

# Dynamical Algebraic Combinatorics, Asynchronous Cellular Automata, and Toggling Independent Sets

Automata 2021

**Alex McDonough**

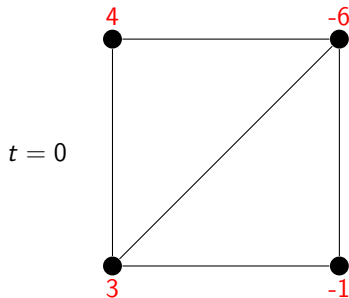
Joint work with Laurent David (UT Dallas), Colin Defant (Princeton), Mike Joseph (Dalton State), and Matthew Macauley (Clemson)

Department of Mathematics  
Brown University  $\rightarrow$  University of California, Davis

13 July, 2021

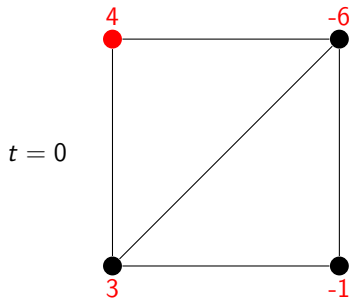
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



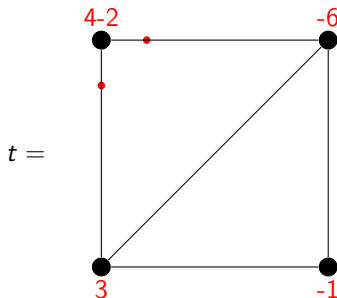
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



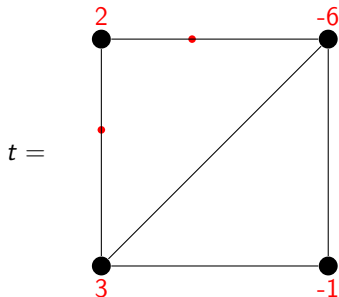
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



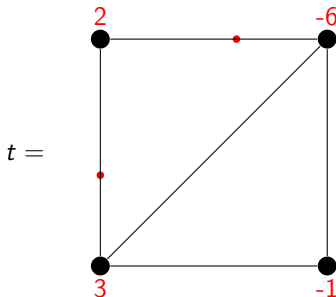
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



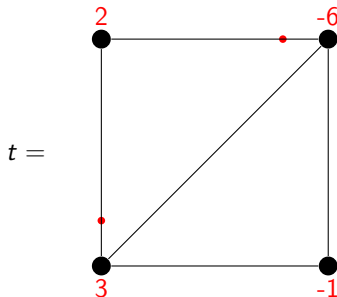
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



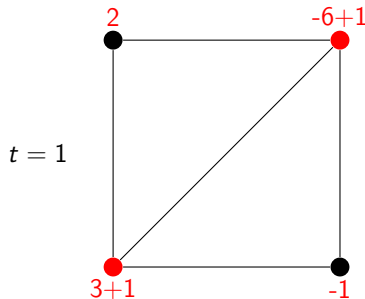
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



# What is Dynamical Algebraic Combinatorics?

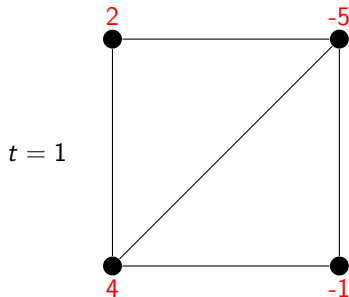
- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.





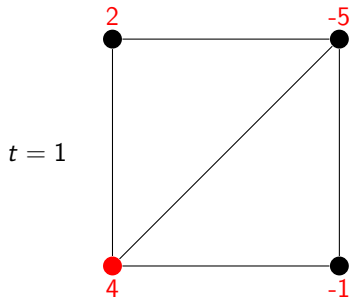
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



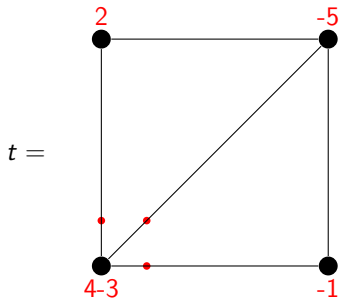
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



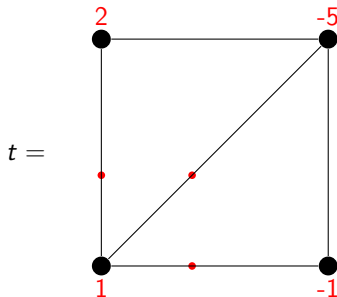
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



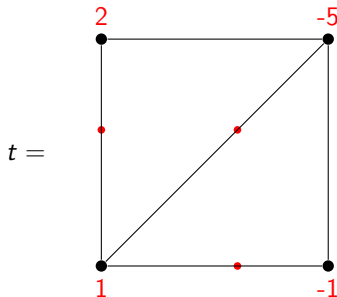
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



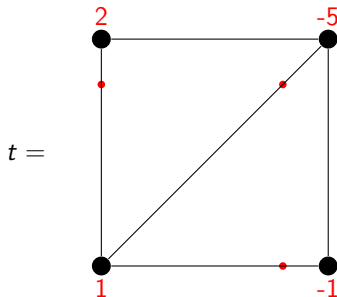
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



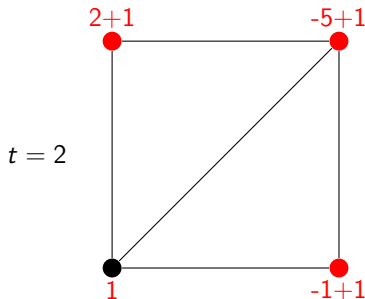
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



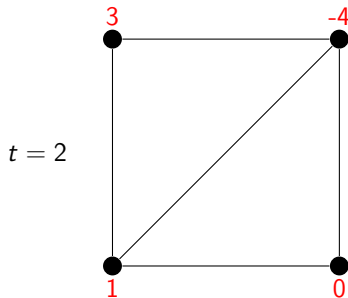
# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



# What is Dynamical Algebraic Combinatorics?

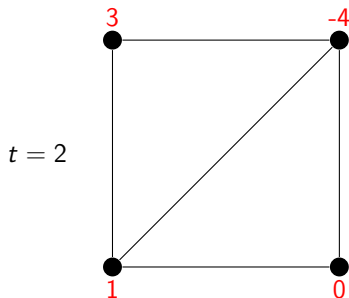
- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.





# What is Dynamical Algebraic Combinatorics?

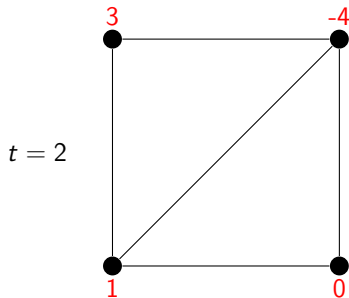
- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



- If the object and action come from algebraic combinatorics, then the study of this system is called **dynamical algebraic combinatorics** (DAC).

# What is Dynamical Algebraic Combinatorics?

- In general, to have a **discrete dynamical system** we need an *object* and an *action* which changes the object for every discrete time step.



- If the object and action come from algebraic combinatorics, then the study of this system is called **dynamical algebraic combinatorics** (DAC).
- Some popular DAC problems can be cast and analyzed as **automata networks**.

# Dynamical Algebraic Combinatorics & Cellular Automata

- DAC and CA theory are fairly separate fields; they were developed to answer different kinds of questions, and there is not a lot of overlap of researchers.

# Dynamical Algebraic Combinatorics & Cellular Automata

- DAC and CA theory are fairly separate fields; they were developed to answer different kinds of questions, and there is not a lot of overlap of researchers.
- However, despite evolving independently, there is a lot of overlap in the structures that are studied.

# Dynamical Algebraic Combinatorics & Cellular Automata

- DAC and CA theory are fairly separate fields; they were developed to answer different kinds of questions, and there is not a lot of overlap of researchers.
- However, despite evolving independently, there is a lot of overlap in the structures that are studied.



Images from Wikipedia

## Definition

A **partially-ordered set** or **poset** is a set  $P$  together with a relation  $\leq$  that is:

- reflexive:  $x \leq x$  for all  $x \in P$ .
- antisymmetric: If  $x \leq y$  and  $y \leq x$ , then  $x = y$ .
- transitive: If  $x \leq y$  and  $y \leq z$ , then  $x \leq z$ .

## Definition

A **partially-ordered set** or **poset** is a set  $P$  together with a relation  $\leq$  that is:

- reflexive:  $x \leq x$  for all  $x \in P$
- antisymmetric: If  $x \leq y$  and  $y \leq x$ , then  $x = y$ .
- transitive: If  $x \leq y$  and  $y \leq z$ , then  $x \leq z$ .

## Example

- Consider ordered pairs where  $(a, b) \leq (c, d)$  if  $a \leq c$  and  $b \leq d$ .

$$(3, 2) \leq (3, 7)$$

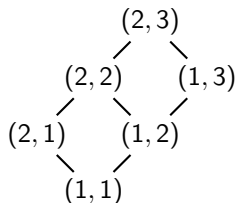
$$(4, 3) \leq (7, 8)$$

$(3, 4)$  and  $(7, 2)$  cannot be compared.

## Definition

Each poset has an associated graph called a **Hasse diagram**.

- The elements of  $P$  form the vertices of the diagram.
- If  $x \leq y$ , we write  $y$  higher up than  $x$ .
- If  $x < y$  and there is no  $z$  such that  $x < z < y$ , we connect  $x$  and  $y$  with an edge and say  $y$  **covers**  $x$  (written  $x \lessdot y$ ).



The poset  $P = [2] \times [3]$ .

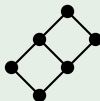


# More Examples of Hasse Diagrams

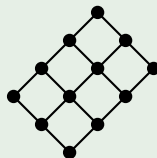
## Example (Products of two chains)



$$[2] \times [2]$$



$$[2] \times [3]$$

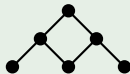


$$[3] \times [4]$$

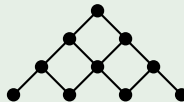
## Example (Type A positive root posets)



$$\Phi^+(A_2)$$



$$\Phi^+(A_3)$$



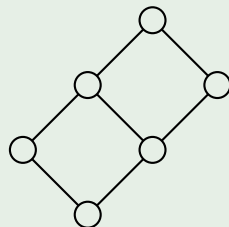
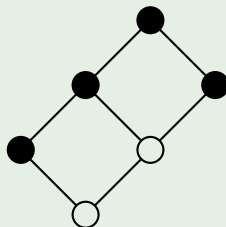
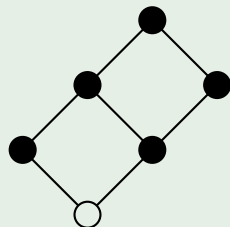
$$\Phi^+(A_4)$$

# Up-sets of Posets

## Definition

- An **up-set** (or **order filter**) of a poset  $P$  is a subset  $U \subseteq P$  such that if  $x \in U$  and  $x \leq y$ , then  $y \in U$ .

## Example



some up-sets of  $[2] \times [3]$

# Rowmotion

- Let  $P$  be a poset and  $\mathcal{U}(P)$  be the set of up-sets.
- We define a group action on  $\mathcal{U}(P)$  called **rowmotion**.

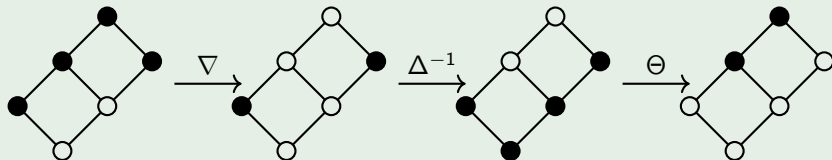
# Rowmotion

- Let  $P$  be a poset and  $\mathcal{U}(P)$  be the set of up-sets.
- We define a group action on  $\mathcal{U}(P)$  called **rowmotion**.
- Given  $U \in \mathcal{U}(P)$ , perform the following 3 steps:
  - 1  $\nabla$ : Take the minimal elements.
  - 2  $\Delta^{-1}$ : Saturate downward.
  - 3  $\Theta$ : Take the complement.
- We call the result  $\text{Row}(U)$

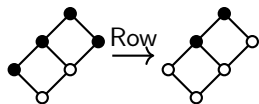
# Rowmotion

- Let  $P$  be a poset and  $\mathcal{U}(P)$  be the set of up-sets.
- We define a group action on  $\mathcal{U}(P)$  called **rowmotion**.
- Given  $U \in \mathcal{U}(P)$ , perform the following 3 steps:
  - 1  $\nabla$ : Take the minimal elements.
  - 2  $\Delta^{-1}$ : Saturate downward.
  - 3  $\Theta$ : Take the complement.
- We call the result  $\text{Row}(U)$

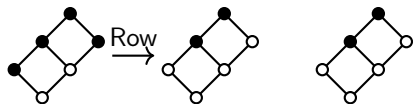
## Example



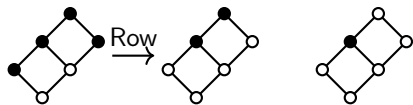
# Orbits of rowmotion on up-sets of $[2] \times [3]$



# Orbits of rowmotion on up-sets of $[2] \times [3]$

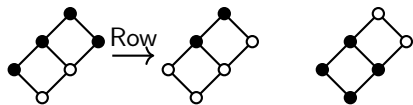


# Orbits of rowmotion on up-sets of $[2] \times [3]$

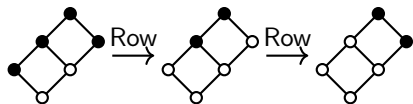




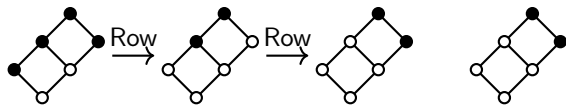
# Orbits of rowmotion on up-sets of $[2] \times [3]$



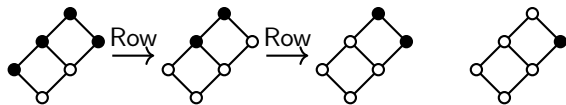
# Orbits of rowmotion on up-sets of $[2] \times [3]$



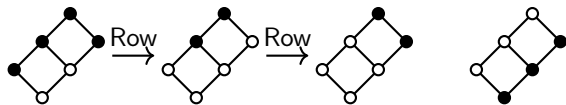
# Orbits of rowmotion on up-sets of $[2] \times [3]$



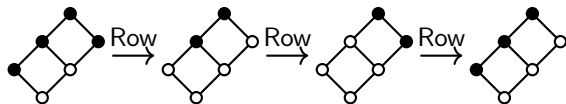
# Orbits of rowmotion on up-sets of $[2] \times [3]$



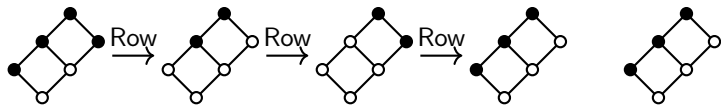
# Orbits of rowmotion on up-sets of $[2] \times [3]$



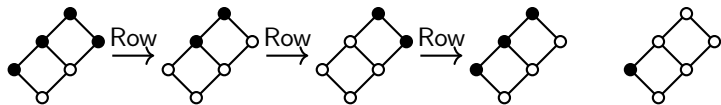
# Orbits of rowmotion on up-sets of $[2] \times [3]$



# Orbits of rowmotion on up-sets of $[2] \times [3]$

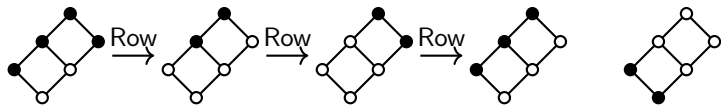


# Orbits of rowmotion on up-sets of $[2] \times [3]$

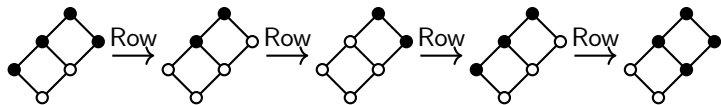




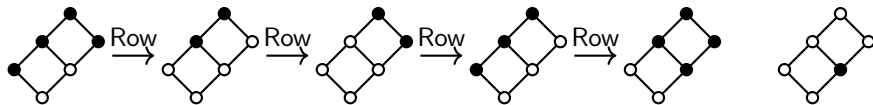
# Orbits of rowmotion on up-sets of $[2] \times [3]$



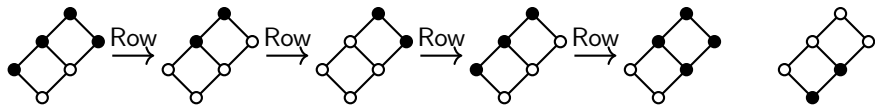
# Orbits of rowmotion on up-sets of $[2] \times [3]$



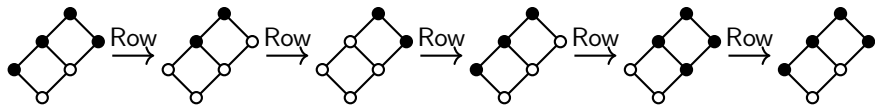
# Orbits of rowmotion on up-sets of $[2] \times [3]$



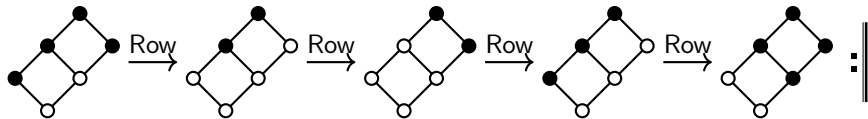
# Orbits of rowmotion on up-sets of $[2] \times [3]$



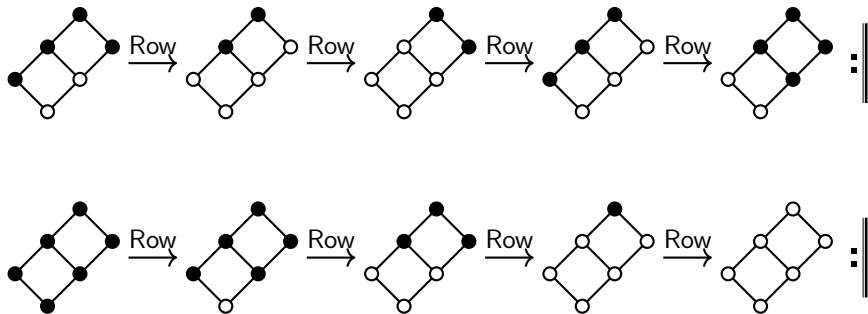
# Orbits of rowmotion on up-sets of $[2] \times [3]$



# Orbits of rowmotion on up-sets of $[2] \times [3]$

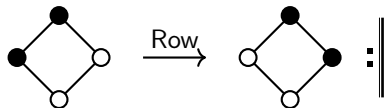
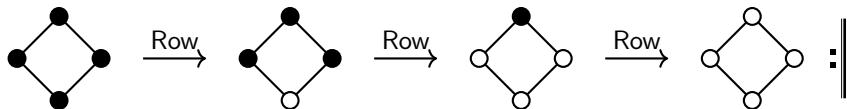


# Orbits of rowmotion on up-sets of $[2] \times [3]$



On  $[2] \times [3]$ , up-set rowmotion is periodic with period 5.

# Orbits of rowmotion on up-sets of $[2] \times [2]$



On  $[2] \times [2]$ , up-set rowmotion is periodic with period 4.

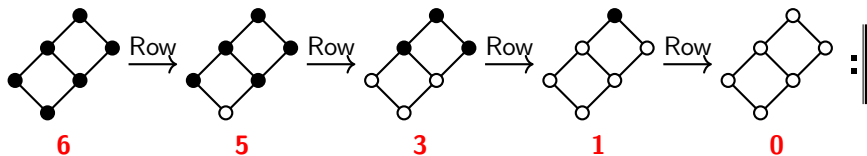
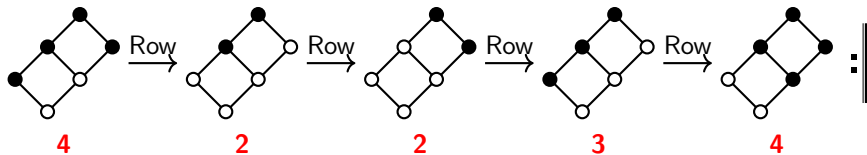


# Properties of Rowmotion on $[a] \times [b]$

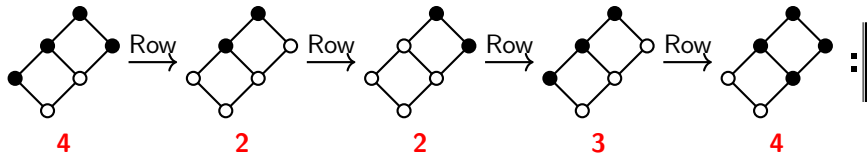
## Theorem (Brouwer–Schrijver 1974)

On  $[a] \times [b]$ , rowmotion is periodic with period  $a + b$ .

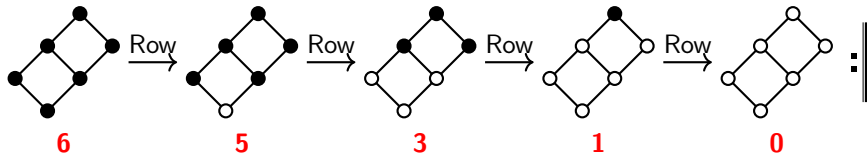
# Cardinalities in $[2] \times [3]$



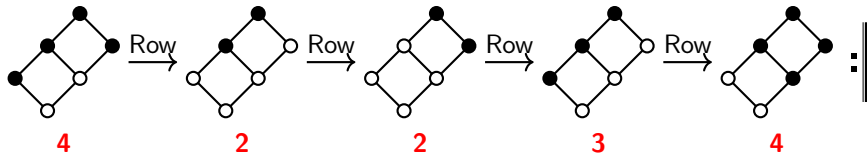
# Cardinalities in $[2] \times [3]$



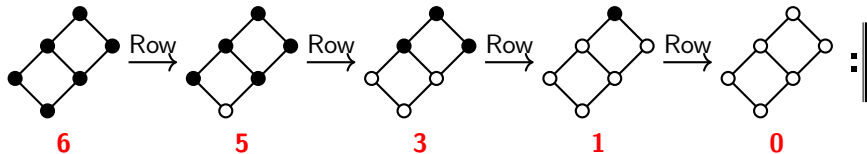
**Average cardinality: 3**



# Cardinalities in $[2] \times [3]$

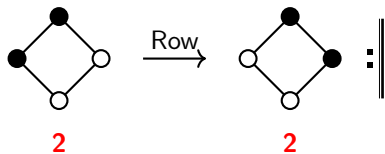
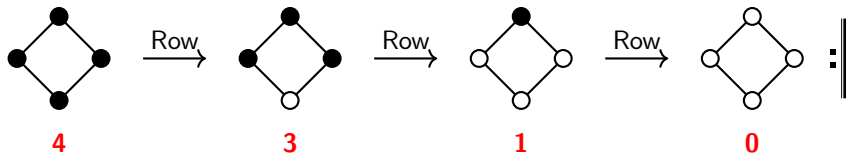


**Average cardinality: 3**

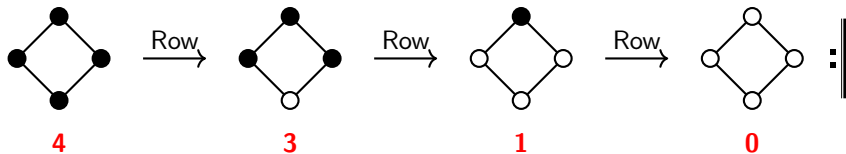


**Average cardinality: 3**

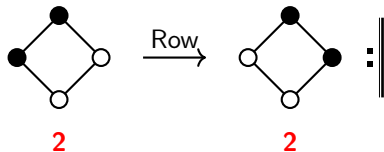
# Cardinalities in $[2] \times [2]$



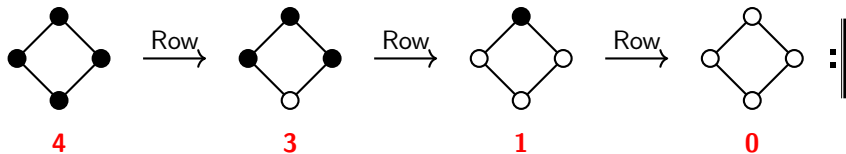
# Cardinalities in $[2] \times [2]$



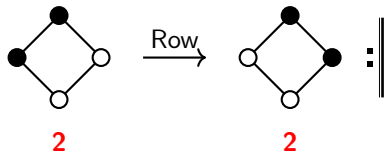
**Average cardinality: 2**



# Cardinalities in $[2] \times [2]$



**Average cardinality: 2**



**Average cardinality: 2**

# Properties of Rowmotion on $[a] \times [b]$

## Theorem (Brouwer–Schrijver 1974)

On  $[a] \times [b]$ , rowmotion is periodic with period  $a + b$ .

## Theorem (Propp–Roby 2013)

On  $[a] \times [b]$ , the average cardinality of up-sets across any rowmotion orbit is  $ab/2$ .



# Properties of Rowmotion on $[a] \times [b]$

## Theorem (Brouwer–Schrijver 1974)

On  $[a] \times [b]$ , rowmotion is periodic with period  $a + b$ .

## Theorem (Propp–Roby 2013)

On  $[a] \times [b]$ , the average cardinality of up-sets across any rowmotion orbit is  $ab/2$ . We say that the cardinality is **homomesic** (or  **$ab/2$ -mesic**) under the action of rowmotion.

# Properties of Rowmotion on $[a] \times [b]$

## Theorem (Brouwer–Schrijver 1974)

On  $[a] \times [b]$ , rowmotion is periodic with period  $a + b$ .

## Theorem (Propp–Roby 2013)

On  $[a] \times [b]$ , the average cardinality of up-sets across any rowmotion orbit is  $ab/2$ . We say that the cardinality is **homomesic** under the action of rowmotion with average  $ab/2$ .

- In general a discrete dynamical system exhibits **homomesy** when the average of some statistic is constant over all orbits.

# Properties of Rowmotion on $[a] \times [b]$

## Theorem (Brouwer–Schrijver 1974)

On  $[a] \times [b]$ , rowmotion is periodic with period  $a + b$ .

## Theorem (Propp–Roby 2013)

On  $[a] \times [b]$ , the average cardinality of up-sets across any rowmotion orbit is  $ab/2$ . We say that the cardinality is **homomesic** under the action of rowmotion with average  $ab/2$ .

- In general a discrete dynamical system exhibits **homomesy** when the average of some statistic is constant over all orbits.
- The homomesy phenomenon was first observed by Panyushev in 2007 and a shocking number of examples have been found since then.

## Definition (Cameron and Fon-Der-Flaass 1995)

Let  $\mathcal{U}(P)$  be the set of up-sets of a finite poset  $P$ .

Let  $e \in P$ . Then the **toggle** corresponding to  $e$  is the map  $T_e : \mathcal{U}(P) \rightarrow \mathcal{U}(P)$  defined by

$$T_e(U) = \begin{cases} U \cup \{e\} & \text{if } e \notin U \text{ and } U \cup \{e\} \in \mathcal{U}(P), \\ U \setminus \{e\} & \text{if } e \in U \text{ and } U \setminus \{e\} \in \mathcal{U}(P), \\ U & \text{otherwise.} \end{cases}$$

## Definition (Cameron and Fon-Der-Flaass 1995)

Let  $\mathcal{U}(P)$  be the set of up-sets of a finite poset  $P$ .

Let  $e \in P$ . Then the **toggle** corresponding to  $e$  is the map  $T_e : \mathcal{U}(P) \rightarrow \mathcal{U}(P)$  defined by

$$T_e(U) = \begin{cases} U \cup \{e\} & \text{if } e \notin U \text{ and } U \cup \{e\} \in \mathcal{U}(P), \\ U \setminus \{e\} & \text{if } e \in U \text{ and } U \setminus \{e\} \in \mathcal{U}(P), \\ U & \text{otherwise.} \end{cases}$$

## Definition

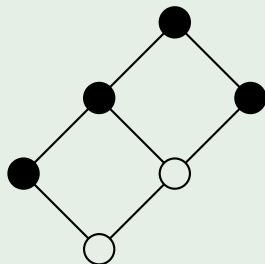
These toggles generate a group called the **toggle group** (also called the **dynamics group** by CA theorists).

# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

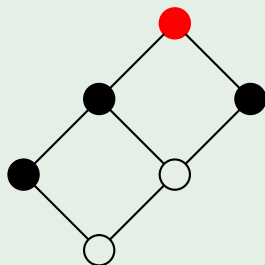


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

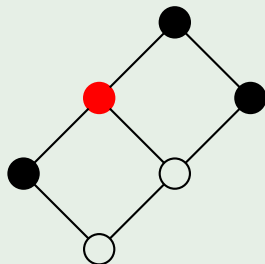


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example



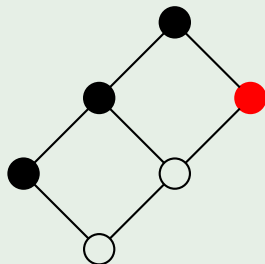


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

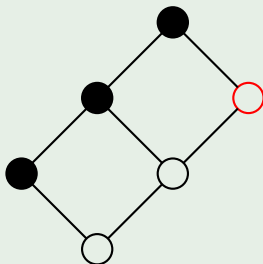


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

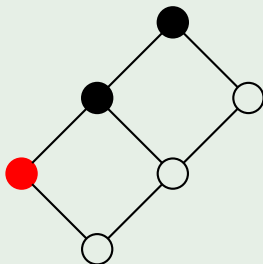


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

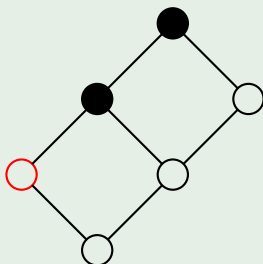


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

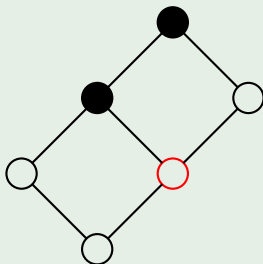


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

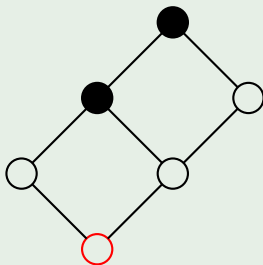


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example

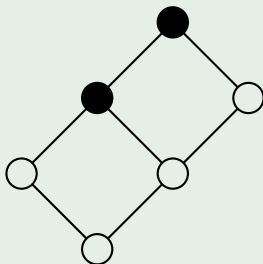


# Toggles and Rowmotion

## Theorem (Cameron and Fon-Der-Flaass 1995)

Applying the toggles  $T_e$  from top to bottom across the rows of  $P$  gives rowmotion on up-sets of  $P$ .

## Example



## Theorem (Striker–Williams 2012)

Applying the toggles  $T_e$  from left to right down the columns of  $P$  gives another known group action called **promotion**.



# Rowmotion and Promotion

## Theorem (Striker–Williams 2012)

Applying the toggles  $T_e$  from left to right down the columns of  $P$  gives another known group action called **promotion**.

## Corollary

There is an **equivariant bijection** between up-sets under the rowmotion group action and up-sets under the promotion group action.

# Rowmotion and Promotion

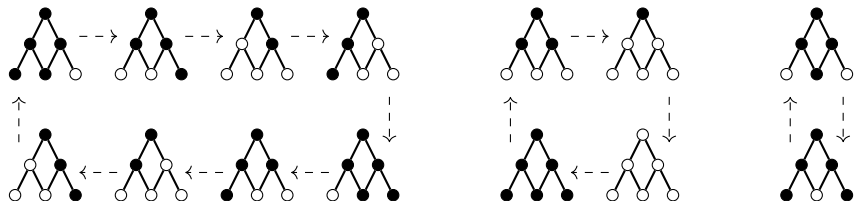


Figure: Rowmotion – toggle “by rows, top-to-bottom”.

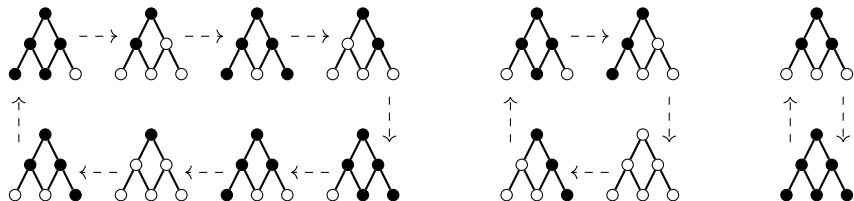


Figure: Promotion – toggle “by columns, left-to-right”.

# Rowmotion and Promotion

## Theorem (Striker–Williams 2012)

Applying the toggles  $T_e$  from left to right down the columns of  $P$  gives another known group action called **promotion**.

## Corollary

There is an **equivariant bijection** between up-sets under the rowmotion group action and promotion group action.

- The corollary follows from *Coxeter theory* (the two actions are *torically equivalent*).

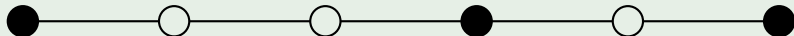
# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.

# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.

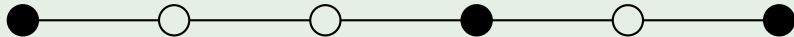
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

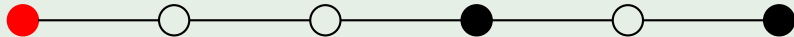
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

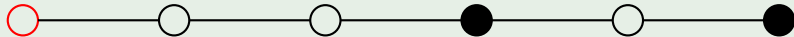
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

## Example

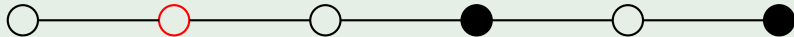




# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

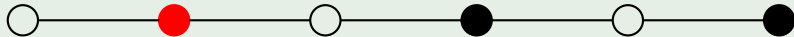
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

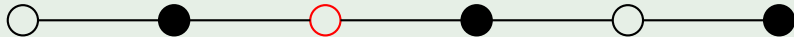
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

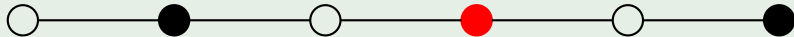
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

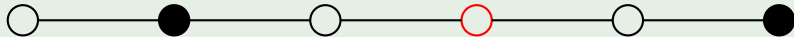
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

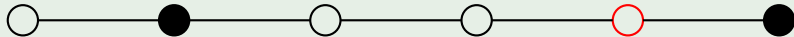
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

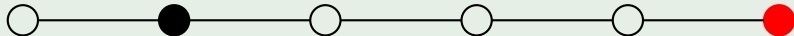
## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

## Example



# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

## Example

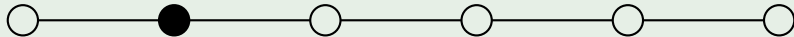




# Toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

## Example



# toggling Independent Sets on a Path

- Rowmotion and promotion aren't the only dynamical system that can be described by toggling.
- Consider an **independent set** (set of nonadjacent vertices) on a path graph.
- From left to right, we **toggle** each vertex, switching its status if we still have an independent set.

## Example



- We end up with a new independent set and can repeat the process indefinitely.

# Toggling Independent Sets on a Path

- The dynamics of this kind of toggling are more easily seen on an array.

$x$	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$
$x^{(0)}$	<b>1</b>	0	0	<b>1</b>	0	<b>1</b>
$x^{(1)}$	0	<b>1</b>	0	0	0	0
$x^{(2)}$	0	0	<b>1</b>	0	<b>1</b>	0
$x^{(3)}$	<b>1</b>	0	0	0	0	<b>1</b>
$x^{(4)}$	0	<b>1</b>	0	<b>1</b>	0	0
$x^{(5)}$	0	0	0	0	<b>1</b>	0
$x^{(6)}$	<b>1</b>	0	<b>1</b>	0	0	<b>1</b>
$x^{(7)}$	0	0	0	<b>1</b>	0	0
$x^{(8)}$	<b>1</b>	0	0	0	<b>1</b>	0
$x^{(9)}$	0	<b>1</b>	0	0	0	<b>1</b>
$x^{(10)}$	0	0	<b>1</b>	0	0	0

- Here, each row is a independent set of entries marked with **1s**.

# Toggling Independent Sets on a Path

- The dynamics of this kind of toggling are more easily seen on an array.

$x$	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$
$x^{(0)}$	<b>1</b>	0	0	<b>1</b>	0	<b>1</b>
$x^{(1)}$	0	<b>1</b>	0	0	0	0
$x^{(2)}$	0	0	<b>1</b>	0	<b>1</b>	0
$x^{(3)}$	<b>1</b>	0	0	0	0	<b>1</b>
$x^{(4)}$	0	<b>1</b>	0	<b>1</b>	0	0
$x^{(5)}$	0	0	0	0	<b>1</b>	0
$x^{(6)}$	<b>1</b>	0	<b>1</b>	0	0	<b>1</b>
$x^{(7)}$	0	0	0	<b>1</b>	0	0
$x^{(8)}$	<b>1</b>	0	0	0	<b>1</b>	0
$x^{(9)}$	0	<b>1</b>	0	0	0	<b>1</b>
$x^{(10)}$	0	0	<b>1</b>	0	0	0
Sum	<b>4</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>4</b>

- Here, each row is a independent set of entries marked with **1s**.

# Toggling Independent Sets on a Path

- The dynamics of this kind of toggling are more easily seen on an array.

$x$	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$
$x^{(0)}$	<b>1</b>	0	0	<b>1</b>	0	<b>1</b>
$x^{(1)}$	0	<b>1</b>	0	0	0	0
$x^{(2)}$	0	0	<b>1</b>	0	<b>1</b>	0
$x^{(3)}$	<b>1</b>	0	0	0	0	<b>1</b>
$x^{(4)}$	0	<b>1</b>	0	<b>1</b>	0	0
$x^{(5)}$	0	0	0	0	<b>1</b>	0
$x^{(6)}$	<b>1</b>	0	<b>1</b>	0	0	<b>1</b>
$x^{(7)}$	0	0	0	<b>1</b>	0	0
$x^{(8)}$	<b>1</b>	0	0	0	<b>1</b>	0
$x^{(9)}$	0	<b>1</b>	0	0	0	<b>1</b>
$x^{(10)}$	0	0	<b>1</b>	0	0	0
Sum	<b>4</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>4</b>

**Theorem (Joseph–Roby 2018)**

The sum vector always reads the same left-to-right as right-to-left.

- Here, each row is a independent set of entries marked with **1s**.

# Toggleing Independent Sets on a Path

- The dynamics of this kind of toggling are more easily seen on an array.

$x$	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$
$x^{(0)}$	<b>1</b>	0	0	<b>1</b>	0	<b>1</b>
$x^{(1)}$	0	<b>1</b>	0	0	0	0
$x^{(2)}$	0	0	<b>1</b>	0	<b>1</b>	0
$x^{(3)}$	<b>1</b>	0	0	0	0	<b>1</b>
$x^{(4)}$	0	<b>1</b>	0	<b>1</b>	0	0
$x^{(5)}$	0	0	0	0	<b>1</b>	0
$x^{(6)}$	<b>1</b>	0	<b>1</b>	0	0	<b>1</b>
$x^{(7)}$	0	0	0	<b>1</b>	0	0
$x^{(8)}$	<b>1</b>	0	0	0	<b>1</b>	0
$x^{(9)}$	0	<b>1</b>	0	0	0	<b>1</b>
$x^{(10)}$	0	0	<b>1</b>	0	0	0
Sum	<b>4</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>4</b>

## Theorem (Joseph–Roby 2018)

The sum vector always reads the same left-to-right as right-to-left.

## Theorem (Joseph–Roby 2018)

If you double the first entry in the sum vector and add the second, this gives the number of rows.

- Here, each row is a independent set of entries marked with **1s**.

# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

# toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example

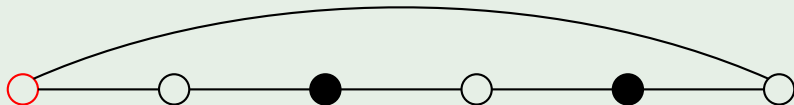




# toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

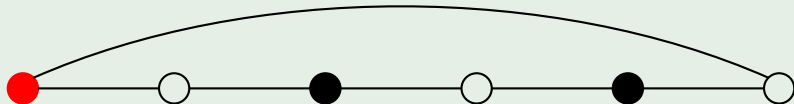
## Example



# toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

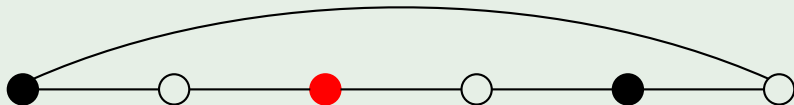
## Example



# Toggleing Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



# toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example

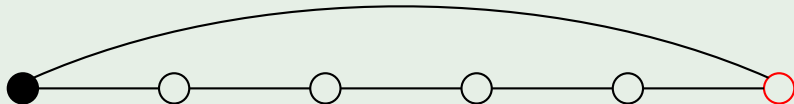




# Toggleing Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

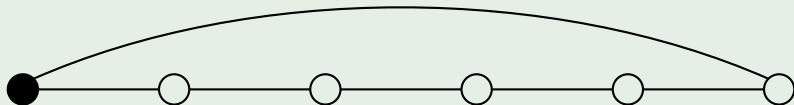
## Example



# Toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

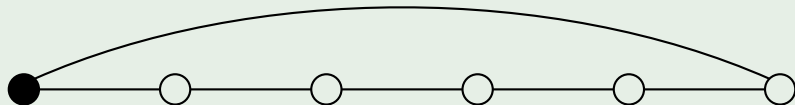
## Example



# toggling Independent Sets on a Cycle

- Instead of working on a path graph, we can also toggle vertices on a cycle.

## Example



- This dynamical system is similar to the one on the path graph, but we no longer have “special” vertices.

# toggling Independent Sets on a Cycle

- As before, we use an array for easier visualization.

$x$	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$	$v_6$
$x^{(0)}$	0	0	<b>1</b>	0	<b>1</b>	0
$x^{(1)}$	<b>1</b>	0	0	0	0	0
$x^{(2)}$	0	<b>1</b>	0	<b>1</b>	0	<b>1</b>
$x^{(3)}$	0	0	0	0	0	0
$x^{(4)}$	<b>1</b>	0	<b>1</b>	0	<b>1</b>	0
$x^{(5)}$	0	0	0	0	0	<b>1</b>
$x^{(6)}$	0	<b>1</b>	0	<b>1</b>	0	0
$x^{(7)}$	0	0	0	0	<b>1</b>	0
$x^{(8)}$	<b>1</b>	0	<b>1</b>	0	0	0
$x^{(9)}$	0	0	0	<b>1</b>	0	<b>1</b>
$x^{(10)}$	0	<b>1</b>	0	0	0	0

- This system is the main focus of our ongoing work, and will be the star of Matthew Macauley's talk tomorrow!

# Toggling and Asynchronous Cellular Automata

- Whenever we toggle vertex  $v_i$ , its new state depends on the states of  $v_{i-1}$ ,  $v_i$ , and  $v_{i+1}$ . Let  $x_i$  be the pre-toggle state of  $v_i$  and  $T_i(x_{i-1}, x_i, x_{i+1})$  be the post-toggle state.

# Toggling and Asynchronous Cellular Automata

- Whenever we toggle vertex  $v_i$ , its new state depends on the states of  $v_{i-1}$ ,  $v_i$ , and  $v_{i+1}$ . Let  $x_i$  be the pre-toggle state of  $v_i$  and  $T_i(x_{i-1}, x_i, x_{i+1})$  be the post-toggle state.

$(x_{i-1}, x_i, x_{i+1})$	111	110	101	100	011	010	001	000
$T_i(x_{i-1}, x_i, x_{i+1})$	$n/a$	$n/a$	0	0	$n/a$	0	0	1

# Toggling and Asynchronous Cellular Automata

- Whenever we toggle vertex  $v_i$ , its new state depends on the states of  $v_{i-1}$ ,  $v_i$ , and  $v_{i+1}$ . Let  $x_i$  be the pre-toggle state of  $v_i$  and  $T_i(x_{i-1}, x_i, x_{i+1})$  be the post-toggle state.

$(x_{i-1}, x_i, x_{i+1})$	111	110	101	100	011	010	001	000
$T_i(x_{i-1}, x_i, x_{i+1})$	$n/a$	$n/a$	0	0	$n/a$	0	0	1

- The  $2^8 = 256$  maps from  $\{0, 1\}^3 \rightarrow \{0, 1\}$  are called **elementary cellular automaton (ECA)** rules.

# Toggling and Asynchronous Cellular Automata

- Whenever we toggle vertex  $v_i$ , its new state depends on the states of  $v_{i-1}$ ,  $v_i$ , and  $v_{i+1}$ . Let  $x_i$  be the pre-toggle state of  $v_i$  and  $T_i(x_{i-1}, x_i, x_{i+1})$  be the post-toggle state.

$(x_{i-1}, x_i, x_{i+1})$	111	110	101	100	011	010	001	000
$T_i(x_{i-1}, x_i, x_{i+1})$	$n/a$	$n/a$	0	0	$n/a$	0	0	1

- The  $2^8 = 256$  maps from  $\{0, 1\}^3 \rightarrow \{0, 1\}$  are called **elementary cellular automaton (ECA)** rules.
- If we repeatedly toggle the vertices in some fixed order using an ECA rule, we get an **asynchronous cellular automata (ACA)**.



# Toggling and Asynchronous Cellular Automata

- Whenever we toggle vertex  $v_i$ , its new state depends on the states of  $v_{i-1}$ ,  $v_i$ , and  $v_{i+1}$ . Let  $x_i$  be the pre-toggle state of  $v_i$  and  $T_i(x_{i-1}, x_i, x_{i+1})$  be the post-toggle state.

$(x_{i-1}, x_i, x_{i+1})$	111	110	101	100	011	010	001	000
$T_i(x_{i-1}, x_i, x_{i+1})$	$n/a$	$n/a$	0	0	$n/a$	0	0	1

- The  $2^8 = 256$  maps from  $\{0, 1\}^3 \rightarrow \{0, 1\}$  are called **elementary cellular automaton (ECA)** rules.
- If we repeatedly toggle the vertices in some fixed order using an ECA rule, we get an **asynchronous cellular automata (ACA)**.
- An ACA is called **toggable** (or **order independent**) if each toggle is a bijection on the periodic points.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

- When an ACA is togglable, each “toggle” map is an involution or identity.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

- When an ACA is togglable, each “toggle” map is an involution or identity.
- Recall that these toggles generate a group called the **toggle group**.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

- When an ACA is togglable, each “toggle” map is an involution or identity.
- Recall that these toggles generate a group called the **toggle group**.
- In 2011, Macauley, McCammond, and Mortveit classified the toggle groups associated with these togglable ACAs.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

- When an ACA is togglable, each “toggle” map is an involution or identity.
- Recall that these toggles generate a group called the **toggle group**.
- In 2011, Macauley, McCammond, and Mortveit classified the toggle groups associated with these togglable ACAs.
- In 2015, Goles, Montalva-Medel, Mortveit, and Ramirez-Flandes generalized to *block update orders*.

## Theorem (Macauley–McCammond–Mortveit 2008)

104 of the 256 ECA rules produce togglable ACAs.

- When an ACA is togglable, each “toggle” map is an involution or identity.
- Recall that these toggles generate a group called the **toggle group**.
- In 2011, Macauley, McCammond, and Mortveit classified the toggle groups associated with these togglable ACAs.
- In 2015, Goles, Montalva-Medel, Mortveit, and Ramirez-Flandes generalized to *block update orders*.
- In 2018, Salo proved a conjecture about ECA rule 57 and Defant further explored ECA rules 150 and 105.

# Coxeter Groups

- Toggle groups are all quotients of **Coxeter groups**.



# Coxeter Groups

- Toggle groups are all quotients of **Coxeter groups**.
- Coxeter groups can be defined geometrically in terms of *reflections*, but I will focus on the combinatorial perspective.

# Coxeter Groups

- Toggle groups are all quotients of **Coxeter groups**.
- Coxeter groups can be defined geometrically in terms of *reflections*, but I will focus on the combinatorial perspective.

## Definition

A **Coxeter system** is a pair  $(W, S)$  where  $W$  is a group generated by the set  $S = \{s_1, \dots, s_n\}$  with the presentation

$$W = \langle s_1, \dots, s_n \mid s_i^2 = 1, (s_i s_j)^{m_{ij}} = 1 \text{ for } i \neq j \rangle,$$

where  $m_{ij} = |s_i s_j| \in \{2, 3, \dots\} \cup \{\infty\}$ .

# Coxeter Groups

- Toggle groups are all quotients of **Coxeter groups**.
- Coxeter groups can be defined geometrically in terms of *reflections*, but I will focus on the combinatorial perspective.

## Definition

A **Coxeter system** is a pair  $(W, S)$  where  $W$  is a group generated by the set  $S = \{s_1, \dots, s_n\}$  with the presentation

$$W = \langle s_1, \dots, s_n \mid s_i^2 = 1, (s_i s_j)^{m_{ij}} = 1 \text{ for } i \neq j \rangle,$$

where  $m_{ij} = |s_i s_j| \in \{2, 3, \dots\} \cup \{\infty\}$ .

- Notice that  $s_i^2 = 1$  implies that each  $s_i$  can be thought of as a toggle.

# Coxeter Diagrams

- We can encode any Coxeter system as a **Coxeter diagram**.

# Coxeter Diagrams

- We can encode any Coxeter system as a **Coxeter diagram**.

## Definition

A **Coxeter diagram** is a graph where edges may be weighted with positive integers or  $\infty$ .

- Each vertex corresponds to an involution  $s_i$ .
- For each  $i \neq j$ , let  $m_{ij} = \min_{k \geq 2} \{k \mid (s_i s_j)^k = 1\}$ .
- If  $m_{ij} \geq 3$ , connect  $s_i$  and  $s_j$  with an edge.
- If  $m_{ij} > 3$ , label this edge with  $m_{ij}$ .

# Coxeter Diagrams

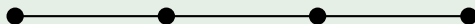
- We can encode any Coxeter system as a **Coxeter diagram**.

## Definition

A **Coxeter diagram** is a graph where edges may be weighted with positive integers or  $\infty$ .

- Each vertex corresponds to an involution  $s_i$ .
- For each  $i \neq j$ , let  $m_{ij} = \min_{k \geq 2} \{k \mid (s_i s_j)^k = 1\}$ .
- If  $m_{ij} \geq 3$ , connect  $s_i$  and  $s_j$  with an edge.
- If  $m_{ij} > 3$ , label this edge with  $m_{ij}$ .

## Example



**Figure:** This is the Coxeter Diagram for the group  $A_5$ .

Thanks for Listening! Merci de Votre Attention!



Image from Wikipedia

Be sure to attend Matthew Macauley's talk tomorrow at 15:30 for more cycle graph toggling!

# Sources I



A. Brouwer and L. Schrijver.

On the period of an operator, defined on antichains.

*Stichting Mathematisch Centrum. Zuivere Wiskunde, (ZW 24/74):1–13, 1974.*



P. Cameron and D. Fon-Der-Flaass.

Orbits of antichains revisited.

*European J. Combin.*, 16(6):545–554, 1995.



Matthew Cook.

Universality in elementary cellular automata.

*Complex Systems*, 15(1):1–40, 2004.



C. Defant.

Flexible toggles and symmetric invertible asynchronous elementary cellular automata.

*Discrete Math.*, 341(9):2367–2379, 2018.



M. Develin, M. Macauley, and V. Reiner.

Toric partial orders.

*Trans. Amer. Math. Soc.*, 368(4):2263–2287, 2016.



# Sources II



H. Eriksson and K. Eriksson.

Conjugacy of Coxeter elements.

*Electron. J. Combin.*, 16(2):#R4, 2009.



D. Einstein, M. Farber, E. Gunawan, M. Joseph, M. Macauley, J. Propp, and S. Rubinstein-Salzedo.

Noncrossing partitions, toggles, and homomesies.

*Electron. J. Combin.*, 23(3), 2016.



D. Einstein and J. Propp.

Combinatorial, piecewise-linear, and birational homomesy for products of two chains.

*ArXiv 1310.5294v3*, 2018.



Eric Goles, Marco Montalva-Medel, Henning Mortveit, and Salvador Ramirez-Flandes.

Block invariance in elementary cellular automata.

*J. Cell. Autom.*, 10:119–135, 2015.





D. Grinberg and T. Roby.


Iterative properties of birational rowmotion.


*ArXiv 1402.6178v6*, 2014.


# Sources III

 S. Haddadan.  
Some instances of homomesy among ideals of posets.  
*ArXiv 1410.4819v3*, 2016.


 M. Joseph and T. Roby.  
Toggling independent sets of a path graph.  
*Electron J. Combin.*, 25(1):1–18, 2018.


 M. Macauley and H.S. Mortveit.  
Cycle equivalence of graph dynamical systems.  
*Nonlinearity*, 22:421–436, 2009.


 M. Macauley and H. Mortveit.  
Posets from admissible Coxeter sequences.  
*Electron. J. Combin.*, 18(1):#R197, 2011.


 M. Macauley, J. McCammond, and H.S. Mortveit.  
Order independence in asynchronous cellular automata.  
*J. Cell. Autom.*, 3(1):37–56, 2008.


# Sources IV

 M. Macauley, J. McCammond, and H. Mortveit.  
Dynamics groups of asynchronous cellular automata.  
*J. Algebraic Combin.*, 33(1):11–35, 2011.

 Y. Numata and Y. Yamanouchi.  
On the action of the toggle group of the Dynkin diagram of type A.  
*ArXiv 2103.16217*, 2021.

 T. K. Petersen.  
*Eulerian Numbers*.  
Springer, New York, 2015.

 J. Propp and T. Roby.  
Homomesy in products of two chains.  
*Electron. J. Combin.*, 22(3), 2015.

 O. Pretzel.  
On reorienting graphs by pushing down maximal vertices.  
*Order*, 3(2):135–153, 1986.



T. Roby.

Dynamical algebraic combinatorics and the homomesy phenomenon.  
*In Recent Trends in Combinatorics*, pages 619–652. Springer, 2016.



V. Salo.

Universal gates with wires in a row.  
*ArXiv 1809.08050*, 2018.



B. Schönfisch and A. de Roos.

Synchronous and asynchronous updating in cellular automata.  
*BioSystems*, 51(3):123–143, 1999.



J. Striker.

The toggle group, homomesy, and the Razumov-Stroganov correspondence.  
*Electron. J. Combin.*, 22(2):P2–57, 2015.  
Also available at *ArXiv 1503.08898v2*.



J. Striker.

Dynamical algebraic combinatorics: promotion, rowmotion, and resonance.  
*Notices Amer. Math. Soc.*, 64(6), 2017.

# Sources VI



J. Striker and N. Williams.

Promotion and rowmotion.

*European J. Combin.*, 33:1919–1942, 2012.



S. Wolfram.

*Theory and applications of cellular automata.*

1986.