Throttling for standard zero forcing on digraphs

Emily Cairncross [1] Joshua Carlson [2] Peter Hollander [2]
Benjamin Kitchen [2] **Emily Lopez** [3] **Ashley Zhuang** [4]

SMALL 2020 Research Program
Williams College
Williamstown, MA

`Nebraska Conference for Undergraduate Women in Mathematics`

23 January 2021

[1]Oberlin College
[2]Williams College
[3]University of California - Santa Barbara
[4]Harvard University

Definition

A **digraph** is an ordered pair $\Gamma = (V(\Gamma); E(\Gamma))$ where $V(\Gamma)$ is a set of vertices and $E(\Gamma)$ is a multi-set of directed edges, or arcs. An **arc** is an ordered pair of vertices. A digraph can have loops, parallel arcs, and doubly-directed arcs.

Definition

A **digraph** is an ordered pair $\Gamma = (V(\Gamma); E(\Gamma))$ where $V(\Gamma)$ is a set of vertices and $E(\Gamma)$ is a multi-set of directed edges, or arcs. An **arc** is an ordered pair of vertices. A digraph can have loops, parallel arcs, and doubly-directed arcs.

Definition

A **simple digraph** is a digraph in which there are no parallel arcs or loops.

Definition

A **digraph** is an ordered pair $\Gamma = (V(\Gamma); E(\Gamma))$ where $V(\Gamma)$ is a set of vertices and $E(\Gamma)$ is a multi-set of directed edges, or arcs. An **arc** is an ordered pair of vertices. A digraph can have loops, parallel arcs, and doubly-directed arcs.

Definition

A **simple digraph** is a digraph in which there are no parallel arcs or loops.

Definition

An **oriented graph** is a simple digraph with no doubly-directed arcs.

(a) digraph (b) simple digraph (c) oriented graph

(a) digraph (b) simple digraph (c) oriented graph

Definition

Let Γ be a digraph with $u, v \in V(\Gamma)$ and $(u, v) \in E(\Gamma)$. Then v is an **out-neighbor** of u and u is an **in-neighbor** of v.

Standard Zero Forcing

Let the digraph Γ have an initial coloring of its vertices as either blue or white. The **zero forcing process** is a repeated application of the following color change rule: If a blue vertex u has exactly one white out-neighbor v, then u will force v to turn blue.

Standard Zero Forcing

Let the digraph Γ have an initial coloring of its vertices as either blue or white. The **zero forcing process** is a repeated application of the following color change rule: If a blue vertex u has exactly one white out-neighbor v, then u will force v to turn blue.

Definition

Begin with $B \quad V(\Gamma)$ colored blue and $V(\Gamma)\, n\, B$ colored white. If it is possible to apply the color change rule repeatedly to color the whole digraph, B is a **zero forcing set**.

De nition

The time steps of $B \quad V(\ )$ are de ned as follows.

- De ne $B^{(0)} = B$:
- For each $t \quad 0$; color $\bigcup_{i=0}^{t} B^{(i)}$ blue and $V(\ ) \ n \bigcup_{i=0}^{t} B^{(i)}$ white.
- $B^{(t+1)}$ is the set of all vertices that can be forced simultaneously using this coloring.

De nition

The time steps of B V( ) are de ned as follows.

- De ne $B^{(0)} = B$:
- For each t 0; color $\bigcup_{i=0}^{t} B^{(i)}$ blue and $V( )$ n $\bigcup_{i=0}^{t} B^{(i)}$ white.
- $B^{(t+1)}$ is the set of all vertices that can be forced simultaneously using this coloring.

De nition

For B V( ) ; the propagation time of B , denoted pt(; B), is the smallest t such that $\bigcup_{i=0}^{t} B^{(i)} = V( )$:

Definition [Butler, Young, 13]

The throttling number of some simple digraph , denoted th( ), is the minimum value of $|B| + \text{pt}(; B)$ where B ranges over all zero forcing sets of .

For example, the throttling number of

is $2 + 2 = 4$.

Theorem [Carlson, 19]

Given a simple graph G and a positive integer t, $\text{th}(G) \le t$ if and only if there exist integers $a \ge 1$ and $b \ge 0$ such that $a + b = t$ and G can be obtained from $K_a \,\square\, P_{b+1}$ by contracting path edges and deleting complete edges.

Definition

To construct the graph $H_{a,b+1}$, we

- arrange the vertices in an $a \times (b+1)$ lattice,
- make each row a directed path to the right,
- make each column a doubly-directed complete graph,
- and insert all other possible arcs oriented going to the left.

Make each row a directed path to the right

Add complete arcs within the columns

Add backwards arcs

Definition

We define path arcs of $H_{a;b+1}$ to be the arcs going to the right along the rows of the lattice to the right.

Then, we call all other arcs non-path arcs.

Definition

To contract arcs, we do the following:

Before

After

Theorem [CCHKLZ20+]

Given a simple digraph and a positive integer t, th( ) t if and only if there exist integers a 1 and b 0 such that $a + b = t$ and can be obtained from $H_{a;b+1}$ by contracting path arcs and deleting non-path arcs.

Let G be a simple graph.

Definition

An orientation G of G is an oriented graph such that ignoring the directions of the arcs gives the graph G.

Let G be a simple graph.

Definition

An orientation $\vec{G}$ of G is an oriented graph such that ignoring the directions of the arcs gives the graph G.

Definition

The orientation throttling interval of G is $OTI(G) = [m, M]$ where $m = \min_{\vec{G}} th(\vec{G})$ and $M = \max_{\vec{G}} th(\vec{G})$.

Let G be a simple graph.

Definition

An orientation $\vec{G}$ of G is an oriented graph such that ignoring the directions of the arcs gives the graph G.

Definition

The orientation throttling interval of G is $OTI(G) = [m, M]$ where $m = \min_{\vec{G}} th(\vec{G})$ and $M = \max_{\vec{G}} th(\vec{G})$.

Definition

G has a full orientation throttling interval if for every k such that $m \leq k \leq M$, there is some orientation $\vec{G}$ such that $th(\vec{G}) = k$.

De nition

Given a digraph , when we replace an arc $(a; b)$ 2 $E( )$ with $(b; a)$, we call this process ipping the arc $(a; b)$.

De nition

Given a digraph , when we replace an arc $(a; b)$ 2 $E( )$ with $(b; a)$, we call this process ipping the arc $(a; b)$.

Lemma [CCHKLZ20+]

If a simple digraph $_0$ is obtained from another simple digraph by ipping a single arc $(a; b)$ 2 $E( )$ where $(b; a)$ 2 $E( )$, then $j\, th(_0) \quad th( )\, j \quad 1$.

We will start by proving that throttling number will not increase by more than 1.

Graph with throttling number 6

Graph with throttling number 6

Graph with throttling number 6

Graph with throttling number 6

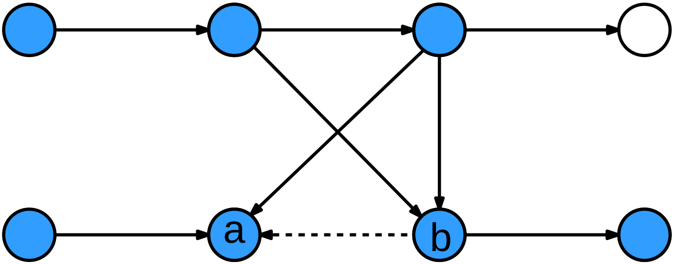
Graph with throttling number 6

Flip a forcing arc

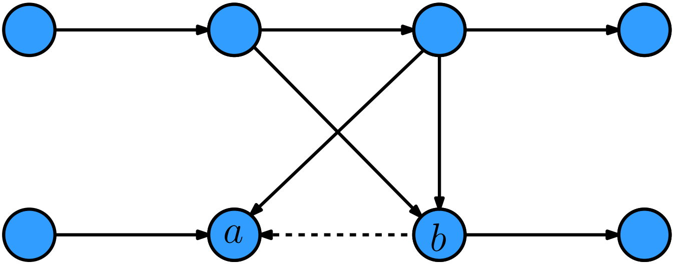
Flip a forcing arc

Add b to the initial forcing set

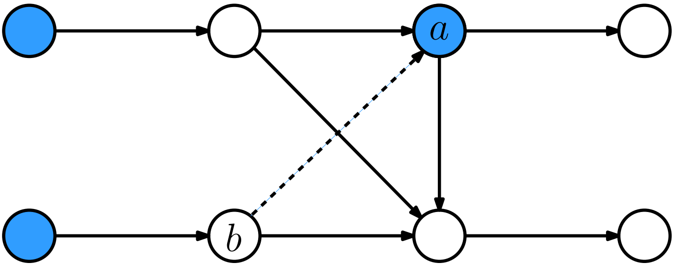
Vertex b can't be forced by a anymore, but b is already blue.



Vertex b is the only vertex that gained an outneighbor, namely a, but a will be blue before the time step that b forced originally.



Propagation time does not increase



A similar argument can be made for the case of flipping a non-forcing arc.

Throttling number also can't decrease by more than 1 because if it did, you could flip the same arc again and have a contradiction.

Theorem [CCHKLZ20+]

The orientation throttling interval of any simple graph is full.

- Start with the orientation that achieves the minimum throttling number.
- Flip a series of arcs to attain the orientation with the maximum throttling number.
- By previous result, every throttling number between the minimum and maximum has to be achieved.

Proposition [CCHKLZ20+]

The maximum possible OTI for a graph G with n vertices is $[\lceil 2\sqrt{n} - 1 \rceil; n]$.

- $\lceil 2\sqrt{n} - 1 \rceil$ is the lower bound for throttling on an undirected graph which relies on the fact that every vertex can only ever force once. This is still true for throttling on a directed graph, so we get the same lower bound.
- Every graph on n vertices can achieve throttling number at most n by coloring all of the vertices blue and forcing in zero time steps.

Theorem [CCHKLZ20+]

The orientation throttling interval (OTI) of a complete graph is maximum.

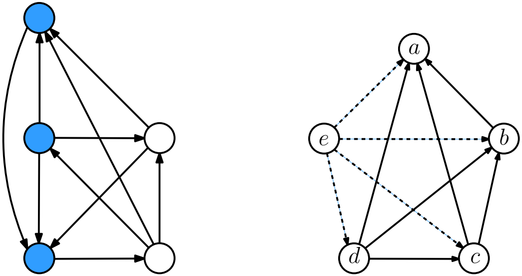


Figure 3: How to construct minimum and maximum orientations for $n = 5$.

Questions yet to be answered:

Is it always possible to achieve the throttling number of a simple graph with one of its orientations?

What is the orientation throttling interval of a path?

Does maximizing the number of vertices with only out-neighbors or only in-neighbors maximize the throttling number as well?

How do other graph operations influence throttling?