces, and 2 turns with a single force. This combination of turns results in $2 \cdot (\frac{n}{4} - 1) = \frac{n}{2} - 2$ vertices forced on turns with two forces, and $1 \cdot 2 = 2$ vertices forced on turns with a single force, totaling $\frac{n}{2}$ vertices. In other words, every turn with two forces that is removed, must be replaced by two turns with a single force.

So, if the process ends in $\frac{n}{4} \leq \frac{n}{4} + k \leq \frac{n}{2}$ turns, then there are exactly $2k$ turns in which the extra vertex is not forced. As there are $\binom{\frac{n}{4}+k}{2k}$ possible choices for these $2k$ turns, the probability that this event occurs is

$$P(T_r = k) = \binom{\frac{n}{4} + k}{2k} (\frac{3}{4})^{n/4-k} (\frac{1}{4})^{2k}.$$

Define T to be the largest of T_r and T_l. So,

$$T = \max\{T_r, T_l\}.$$

By Theorem 1.7, we have an expression to compute the maximum value of these two random variables:

$$\begin{aligned}
\mathbb{E}(T) &= \mathbb{E}(\max\{T_r, T_l\}) \\
&= \sum_{x=0}^{\infty} P(T_r = x) \left(x + \sum_{y=x+1}^{\infty} P(T_l \geq y) \right) \\
&\leq \sum_{x=0}^{n/4} x P(T_r = x) + \sum_{x=0}^{n/4} P(T_r = x) \sum_{y=x+1}^{n/4} 1 \\
&\leq \mathbb{E}(T_r) + \sum_{x=0}^{n/4} \binom{\frac{n}{4} + x}{2x} \left(\frac{3}{4}\right)^{n/4-x} \left(\frac{1}{4}\right)^{2x} \left(\frac{n}{4}\right) \\
&\leq \mathbb{E}(T_r) + \left(\frac{n}{4 \cdot 2^n}\right) \sum_{x=0}^{n/4} \frac{n^x 3^{n/4}}{(2x)!}, \text{ because } \binom{n}{k} \leq \left(\frac{n^k}{k!}\right).
\end{aligned}$$

Recall from calculus, the series root test: Suppose we have a series $\sum_n a_n$. Define

$$L = \lim_{x \to \infty} \sqrt[x]{|a_x|}.$$

Then, if $L < 1$, the series converges. If $L > 1$, the series diverges. If $L = 1$, the series may converge or diverge.

Considering the series $\sum_{x=1}^{\infty} \frac{n^x}{(2x)!}$, we can see that

$$\begin{aligned} L &= \lim_{x\to\infty} \sqrt[x]{\frac{n^x 3^{n/4}}{(2x)!}} \\ &= \lim_{x\to\infty} \frac{n3^{n/4x}}{((2x)!)^{1/x}} \\ &= 0. \end{aligned}$$

So, for this series, $L < 1$ and therefore the series converges to some value, M.

Looking more closely at the final sum, we can see that

$$\begin{aligned} \left(\frac{n}{2^{n+2}}\right) \sum_{x=1}^{n/4} \frac{n^x 3^{n/4}}{(2x)!} &\leq \left(\frac{n3^{n/4}}{2^{n+2}}\right) \sum_{k=1}^{\infty} \frac{n^x}{(2x)!} \\ &\leq \left(\frac{n}{2^{n+2}}\right) M. \end{aligned}$$

So this term is simply $o(1)$, which gives a final expectation of T as

$$\begin{aligned} \mathbb{E}(T) &\leq \mathbb{E}(T_r) + \left(\frac{n}{2^{n+2}}\right) M \\ &\leq \frac{2}{7} n(1 + o(1)) + \left(\frac{n}{2^{n+2}}\right) M \\ &= \frac{2}{7} n(1 + o(1)). \end{aligned}$$

□

The preceding two theorems give an idea for the expected propagation time of P_n^2 with respect to two different starting vertices. As the time to force the neighbourhood of the initial vertex is so small in the case of 2-paths, it would appear that there is not a length for which beginning at the end of the graph results in a faster expected propagation time. This agrees with the experimental results in the previous section.

4.3 q-paths

Similar to 2-paths, We will begin by finding a bound for the expected propagation time when the initial vertex is an end vertex. Consider the set S_t of blue vertices in P_n^q after turn t. The first thing to notice is that once the initial clique is contained within S_t, this constitutes a classical zero forcing set. So, the graph will be in a state similar to those discussed in Chapter 4.2. I will call State A the event when the rightmost vertex of the set of blue vertices has no white vertices to its left. State B will be when the rightmost vertex has exactly one white neighbour to its left. There are many more potential states that the graphs P_n^q can attain, simply because of the fact that the graphs can transition out of State A. Since the rightmost blue vertex will have more than two white neighbours to its right, there will, on each turn, be a chance that the furthest neighbour from S_t will be forced, without all other vertices between becoming forced.

We can determine the expected number of vertices forced each turn while P_n^q is in State A. Consider the following lemma:

Lemma 4.4. *If P_n^q is in State A after turn t, then the expected number of vertices that will be forced on turn $t+1$ is at least $q-1+\frac{1}{2^{q-1}}$.*

Proof. Let S_t be the set of blue vertices in P_n^q after turn t, with rightmost vertex v. Every white neighbour of v, $\{v+1, v+2, ..., v+q\}$, has a chance to be forced on the next turn. To determine the probability that each of these neighbours becomes forced on turn $t+1$, consider the neighbour $v+i$. When $i=1$, then vertex $v+1$ is forced with probability 1. Now, consider the neighbour $v+i$, for $2 \leq i \leq q$.

Note that the number of blue neighbours of vertex $v+i$ is completely determined by i. The only blue neighbours can be to the left of $v+i$, and we know that v is the last blue vertex in S_t. So, of the q neighbours to the left of $v+i$, the $i-1$ vertices between v and $v+i$ are white, and the remaining $q-i+1$ neighbours are blue. Vertex $v+i$ will be forced by vertex $v-k$ with probability $\frac{q+k+1}{2q}$, and will not be forced with probability $\frac{q-k-1}{2q}$, for $0 \leq k \leq q-i$.

The probability that vertex $v+i$ is not forced on turn $t+1$ given that we are in State A is

$$P(\nrightarrow v+i|A) = \frac{q-1}{2q} \cdot \frac{q-2}{2q} \cdots \frac{i-1}{2q}$$

because each vertex in $\{v-q+i, \ldots, v\}$ can force $v+i$ independently. Then, the probability that $\rightarrow v+i$ on turn $t+1$ can be computed as

$$1 - P(\nrightarrow v+i|A),$$

which results in a probability of being forced on turn $t+1$ of

$$1 - \left(\frac{q-1}{2q} \cdot \frac{q-2}{2q} \cdots \frac{i-1}{2q}\right) = 1 - \frac{(q-1)!}{(i-2)!(2q)^{q-i+1}}.$$

Now, define

$$X_{t,i} = \begin{cases} 1, & \text{if} \to v+i \text{ on turn } t \\ 0, & \text{otherwise} \end{cases},$$

$$X_t = \sum_{i=1}^{q} X_{t,i}.$$

So, $X_{t+1,i}$ is the Bernoulli random variable that has probability of success equal to the probability that vertex $v+i$ is forced on turn $t+1$. This gives that X_{t+1} is the number of vertices forced, in total, on turn $t+1$. Taking the expectation of these variables yields:

$$\begin{aligned}
\mathbb{E}(X_{t+1,i}) &= 1 - \frac{(q-1)!}{(i-2)!(2q)^{q-i+1}} \\
&\geq 1 - \left(\frac{(q-1)^{q-i+1}}{(2q)^{q-i+1}}\right) \\
&\geq 1 - \left(\frac{1}{2}\right)^{q-i+1},
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}(X_{t+1}) &= \sum_{i=1}^{q} \mathbb{E}(X_{t+1,i}) \\
&\geq 1 + \sum_{i=2}^{q} \left(1 - \left(\frac{1}{2}\right)^{q-i+1}\right) \\
&= 1 + (q-1) - \sum_{i=2}^{q} \left(\frac{1}{2}\right)^{q-i+1} \\
&= q - \left(\frac{1}{2}\right)^{q+1} \sum_{i=2}^{q} 2^i \\
&= q - \left(1 - \frac{1}{2^{q-1}}\right) \\
&= q - 1 + \frac{1}{2^{q-1}}.
\end{aligned}$$

□

So if the graph is in State A, at least $q-1$ vertices would be expected to be forced on the next turn.

Based on the results in the previous section, we will assume that State A is the state with the lowest expected number of forces. Intuitively, this assumption makes sense, as once the neighbourhood of the initial vertex is forced, there is a subset of blue vertices that is in State A. So, every possible set of blue vertices can be thought of as a subset in State A, plus some vertices to the right of this set. Every vertex we add to the set in State A will increase the number of vertices that can be forced on the next turn, as well as increase the probabilities that its neighbours will be forced.

To support this assumption, we will show that adding any one vertex to a set in State A will increase the expected number of forces on the next turn. More precisely,

Lemma 4.5. *Let S_t be a set of blue vertices in State A, with v the rightmost vertex, and let C be the event that P_n^q has blue vertices $S'_t = S_t \cup \{v+j\}$ for $j \geq 1$. Let $Y_t = |S_t| - |S_{t-1}|$, and $Z_t = |S'_t| - |S'_{t-1}|$ so Y_t and Z_t are the numbers of vertices forced on turn t. Then,*

$$\mathbb{E}(Y_{t+1}) \leq \mathbb{E}(Z_{t+1}).$$

Proof. Let P_n^q have S_t as its set of blue vertices after t turns, and let v be the rightmost vertex in S_t. Define $S'_t = S_t \cup \{v+j\}$. When $j = 1$, then the set S'_t is still in State A, and will have the same number of expected forces. If $j \geq q+1$, then the only change in forcing probabilities is non-negative: every vertex that could be forced by S_t has the same or greater chance of being forced, and the neighbours of $v+j$ can now be forced as well. So this case is trivially true. Now, assume $2 \leq j \leq q$.

Define the random variables $X_{t,w}$, and X_t to be

$$X_{t,w} = \begin{cases} 1, & \text{if } \to w \text{ on turn } t \\ 0, & \text{otherwise} \end{cases}$$

and

$$X_t = \sum_{w \in V(P_n^q)} X_{t,w}$$

So, X_t is the expected number of vertices forced on turn t. In State A, as above, we see that

$$\begin{aligned} \mathbb{E}(X_{t+1}|A) &= \sum_{w \in V(P_n^q)} \mathbb{E}(X_{t+1,w}|A) \\ &= \sum_{i=1}^{q} P(\to v+i|A). \end{aligned} \tag{4.1}$$

Now, consider the vertex $v + j$, for $2 \leq j \leq q$. Let r be the number of white neighbours of vertex $v + j$. Then, $v + j$ has $2q - r$ blue neighbours. The probability that $v + j$ forces, or does not force, some neighbour, u, is

$$\begin{aligned} P(v + j \to u|C) &= \frac{2q - r + 1}{2q}, \\ P(v + j \nrightarrow u|C) &= \frac{r - 1}{2q}. \end{aligned}$$

Notice, that all vertices larger than $v + q$ cannot be forced by any vertex other than $v + j$, so the probability of any vertex, $v + i$ being forced on turn $t + 1$, for $q < i \leq q + j$ is just $\frac{2q-r+1}{2q}$. This quickly gives the expected number of vertices greater than $v + q$ to be forced on turn $t + 1$ given C as

$$\begin{aligned} \sum_{i=q+1}^{q+j} \mathbb{E}(X_{t+1,i}|C) &= \sum_{i=q+1}^{q+j} \frac{2q - r + 1}{2q} \\ &= j\left(\frac{2q - r + 1}{2q}\right). \end{aligned} \tag{4.2}$$

The probability that any blue neighbour of $v + j \in S_t'$ forces any of its neighbours increases when compared to A, because it will now have one more blue neighbour, being $v + j$. Its number of white neighbours will decrease by one, lowering the probability that its does not force its neighbours. This tells us that for a vertex $v + i$ for $1 < i \leq q$, $i \neq j$,

$$P(\nrightarrow v + i|C) \leq P(\nrightarrow v + i|A)\left(\frac{r - 1}{2q}\right),$$

because all blue neighbours of $v + i$ in S_t have a lower chance of not forcing $v + i$ in S_t', and vertex $v + j$ does not force $v + i$ with probability $\frac{r-1}{2q}$.

From this, we can see that on turn $t + 1$, a vertex $v + i$ for $1 < i \leq q$, $i \neq j$, will be forced with probability

$$\begin{aligned} P(\to v + i|C) &= 1 - P(\nrightarrow v + i|C) \\ &\geq 1 - P(\nrightarrow v + i|A)\left(\frac{r - 1}{2q}\right) \\ &\geq P(\to v + i|A). \end{aligned} \tag{4.3}$$

Then, we can consider the expected number of forces on the next turn conditional on C. We can see that

$$\begin{aligned}\mathbb{E}(X_{t+1}|C) &= \sum_{w\in V(P_n^q)} \mathbb{E}(X_{t+1,w}|C) \\ &= \sum_{i=1}^{j-1} P(\to v+i|A) + \sum_{i=j+1}^{q} P(\to v+i|A) + \sum_{i=q+1}^{q+j} \frac{2q-r+1}{2q}.\end{aligned} \tag{4.4}$$

Using the differences found in equations 4.1 and 4.4 gives us that

$$\begin{aligned}\mathbb{E}(X_{t+1}|C) - \mathbb{E}(X_{t+1}|A) = \sum_{i=1}^{j-1} P(\to v+i|C) + \sum_{i=j+1}^{q} P(\to v+i|C) \\ + \sum_{i=q+1}^{q+j} \frac{2q-r+1}{2q} - \left(\sum_{i=1}^{q} P(\to v+i|A)\right) \\ = \sum_{i=1}^{j-1}(P(\to v+i|C) - P(\to v+i|A)) + \sum_{i=j+1}^{q} (P(\to v+i|C) - P(\to v+i|A)) \\ + \sum_{i=q+1}^{q+j} \frac{2q-r+1}{2q} - P(\to v+j|A).\end{aligned} \tag{4.5}$$

By inequality 4.3, we can see that the first two summations in equation 4.5 are positive. So, to prove the result, it would suffice to show that

$$\sum_{i=q+1}^{q+j} \frac{2q-r+1}{2q} \geq P(\to v+j|A) \tag{4.6}$$

in order to show that the whole difference is positive, and therefore the expected number of forces would increase. To achieve this, note that we can calculate the value of r in terms of q and j because we know the structure of the set of blue vertices. In this case,

$$\begin{aligned} r &= 2q - (q-j+1) \\ &= q+j-1. \end{aligned}$$

Then, the above sum becomes

$$\begin{aligned}\sum_{i=q+1}^{q+j} \frac{2q-r+1}{2q} &= \sum_{i=q+1}^{q+j} \frac{q-j+2}{2q} \\ &= j\left(\frac{q-j+2}{2q}\right)\end{aligned}$$

which is a quadratic in j. We can solve for when this expression is equal to 1, as this will be greater than or equal to any probability, and we get,

$$\begin{aligned} 0 &= \frac{-j^2 + (q+2)j - 2q}{2q} \\ &= \frac{-(j-2)(j-q)}{2q}. \end{aligned}$$

This tells us that the sum from inequality 4.6 is equal to 1 when $j = 2$ or $j = q$, which are the extremal values that j can take in this context. As this quadratic has a negative leading term, all values between these two points must be at least as large as 1. Therefore, inequality (4.6) holds for all values of $2 \leq j \leq q$. This tells us that equation (4.5) must be non-negative, and therefore the expected number of forces increases with the addition of vertex $v + j$ to the set of blue vertices.

□

For values of $q > 2$, the expected propagation time of P_n^q, when starting with the initial set being the end vertex, can be bounded above by the following, if we assume that State A is the slowest possible forcing state:

Let P_n^q, $q > 2$, be a q-path of size n, and let $q = q(n)$ be a function of n. Then

$$ept(P_n^q, \{1\}) \leq O(\log(q)) + \frac{n-q-1}{q-1}(1 + o(1)).$$

Justification: By Theorem 1.8, the time it takes for $\{1\}$ to force its neighbourhood is $O(\log q)$. Let S_t be the set of blue vertices after turn t.

Since we are assuming State A is the worst case for expected number of forces, the graph being in State A acts as an upper bound on the expected propagation time. By Lemma 4.4, we expect at least $q - 1$ vertices to be forced each turn. Define $Z_t = |S_t|$, so Z_t is the number of blue vertices after turn t. Let T be the first time that $Z_T \geq n - q$. Then T is a stopping time for this process. If we let the expected number of forces in State A be the lower bound, then

$$\mathbb{E}(Z_{t+1}|Z_t) \geq Z_t + q - 1.$$

As in Theorem 4.1, define the random variable $Y_t = Z_t - (q-1)t$. This random variable is a submartingale, so by the stopping time theorem

$$\mathbb{E}(Y_T) \geq Y_0 = 1.$$

We know that at most q vertices can be forced while in State A. Therefore the value of Z_T must be

$$n \leq Z(T) \leq n + q - 1.$$

Now, we can solve for the expected value of T by using the definition of Y_t,

$$\mathbb{E}(Y_T) = \mathbb{E}(Z_T) - (q-1)\mathbb{E}(T).$$

Inserting the known values and rearranging for $\mathbb{E}(T)$ yields

$$\begin{aligned}
\mathbb{E}(T) &= \frac{1}{q-1}(\mathbb{E}(Z_T) - \mathbb{E}(Y_T)) \\
&\leq \frac{1}{q-1}(n + (q-1) - 1) \\
&= \frac{n}{q-1}(1 + o(1)).
\end{aligned}$$

So once P_n^q is in State A, we would expect the time it takes to finish the probabilistic zero forcing process to be at most $\frac{n'}{q-1}(1 + o(1))$ turns, were n' is the number of white vertices remaining once the graph reaches State A.

Since the graph cannot be in State A at the beginning of the probabilistic zero forcing process, the total expected time to completely force the graph would be at most the time it takes to force the initial neighbourhood of $\{1\}$ plus the expected time to force the remaining vertices if the graph were always in State A. More precisely

$$ept(P_n^q, \{1\}) \leq O(\log(q)) + \frac{n-q-1}{q-1}(1 + o(1)),$$

which concludes this discussion.

Now, considering the case when the probabilistic zero forcing process begins with an initial vertex in the center of a q-path, we can determine a similar bound on the expected propagation time under the same assumption, that State A is the slowest forcing state for P_n^q.

If our assumption holds, let P_n^q be a q-path of order n with $q > 2$. Then,

$$ept(P_n^q, \{\lfloor \frac{n}{2} \rfloor\}) \leq O(\log q) + \frac{n - 2q - 1}{2(q-1)}(1 + o(1)).$$

Justification: Let $v = \{\lfloor \frac{n}{2} \rfloor\}$. By Theorem 1.8, the time it takes v to force its neighbourhood is $O(\log 2q) = O(\log q)$. Once this initial closed neighbourhood is entirely forced, the halves of the graph, as defined in Theorem 4.3, will be in equivalent states to those discussed above. By our assumption, the slowest forcing state is State A, and the expected number of forces each turn will be at least $q-1$. Thus, this state can be used to bound the expected propagation time of one half of P_n^q, as follows.

Once the vertex v has no white neighbours remaining, there will be at most $\frac{n-2q-1}{2} = \frac{n-1}{2} - q$ white neighbours on one half of the graph. Each turn, there will be at least $q-1$ vertices expected to be forced on one half of the graph. As each half of P_n^q can be viewed as $P_{n/2}^q$ with initial vertex at the end, we can apply the above discussion to conclude that each half of P_n^q will be expected to be forced in at most

$$O(\log q) + \frac{n/2 - q - 1}{q-1} \leq O(\log q) + \frac{n-2q-1}{2(q-1)}$$

turns. As in Theorem 4.3, we now consider which half of the graph will finish forcing second. Once the slower of the two halves is completely forced is when the entire graph is forced. Let T_r be the time it takes for the right half of P_n^q to be forced, and T_l the time for the left half. Define

$$T = \max\{T_r, T_l\}.$$

To approximate the probabilities that T_r takes k turns, we will assume that on each turn, either q or $q-1$ vertices are forced, as any other events are increasingly rare, and this will give an upper bound on the expected propagation time. As the furthest vertex from v, $v+q$, is the least likely to be forced, we will assume that all vertices between $v+1$ and $v+q-1$ are forced, and only $v+q$ has a chance of not being forced each turn. Recall that $P(\to v+q|A) = \frac{q+1}{2q} > \frac{1}{2}$, so the probability that q vertices are forced each turn is greater than $\frac{1}{2}$. For simplicity, we will say that each event occurs with probability $\frac{1}{2}$.

Then, the event that the right half of P_n^q is forced in $\frac{n}{2q}$ turns occurs with probability $(\frac{1}{2})^{n/2q}$. If it is forced in $\frac{n}{2q}+1$ turns, then there are two turns in which there must have been less than q forces. This will occur with the same probability, but there will be $\binom{\frac{n}{2q}+1}{2}$ ways to arrange the two turns where less than q forces occur. This gives this event a probability of $\binom{\frac{n}{2q}+1}{2}(\frac{1}{2})^{n/2q+1}$ of occurring. This pattern continues until the right half of P_n^q is forced in $\frac{n}{q-1}$ turns, which happens with probability $(\frac{1}{2})^{\frac{n}{q-1}}$. Using these probabilities, we can apply Theorem 1.7 again to compute the expected value of T. So

$$\begin{aligned}
\mathbb{E}(T) &= \mathbb{E}(\max\{T_r, T_l\} \\
&= \sum_{x=0}^{\infty} P(T_r = x)\left(x + \sum_{y=x+1}^{\infty} P(T_l \geq y)\right) \\
&= \sum_{x=0}^{n/2q(q-1)} xP(T_r = x) + \sum_{x=0}^{n/2q(q-1)} P(T_r = x) \sum_{y=x+1}^{n/2q(q-1)} P(T_l \geq y) \\
&\leq \mathbb{E}(T_r) + \sum_{x=0}^{n/2q(q-1)} \binom{\frac{n}{2q}+x}{2x}\left(\frac{1}{2}\right)^{n/2q} \sum_{y=x+1}^{n/2q(q-1)} 1 \\
&\leq \mathbb{E}(T_r) + n\left(\frac{1}{2}\right)^{n/2q} \sum_{x=0}^{n/2q(q-1)} \left(\frac{n}{2(q-1)}\right)^{2x} \frac{1}{(2x)!}.
\end{aligned}$$

By the same series test argument as in Theorem 4.3, we can see for the series $\sum_{k=1}^{\infty}\left(\frac{n}{2(q-1)}\right)^{2x}\frac{1}{(2x)!}$ that

$$\begin{aligned}
L &= \lim_{x\to\infty} \sqrt[x]{\left(\frac{n}{2(q-1)}\right)^{2x}\frac{1}{(2x)!}} \\
&= \lim_{x\to\infty} \frac{n^2}{2^{2x}(q-1)^{2x}(2x!)^{1/x}} \\
&= 0.
\end{aligned}$$

So $L < 1$, and therefore the series converges to some value M. The final sum in the above equation is

$$\begin{aligned}
n\left(\frac{1}{2}\right)^{n/2q} \sum_{x=0}^{n/2q(q-1)} \left(\frac{n}{2(q-1)}\right)^{2x} \frac{1}{(2x)!} &\leq n\left(\frac{1}{2}\right)^{n/2q} \sum_{x=1}^{\infty)} \left(\frac{n}{2(q-1)}\right)^{2x} \frac{1}{(2x)!} \\
&\leq n\left(\frac{1}{2}\right)^{n/2q} M.
\end{aligned}$$

Which gives the expected value of T to be

$$\begin{aligned}
\mathbb{E}(T) &\leq \mathbb{E}(T_r) + n\left(\frac{1}{2}\right)^{n/2q} \sum_{x=0}^{n/2q(q-1)} \left(\frac{n}{2(q-1)}\right)^{2x} \frac{1}{(2x)!} \\
&\leq O(\log q) + \frac{n-2q-1}{2(q-1)} + n\left(\frac{1}{2}\right)^{n/2q} M \\
&= O(\log q) + \frac{n-2q-1}{2(q-1)}(1+o(1))
\end{aligned}$$

concluding the discussion.

Now we can return to the question of whether any q-path's expected propagation time benefits from starting at one end of the graph, rather than in the center of the graph. These conditional results, while they are only upper bounds, and no concrete conclusions can be drawn from them, would indicate that the slowest expected propagation time for initial vertex being an end vertex is slower than the slowest expected propagation time for initial vertex being a central vertex, provided n is large.

It would also appear that, for large q and small n, there is a chance that beginning at the end could be faster than beginning in the center, as increasing q theoretically slows down the central vertex propagation time more than the end vertex. This is because of the first phase in the probabilistic zero forcing process, where the initial vertex's neighbourhood must be forced. Any increase in the size of the neighbourhood of the end vertices is doubled for the neighbourhood of the central vertices.

Experimentally, I was able to find that for values of $q \geq 3$, there are q-paths where the expected propagation time could be faster when taking the initial vertex outside of the center of the graph. This was the case when q was relatively large when compared to n. Based on the results of this section, I conclude with a conjecture

Conjecture 2. Let P_n^q be a q-path on n vertices. If 1 is the first vertex, and v is a central vertex, then

$$ept(P_n^q, \{1\}) \leq ept(P_n^q, \{v\})$$

provided $q \gg \frac{n}{\log n}$

Chapter 5

Conclusion

This thesis was concerned with two variants of zero forcing. The first, now known as classical zero forcing, began life as a bound for another graph parameter, but has more recently been studied in its own right. We found the zero forcing number of proper interval graphs, and determined the effect of edge removal on the zero forcing number for this family of graphs. We used these results to find the expected zero forcing number of random subgraphs of these proper interval graphs.

We then turned our attention to twisted hypercubes, a variation on the hypercube family of graphs. We compared the zero forcing number of the hypercube to randomly twisted hypercubes, and were able to show that twisted hypercubes have smaller zero forcing number than hypercubes of the same dimension. Our results gave an upper bound on the zero forcing number of twisted hypercubes that is smaller than the zero forcing number of hypercubes, and led to Conjecture 1. This conjecture poses that there are twisted hypercubes of dimension $k \geq 4$ that have zero forcing number at most

$$2^{k-1}(1 - 2^{-3} - 2^{-4} - \cdots - 2^{-(k-1)}).$$

We know this to be true for $k = 4, 5, 6$.

The second variant on zero forcing that was addressed is probabilistic zero forcing. This is a probabilistic variant that reduces to classical zero forcing when enough vertices are forced. Within the study of probabilistic zero forcing, the primary parameter of interest is the expected propagation time. In Chapter 4, we gave empirical evidence to support that the standard method of computing expected propagation time does not always give optimal bounds. Historically, vertices in the center of the graph were thought to give the lowest upper bounds on propagation time, as they minimize distance to all other vertices. Upon testing propagation time of q-paths with various initial vertices, we found that there are cases where less time is required to force the graph from the end of the q-path, not the center.

We then, gave partial results for small values of q to support this claim. We also gave conditional results for general values of q that could help to explain this phenomenon.

Bibliography

[1] AIM Minimum Rank – Special Graphs Work Group. Zero forcing sets and the minimum rank of graphs. *Linear Algebra and its Applications*, 428(7):1628–1648, 2008.

[2] David Amos, Yair Caro, Randy Davila, and Ryan Pepper. Upper bounds on the k-forcing number of a graph. *Discrete Applied Mathematics*, 181:1–10, 2015.

[3] Francesco Barioli, Wayne Barrett, Shaun M. Fallat, H. Tracy Hall, Leslie Hogben, Bryan Shader, P. van den Driessche, and Hein van der Holst. Zero forcing parameters and minimum rank problems. *Linear Algebra and its Applications*, 433(2):401–411, 2010.

[4] Kirk Boyer, Boris Brimkov, Sean English, Daniela Ferrero, Ariel Keller, Rachel Kirsch, Michael Phillips, and Carolyn Reinhart. The zero forcing polynomial of a graph. *Discrete Applied Mathematics*, 258:35–48, 2019.

[5] Boris Brimkov, Caleb C. Fast, and Illya V. Hicks. Computational approaches for zero forcing and related problems. *European Journal of Operational Research*, 273(3):889–903, 2019.

[6] Daniel Burgarth, Domenico D'Alessandro, Leslie Hogben, Simone Severini, and Michael Young. Zero forcing, linear and quantum controllability for systems evolving on networks. *IEEE Transactions on Automatic Control*, 58(9):2349–2354, 2013.

[7] Yu Chan, Emelie Curl, Jesse Geneson, Leslie Hogben, Kevin Liu, Issac Odegard, and Michael S Ross. Using markov chains to determine expected propagation time for probabilistic zero forcing. *Electronic Journal of Linear Algebra*, 36:318–333, 2020.

[8] Randy Davila, Thomas Kalinowski, and Sudeep Stephen. A lower bound on the zero forcing number. *Discrete Applied Mathematics*, 250:363–367, 2018.

[9] Shannon Dillman and Franklin Kenter. Leaky forcing: A new variation of zero forcing. *arXiv preprint arXiv:1910.00168*, 2019.

[10] Christina J. Edholm, Leslie Hogben, My Huynh, Joshua LaGrange, and Darren D. Row. Vertex and edge spread of zero forcing number, maximum nullity, and minimum rank of a graph. *Linear Algebra and its Applications*, 436(12):4352–4372, 2012. Special Issue on Matrices Described by Patterns.

[11] Michael Evans and Jeffrey S. Rosenthal. *Probability and statistics: The Science of Uncertainty*. W.H. Freeman and Company, 2010.

[12] Jesse Geneson and Leslie Hogben. Expected propagation time for probabilistic zero forcing. *Australian Journal of Combinatorics*, 83(3):397–417, 2022.

[13] Michael Gentner, Lucia D. Penso, Dieter Rautenbach, and Uéverton S. Souza. Extremal values and bounds for the zero forcing number. *Discrete Applied Mathematics*, 214:196–200, 2016.

[14] Leslie Hogben. Minimum rank problems. *Linear Algebra and its Applications*, 432(8):1961–1974, 2010. Special issue devoted to the 15th ILAS Conference at Cancun, Mexico, June 16-20, 2008.

[15] David Hu and Alec Sun. Probabilistic zero forcing on grid, regular, and hypercube graphs. *arXiv e-prints*, page arXiv:2010.12343, October 2020.

[16] Liang-Hao Huang, Gerard J. Chang, and Hong-Gwa Yeh. On minimum rank and zero forcing sets of a graph. *Linear Algebra and its Applications*, 432(11):2961–2973, 2010.

[17] Cong X Kang and Eunjeong Yi. Probabilistic zero forcing in graphs. *Bull. Inst. Combin. Appl.*, 67:9–16, 2013.

[18] Franklin H.J. Kenter and Jephian C.-H. Lin. A zero forcing technique for bounding sums of eigenvalue multiplicities. *Linear Algebra and its Applications*, 629:138–167, 2021.

[19] Shyam Narayanan and Alec Sun. Bounds on expected propagation time of probabilistic zero forcing. *European Journal of Combinatorics*, 98:103405, 2021.

[20] P.M. Nylen. Minimum-rank matrices with prescribed graph. *Linear Algebra and its Applications*, 248:303–316, 1996.

[21] Fatemeh Alinaghipour Taklimi. *Zero Forcing Sets for Graphs.* PhD thesis, University of Regina, Regina, Saskatchewan, 2013.

Appendix A

Twisted Hypercubes with Small Zero Forcing Number

Construct the following twisted hypercubes according to the specified permutations:

Let $\hat{Q}_3$ be the 3-dimensional hypercube, Q_3. Denote its adjacency matrix T_3.

Let P_8 be the 8×8 permutation matrix with first and second rows swapped:

$$P_8 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Let P_{16} be the 16×16 permutation matrix with second and third rows swapped, and let P_{32} be the 32×32 permutation matrix with third and fourth rows swapped.

Then T_4, generated by T_3 with permutation P_8,

$$T_4 = \begin{bmatrix} T_3 & P_8 \\ P_8^T & T_3 \end{bmatrix}$$

is the 4-dimensional twisted hypercube from Figure 3, which was shown to have zero forcing number 7.

Generate T_5 with T_4 and permutation P_{16},

$$T_5 = \begin{bmatrix} T_4 & P_{16} \\ P_{16}^T & T_4 \end{bmatrix}$$

and T_6 with T_5 and permutation P_{32},

$$T_6 = \begin{bmatrix} T_5 & P_{32} \\ P_{32}^T & T_5 \end{bmatrix}$$

Then the 5-dimensional twisted hypercube corresponding to T_5 has zero forcing number 13, with a minimal zero forcing set being

$$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 12, 13, 15, 16\}$$

where the vertices are labeled according to the rows in T_5. The 6-dimensional twisted hypercube corresponding to T_5 has zero forcing number 25, with a minimal zero forcing set being

$$\{1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13, 14, 15, 16, 17, 18, 21, 22, 23, 24, 25, 29, 30, 31, 32\}$$

where the vertices are labeled according to the rows in T_6.

Appendix B

Algorithms

```python
import networkx as nx
import numpy as np
import itertools


def force(graph, vertex):
    """ Takes a nx.Graph() object with vertices coloured white or black and checks whether the given vertex can force
    one of its neighbours. If yes, that neighbour's colour is changed to black.
    :param graph: An nx.Graph() object whose vertices are all coloured either white or black
    :param vertex: The vertex to check whether it can force a neighbour
    :return: Returns a list, where the first entry is True if the force was successful, and False otherwise.
    The second entry is the forced vertex if [0] is True. If [0] is False, and [1] is not None, then the given vertex
    did not force because all neighbours are already black.
    """
    if graph.nodes[vertex]['colour'] == 'white':
        return [False, None]
    count = 0
    forced = None
    for nbr in graph.adj[vertex]:
        if graph.nodes[nbr]['colour'] == 'white':
            count += 1
            forced = nbr
    if count == 1:
        graph.nodes[forced]['colour'] = 'black'
        return [True, forced]
    elif count == 0:
```

```
        return [False, vertex]
    return [False, None]


def zero_force(graph):
    """ Carries out the colour change rule on the given graph until no more forces are possible.
        param graph: A nx.Graph() object with vertices coloured either white or
        black.
        :return: Returns True if successfully forced, False otherwise
        """
    forcing_list = []
    index = -1
    check = [True, None]
    for vertex in graph.nodes:
        if graph.nodes[vertex]['colour'] == 'black':
            forcing_list.append(vertex)
    black_vertices = list(forcing_list)
    if len(black_vertices) == len(graph.nodes):
        return True
    for vertex in forcing_list:
        if check[0] is False:
            if vertex == forcing_list[index + 1]:
                return False
        check = force(graph, vertex)
        if check[1] is None:
            forcing_list.append(vertex)
        elif check[0] is True:
            black_vertices.append(check[1])
            forcing_list.append(check[1])
            index = len(forcing_list) - 1 - forcing_list[::-1].index(vertex)
            if len(black_vertices) == len(graph.nodes):
                return True
        elif vertex == forcing_list[index + 1]:
            forcing_list.append(vertex)

```

```python

def zf_setup(graph, vertices):
    """Sets all nodes in graph to be white, then changes vertices to
    black"""
    nx.set_node_attributes(graph, 'white', 'colour')
    for vert in vertices:
        graph.nodes[vert]['colour'] = 'black'


def zf_num_top(graph):
    """Calculates the zero forcing number of graph by checking all
   possible subsets of vertices of a given size, beginning from the
   number of vertices"""
    i = len(graph.nodes)
    found = True
    while found is True:
        found = False
        i -= 1
        for vertices in itertools.combinations(graph.nodes, i):
            zf_setup(graph, vertices)
            if zero_force(graph):
                found = True
                break
    return i+1
```

Listing B.1: Algorithm to find the zero forcing number of a graph.

```python
import networkx as nx
import random
from zeroforcing2 import zf_setup


def pccr(graph, vertex, prob):
    """Returns the number of vertices forced"""
    count = 0
    for nbr in graph.adj[vertex]:
        if graph.nodes[nbr]['colour'] == 'white':
            rand = random.random()
            if rand < prob:
                graph.nodes[nbr]['colour'] = 'black'
```

```
                count += 1
    return count


def find_probabilities(graph):
    probabilities = {}
    for vertex in graph.nodes:
        if graph.nodes[vertex]['colour'] == 'white':
            continue
        denominator = len(graph.adj[vertex])
        numerator = 1
        for nbr in graph.adj[vertex]:
            if graph.nodes[nbr]['colour'] == 'black':
                numerator += 1
        prob = numerator / denominator
        if prob <= 1:
            probabilities[vertex] = prob
    return probabilities


def propagate_sequence(graph):
    num_white_verts = 0
    for vertex in graph.nodes:
        if graph.nodes[vertex]['colour'] == 'white':
            num_white_verts += 1
    turns = 0
    while num_white_verts > 0:
        probs = find_probabilities_sequence(graph)
        for vertex in probs:
            num_white_verts -= pccr(graph, vertex, probs[vertex])
        turns += 1
    return turns


def simulate(graph, init_vertex, num):
    results = {}
    for i in range(num):
        zf_setup(graph, init_vertex)
        result = propagate(graph)
```

```
        if result in results:
            results[result] += 1
        else:
            results[result] = 1
    count = 0
    for value in results:
        count += value * results[value]
    print('Average:', count/num)
    return results
```

Listing B.2: Probabilistic zero forcing algorithm.