The Mathematical Dynamics of Ciliate Gene Sorting Algorithms

Colby Brown, Claudia Sofia Carrillo Vazquez, Rashmika Goswami, Sam Heil

July 28, 2017
Overview

1 Introduction
   - Ciliates, Permutations, and Pointers
   - CDS: Definition and Example

2 Graphs and Matrices
   - Overlap Graphs, Parity Cuts, and Matrix Sortability
   - Adjacency-Precedence Matrix, Counting Formula, and Matrix Tightness

3 Games and Complexity
   - CDS Game Definition and 3- vs. 4- Element SP Cases
   - Space Complexity of CDS Game

4 Acknowledgements and Thank You
Introduction
Ciliates and Mathematics
Ciliates and Mathematics
Ciliates and Mathematics
Ciliates and Mathematics

Ciliates and Mathematics

$\pi = [3, 2, 5, 1, 4]$
$p = (1, 2)$
$q = (4, 5)$

Modified from: http://photobiology.info/LenciCiliatesFiles/Fig1.png
Ciliate Genome
Ciliate Genome

Micronucleus
Ciliate Genome
Ciliate Genome
CDS on DNA

Micronucleus

Macronucleus
CDS on DNA
CDS on DNA
CDS on DNA
CDS on DNA
CDS on DNA
CDS on DNA

Micronucleus

Macronucleus
CDS on DNA

Micronucleus

Macronucleus
Ciliate Genome

Micronucleus

Macronucleus

1 2 3 4
Ciliate Genome

Micronucleus

Macronucleus

arrangement

1  2  3  4
Ciliate Genome

Micronucleus

Macronucleus

Context Directed Swaps
From Genes to Mathematics

Context Directed Swaps
From Genes to Mathematics

\[ [4, 3, 5, 2, 1] \]

Context Directed Swaps

\[ [1, 2, 3, 4, 5] \]
Pointers

\[(n - 1) \cdot n + (n + 1) \cdot (n - 1) = n \cdot (n - 1)\]
$n$
Pointers

\[(n - 1) \ n \ (n + 1)\]
Pointers

\[(n - 1) \quad n \quad (n + 1)\]

\[(n - 1, n) \quad n \quad (n, n+1)\]
Pointers

\[(n - 1) \quad n \quad (n + 1)\]

\[(n-1,n) \quad n \quad (n,n+1)\]
Without pointers:

\[(3, 4), 4, (4, 5), (2, 3), (3, 4), (4, 5), (5, 6), (1, 2), (1, 2)\]

With pointers:
Pointers

\[ [4, 3, 5, 2, 1] \]

Without pointers
Pointers

Without pointers

\[
[4, 3, 5, 2, 1]
\]

With pointers

\[
[(3,4), (4,5):(2,3), (3,4):(4,5), (5,6):(1,2), (2,3):(0,1), (1,2)]
\]
Considering the Pointers

\[ [4, 3, 5, 2, 1] \]

Context Directed Swaps

\[ [1, 2, 3, 4, 5] \]
Considering the Pointers

\[
\begin{bmatrix}
(3,4) & 4 & (4,5) \cdot (2,3) & 3 & (3,4) \cdot (4,5) & 5 & (5,6) \cdot (1,2) & 2 & (2,3) \cdot (0,1) & 1 \\
(3,4) & 4 & (4,5) & 3 & (4,5) & 5 & (1,2) & 2 & (0,1) & 1
\end{bmatrix}
\]

Context Directed Swaps

\[
\begin{bmatrix}
(0,1) & 1 & (1,2) \cdot (1,2) & 2 & (2,3) \cdot (2,3) & 3 & (3,4) \cdot (3,4) & 4 & (4,5) \cdot (4,5) & 5 & (5,6)
\end{bmatrix}
\]
Context Directed Swaps (CDS)
Context Directed Swaps (CDS)

\[ \pi = [(3, 4), (4, 5), (2, 3), (3, 4), (4, 5), (5, 6), (1, 2), (2, 3), (0, 1), (1, 2)] \]

...
Context Directed Swaps (CDS)

\[ \pi = \left[ (3,4), 4 \rightarrow (4,5); (2,3), 3 \rightarrow (3,4); (4,5), 5 \rightarrow (5,6); (1,2), 2 \rightarrow (2,3); (0,1), 1 \rightarrow (1,2) \right] \]
Context Directed Swaps (CDS)

\[ \pi = [\begin{array}{cccccc}
(3,4) & 4 & (4,5),(2,3) & 3 & (3,4),(4,5) & 5 \\
(5,6),(1,2) & 2 & (2,3),(0,1) & 1 \\
\end{array}] \]

\[ \text{cds}_{p,q}(\pi) : \]

![Diagram showing the application of CDS operation on a permutation]
Context Directed Swaps (CDS)

\[ \pi = \begin{bmatrix} (3,4) & 4 & (4,5), (2,3) & 3 & (3,4), (4,5) & 5 & (5,6), (1,2) & 2 & (2,3), (0,1) & 1 & (1,2) \end{bmatrix} \]

\text{cds}_{p,q}(\pi):

- Operation applied to a permutation, taking pointers \( p \) and \( q \).
Context Directed Swaps (CDS)

\[ \pi = [ (3,4), 4 \rightarrow (4,5), (2,3) \rightarrow 3 (3,4), (4,5) \rightarrow 5 (5,6), (1,2) \rightarrow 2 (2,3), (0,1) \rightarrow 1 (1,2) ] \]

\[ \text{cds}_{p,q}(\pi) : \]

- Operation applied to a permutation, taking pointers \( p \) and \( q \).
- SWAPS the blocks that are in between the pointers \( p \) and \( q \).
Context Directed Swaps (CDS)

\[ \pi = [ (3,4), 4, (4,5), (2,3), 3, (3,4), (4,5), 5, (5,6), (1,2), 2, (2,3), (0,1), 1, (1,2) ] \]

\[ \text{cds}_{p,q}(\pi) : \]
- Operation applied to a permutation, taking pointers \( p \) and \( q \).
- SWAPS the blocks that are in between the pointers \( p \) and \( q \).
Context Directed Swaps

\[ [4, 3, 5, 2, 1] \]

\[ [(3, 4) \quad 4 \quad (4, 5) ; (2, 3) \quad 3 \quad (3, 4) ; (4, 5) \quad 5 \quad (5, 6) ; (1, 2) \quad 2 \quad (2, 3) ; (0, 1) \quad 1 \quad (1, 2)] \]
Context Directed Swaps

\[
\begin{bmatrix}
4, & 3, & 5, & 2, & 1 \\
(3,4), & (4,5), & (2,3), & (3,4), & (4,5), & (5,6), & (1,2), & (2,3), & (0,1), & 1
\end{bmatrix}
\]
Context Directed Swaps

\[ [5, 2, 3, 4, 1] \]
Context Directed Swaps

\[ [5, 2, 3, 4, 1] \]

\[ \begin{array}{cccccc}
5 & (4,5) & 2 & (2,3) & 3 & (3,4) \\
(5,6) & (2,3) & (3,4) & (4,5) & (1,2) \\
(1,2) & (1,2) & (1,2) & (1,2) & (1,2) \\
\end{array} \]
Context Directed Swaps

\[
[5, 2, 3, 4, 1]
\]

\[
[(4,5), (5,6), (1,2)]
\]

\[
[(2,3), (2,3)]
\]

\[
[(3,4), (3,4)]
\]

\[
[(4,5), (0,1)]
\]
Context Directed Swaps

\[
[1, 2, 3, 4, 5]
\]

\[
[(0,1), (1,2), (2,3), (3,4), (4,5)]
\]
Context Directed Swaps

\[
[1, 2, 3, 4, 5]
\]

\[
[(0, 1) \quad 1 \quad (1, 2) \quad (1, 2) \quad 2 \quad (2, 3) \quad (2, 3) \quad 3 \quad (3, 4) \quad (3, 4) \quad 4 \quad (4, 5) \quad (4, 5) \quad 5 \quad (5, 6)]
\]
Observation: Not all permutations are sortable under \texttt{cds}.

Definition derived from the above: A non-\texttt{cds}-sortable permutation, after the maximum number of \texttt{cds} operations has been applied to it, arrives to a permutation called a \texttt{cds}-fixed point.

Example 1: [5,1,2,3,4].

Example 2: [3,4,5,1,2].
Observation: Not all permutations are sortable under \textit{cds}. 
Observation: Not all permutations are sortable under \texttt{cds}.

Definition derived from the above: A non \texttt{cds}-sortable permutation, after the max. number of \texttt{cds} operations has been applied to it, arrives to a permutation called a \texttt{cds-fixed point}.
The \textbf{cds}-fixed points

\textbf{Observation}: Not all permutations are sortable under \textbf{cds}.

\textbf{Definition derived from the above}: A non \textbf{cds}-sortable permutation, after the max. number of \textbf{cds} operations has been applied to it, arrives to a permutation called a \textbf{cds}-fixed point.

- Example 1: [5,1,2,3,4].
- Example 2: [3,4,5,1,2].
Remark: All fixed points of size $n$ are of the form $[k, k+1, \ldots, n, 1, \ldots, (k-1)]$.

Definition constructed from the above: The strategic pile of a permutation $\pi$, $SP(\pi)$, is the set of fixed points that indicate to which fixed points a permutation can reach.

In particular, the element $k-1$ in the SP corresponds to the fixed point $[k, \ldots, n, 1, \ldots, (k-1)]$. 
Remark: All \texttt{cds}-fixed points of size $n$ are of the form

$$[k, k + 1, ..., n, 1, ..., (k - 1)].$$
**Remark:** All $\text{cds}$-fixed points of size $n$ are of the form

$$[k, k + 1, ..., n, 1, ..., (k - 1)].$$

**Definition constructed from the above:** The *strategic pile* of a permutation $\pi$, $\text{SP}(\pi)$, is the set of fixed points that indicate to which fixed points can a permutation reach to.
Remark: All cds-fixed points of size $n$ are of the form

$$[k, k+1, ..., n, 1, ..., (k - 1)].$$

Definition constructed from the above: The strategic pile of a permutation $\pi$, $\text{SP}(\pi)$, is the set of fixed points that indicate to which fixed points can a permutation reach to.

- In particular, the element $k - 1$ in the SP corresponds to the fixed point $[k, k+1, ..., n, 1, ..., (k - 1)]$. 
Graphs and Matrices
Permutation $\rightarrow$ Overlap Graph

$[4, 3, 5, 2, 1]$
Permutation → Overlap Graph

0 [4 3 5 2 1] 6
Permutation $\rightarrow$ Overlap Graph

0 [ 4 3 5 2 1 ] 6

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes

CDS $\rightarrow$ [ ]

0, 1
5, 6
1, 2
4, 5
2, 3
3, 4
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Permutation $\rightarrow$ Overlap Graph

CDS $\rightarrow$ [4, 1, 2, 3, 5]

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Permutation $\rightarrow$ Overlap Graph

CDS $\rightarrow$ [ 5 2 3 4 1 ]

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Permutation → Overlap Graph

CDS → [ 4  5  2  3  1 ]

Pointers → Nodes

Overlaps → Edges
Permutation → Overlap Graph

Pointers → Nodes
Overlaps → Edges
Permutation → Overlap Graph

CDS → [3 4 5 2 1]

Pointers → Nodes

Overlaps → Edges
Permutation $\rightarrow$ Overlap Graph

Pointers $\rightarrow$ Nodes

Overlaps $\rightarrow$ Edges
Definition

Let $G$ be a graph with an edge between nodes $p$ and $q$. Then, performing gcds on the pointer pair $p, q$ is equivalent to the following:

For each pair of nodes $i, j$, swap the existence of the edge between $i$ and $j$ if $f_p(i)f_q(j) + f_p(j)f_q(i) = 1 \mod 2$, where

$$f_n(x) = \begin{cases} 
1 & \text{edge between } n \text{ and } x \text{ exists} \\
0 & \text{otherwise}
\end{cases}$$
GCDS Sortability

Definition
A graph is gcds sortable if, after finitely many applications of the gcds operation, we reach the discrete graph (the graph with no edges).

Definition
The roots of a graph that represents a permutation of size $n$ are the nodes that represent the $(0, 1)$ and the $(n, n+1)$ pointers (the nodes in shape of a diamond).
Definition

A graph is *gcds sortable* if, after finitely many applications of the gcds operation, we reach the discrete graph (the graph with no edges).
GCDS Sortability

**Definition**

A graph is *gcds sortable* if, after finitely many applications of the gcds operation, we reach the discrete graph (the graph with no edges).

**Definition**

The *roots* of a graph that represents a permutation of size $n$ are the nodes that represent the $(0, 1)$ and the $(n, n+1)$ pointers (the nodes in shape of a diamond).
A parity cut of $G = (V, E)$ is a partition of $V$ into two sets $V_1, V_2$, such that for each vertex, the number of edges between that vertex and vertices in the opposite partition is even.
### Definition (2015 REU - Adapted)

A parity cut of $G = (V, E)$ is a partition of $V$ into two sets $V_1, V_2$, such that for each vertex, the number of edges between that vertex and vertices in the opposite partition is even.

### Definition

A generalized parity cut of $G = (V, E)$ is a set $S \subseteq V$ such that, for each vertex in the graph, the number of edges between that vertex and vertices in $S$ is even.
**Parity Cuts**

**Definition**

A generalized parity cut of $G = (V, E)$ is a set $S \subseteq V$ such that, for each vertex in the graph, the number of edges between that vertex and vertices in $S$ is even.

**Example:**

![Graph Diagram]
Parity Cuts

**Definition**

A generalized parity cut of \( G = (V, E) \) is a set \( S \subseteq V \) such that, for each vertex in the graph, the number of edges between that vertex and vertices in \( S \) is even.

**Example:**
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- **a:** \( f_p(a) = 1, f_q(a) = 0 \)
  (edges with \( u \) are switched)
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

\[ \begin{align*} 
\text{a:} & \quad f_p(a) = 1, f_q(a) = 0 \\
& \text{(edges with } u \text{ are switched)} \\
\text{b:} & \quad f_p(b) = 0, f_q(b) = 1 \\
& \text{(edges with } u \text{ are not switched)} 
\end{align*} \]
Case 1: $f_p(u) = 0, f_q(u) = 1$, $q \in S$, $u, p \notin S$

- **a:** $f_p(a) = 1, f_q(a) = 0$ (edges with $u$ are switched)
- **b:** $f_p(b) = 0, f_q(b) = 1$ (edges with $u$ are not switched)
- **c:** $f_p(c) = 1, f_q(c) = 1$ (edges with $u$ are switched)
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- \( p \) is adjacent to \( i + k + 1 \) vertices in \( S \), so \( i + k \) must be odd
Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- \( p \) is adjacent to \( i + k + 1 \) vertices in \( S \), so \( i + k \) must be odd
- \( q \) is adjacent to \( j + k \) vertices in \( S \), so \( j + k \) must be even
Case 1: $f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S$

The edges between $u$ and vertices in $\{a_t | t = 1, ..., i\} \cup \{c_t | t = 1, ..., i\}$ are switched.
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- The edges between \( u \) and vertices in \( \{a_t | t = 1, \ldots, i\} \cup \{c_t | t = 1, \ldots, i\} \) are switched.
- The edge between \( u \) and \( q \) is removed.
Generalized Parity Cut Preservation

Case 1: $f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S$

Suppose $u$ is adjacent to $x$ vertices in $\{a_t | t = 1, \ldots, i\} \bigcup \{c_t | t = 1, \ldots, i\}$

Since these edges are switched, $u$ loses $x + 1$ edges and gains $i + k - x$. Change in degree of $u$ is $i + k - 2x - 1$, which is even. Parity is preserved.
Case 1: $f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S$

- Suppose $u$ is adjacent to $x$ vertices in $\{a_t | t = 1, \ldots, i\} \cup \{c_t | t = 1, \ldots, i\}$
- Since these edges are switched, $u$ loses $x + 1$ edges and gains $i + k - x$
Case 1: $f_p(u) = 0$, $f_q(u) = 1$, $q \in S$, $u, p \not\in S$

- Suppose $u$ is adjacent to $x$ vertices in $\{a_t| t = 1, \ldots, i\} \cup \{c_t| t = 1, \ldots, i\}$
- Since these edges are switched, $u$ loses $x + 1$ edges and gains $i + k - x$
- Change in degree of $u$ is $i + k - 2x - 1$, which is even
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- Suppose \( u \) is adjacent to \( x \) vertices in \( \{a_t | t = 1, \ldots, i\} \cup \{c_t | t = 1, \ldots, i\} \)
- Since these edges are switched, \( u \) loses \( x + 1 \) edges and gains \( i + k - x \)
- Change in degree of \( u \) is \( i + k - 2x - 1 \), which is even
- Parity is preserved
Case 1: $f_p(u) = 0$, $f_q(u) = 1$, $q \in S$, $u, p \notin S$

Similar for all cases
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \notin S \)

- Similar for all cases
- In other direction: can place \( p \) and \( q \) so parity is preserved for those vertices as well
Generalized Parity Cut Preservation

Case 1: \( f_p(u) = 0, f_q(u) = 1, q \in S, u, p \not\in S \)

- Similar for all cases
- In other direction: can place \( p \) and \( q \) so parity is preserved for those vertices as well
- Generalized parity cut is preserved in both directions
Theorem

*Generalized parity cut is preserved in both directions*
Graph Sortability

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Generalized parity cut is preserved in both directions</em></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>The only fixed point graph (a graph on which gcds cannot be performed) with a generalized parity cut for each root containing that root and not the other is the discrete graph.</td>
</tr>
</tbody>
</table>
Graph Sortability

**Theorem**

*Generalized parity cut is preserved in both directions*

**Observation**

The only fixed point graph (a graph on which gcds cannot be performed) with a generalized parity cut for each root containing that root and not the other is the discrete graph.

**Theorem**

*A two-rooted graph with roots \(x, y\) is gcds-sortable if and only if for each of \(x, y\) there is a generalized parity cut containing that root and not the other.*
### Definition (Adjacency Matrix)

The adjacency matrix $M$ of a graph $G = (V, E)$ is a $|V| \times |V|$ matrix such that:

$$M_{ij} = \begin{cases} 
1 & \text{if there is an edge between vertex } i \text{ and vertex } j \\
0 & \text{otherwise}
\end{cases}$$
### Definition (Adjacency Matrix)

The adjacency matrix $M$ of a graph $G = (V, E)$ is a $|V| \times |V|$ matrix such that:

$$M_{ij} = \begin{cases} 1 & \text{if there is an edge between vertex } i \text{ and vertex } j \\ 0 & \text{otherwise} \end{cases}$$

### Definition (Characteristic Vector)

For a graph $G = (V, E)$, $V = \{v_1, ..., v_n\}$, the characteristic vector of a set $S \subseteq V$ is the vector $\vec{v} \in GF(2)^n$ such that

$$v_i = \begin{cases} 1 & \text{if } v_i \in S \\ 0 & \text{otherwise} \end{cases}$$
Parity Cut Example

Overlap Graph

Adjacency Matrix

Characteristic Vector

of Parity Cut
Parity Cut Example

Overlap Graph

Adjacency Matrix

\[
\begin{bmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0
\end{bmatrix}
\]
Parity Cut Example

Overlap Graph

Adjacency Matrix

Characteristic Vector of Parity Cut
Parity Cut Example

Overlap Graph

Adjacency Matrix

Characteristic Vector of Parity Cut

$$\begin{bmatrix}
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0
\end{bmatrix} \times \begin{bmatrix}
1 \\
1 \\
1 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix} = \begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
1 \\
0
\end{bmatrix}$$
Theorem

Given an overlap graph $G$ with adjacency matrix $M$, the kernel of $M$ is the set of characteristic vectors of all parity cuts of $G$. 
The matrix cds operation $MCDS$ on rows $p, q$ is defined:

$$MCDS_{p,q}(M) = M + MI_{pq}M = (I + MI_{pq})M$$

where

$$(I_{pq})_{ij} = \begin{cases} 
1 & \text{if } i = p, j = q \text{ or } i = q, j = p \\
0 & \text{otherwise}
\end{cases}$$

A matrix is mcds sortable if, after finitely many applications of the mcds operation, we reach the zero matrix.
Matrix Sortability

Theorem

Let $M$ be a matrix and $M' = MCDS(M)$. The kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 + \vec{x}_n = 1$ if and only if the kernel of $M'$ does as well.
Matrix Sortability

**Theorem**

Let $M$ be a matrix and $M' = MCDS(M)$. The kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 + \vec{x}_n = 1$ if and only if the kernel of $M'$ does as well.

**Observation:** The only fixed point matrix such that the kernel of the adjacency matrix of $G$ contains an element $\vec{x}$ such that $\vec{x}_0 = 1$ and $\vec{x}_n = 0$ and an element $\vec{y}$ such that $\vec{y}_0 = 0$ and $\vec{y}_n = 1$ is the zero matrix.
Matrix Sortability

**Theorem**

Let $M$ be a matrix and $M' = \text{MCDS}(M)$. The kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 + \vec{x}_n = 1$ if and only if the kernel of $M'$ does as well.

**Observation:** The only fixed point matrix such that the kernel of the adjacency matrix of $G$ contains an element $\vec{x}$ such that $\vec{x}_0 = 1$ and $\vec{x}_n = 0$ and an element $\vec{y}$ such that $\vec{y}_0 = 0$ and $\vec{x}_n = 1$ is the zero matrix.

**Theorem**

A matrix $M$ is mcds sortable if and only if the kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 = 1$ and $\vec{x}_n = 0$ and an element $\vec{y}$ such that $\vec{y}_0 = 0$ and $\vec{x}_n = 1$. 
Converting Between Permutations and Matrices

- Given a permutation $\pi$, the adjacency matrix $A$ of the overlap graph of $\pi$ can be computed directly without the intermediate step of converting to a graph.
Given a permutation $\pi$, the adjacency matrix $A$ of the overlap graph of $\pi$ can be computed directly without the intermediate step of converting to a graph.

Similarly, if $A$ is an adjacency matrix of a permutation $\pi$, the permutation $\pi$ can be efficiently determined from $A$. 

\[\pi = [4, 3, 5, 1, 2] \iff A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}\]
Converting Between Permutations and Matrices

- Given a permutation $\pi$, the adjacency matrix $A$ of the overlap graph of $\pi$ can be computed directly without the intermediate step of converting to a graph.
- Similarly, if $A$ is an adjacency matrix of a permutation $\pi$, the permutation $\pi$ can be efficiently determined from $A$.
- Both of these conversions can be computed in $O(n^2)$ time for a permutation of length $n$. 

\[ \pi = [4, 3, 5, 1, 2] \implies A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \]
Converting Between Permutations and Matrices

- Given a permutation \( \pi \), the adjacency matrix \( A \) of the overlap graph of \( \pi \) can be computed directly without the intermediate step of converting to a graph.
- Similarly, if \( A \) is an adjacency matrix of a permutation \( \pi \), the permutation \( \pi \) can be efficiently determined from \( A \).
- Both of these conversions can be computed in \( O(n^2) \) time for a permutation of length \( n \).

\[
\pi = [4, 3, 5, 1, 2] \quad \iff \quad A = \begin{pmatrix}
0 & 0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0
\end{pmatrix}
\]
Counting Formula for GCDS-Sortable Graphs

Sortability Criterion

A matrix $M$ is mcds sortable if and only if the kernel of $M$ contains an element $\vec{x}$ such that $x_0 = 1$ and $x_n = 0$ and an element $\vec{y}$ such that $y_0 = 0$ and $y_n = 1$. 
## Counting Formula for GCDS-Sortable Graphs

### Sortability Criterion

A matrix $M$ is mcds sortable if and only if the kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 = 1$ and $\vec{x}_n = 0$ and an element $\vec{y}$ such that $\vec{y}_0 = 0$ and $\vec{y}_n = 1$.

**Restated Sortability Criterion:**

\[
M = \begin{pmatrix}
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
\end{pmatrix} \implies \vec{m}_0 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}, \quad A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad \vec{m}_n = \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]
Counting Formula for GCDS-Sortable Graphs

Sortability Criterion

A matrix $M$ is mcds sortable if and only if the kernel of $M$ contains an element $\vec{x}$ such that $\vec{x}_0 = 1$ and $\vec{x}_n = 0$ and an element $\vec{y}$ such that $\vec{y}_0 = 0$ and $\vec{y}_n = 1$.

Restated Sortability Criterion:

$$M = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix} \quad \Longrightarrow \quad m_0 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} , \quad A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} , \quad m_n = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

$M\vec{x} = \vec{0}$, $x_0 = 1$, $x_n = 0 \iff A\vec{u} = m_0$ for some $\vec{u} \in \mathbb{Z}_{2}^{n-2}$

$M\vec{y} = \vec{0}$, $y_0 = 0$, $y_n = 1 \iff A\vec{v} = m_n$ for some $\vec{v} \in \mathbb{Z}_{2}^{n-2}$
Reformulated Sortability Criterion

A matrix \( M = [m_0 \mid A \mid m_n] \) is mcds sortable if and only if there are vectors \( \vec{u} \) and \( \vec{v} \) such that \( A\vec{u} = m_0 \) and \( A\vec{v} = m_n \).
Reformulated Sortability Criterion

A matrix $M = [\vec{m}_0 \mid A \mid \vec{m}_n]$ is mcds sortable if and only if there are vectors $\vec{u}$ and $\vec{v}$ such that $A\vec{u} = \vec{m}_0$ and $A\vec{v} = \vec{m}_n$.

Form of sortable matrix with central submatrix $C$ for vectors $\vec{u}$, $\vec{v}$:

$$
\begin{pmatrix}
(C \vec{u}) \cdot \vec{u} & (C \vec{u})_1 & (C \vec{u})_2 & \ldots & (C \vec{u})_n & (C \vec{u}) \cdot \vec{v} \\
(C \vec{u})_1 & c_{11} & c_{12} & \ldots & c_{1n} & (C \vec{v})_1 \\
(C \vec{u})_2 & c_{21} & c_{22} & \ldots & c_{2n} & (C \vec{v})_2 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
(C \vec{u})_n & c_{n1} & c_{n2} & \ldots & c_{nn} & (C \vec{v})_n \\
(C \vec{v}) \cdot \vec{u} & (C \vec{v})_1 & (C \vec{v})_2 & \ldots & (C \vec{v})_n & (C \vec{v}) \cdot \vec{v}
\end{pmatrix}
$$
Counting Formula for GCDS-Sortable Graphs

- We have $C\vec{u} = C\vec{u}'$ iff $C(\vec{u}' - \vec{u}) = \vec{0}$, so $(\vec{u}' - \vec{u}) \in \ker(C)$ and thus $C\vec{u} = C\vec{u}'$ and $\vec{u}' = \vec{u} + \vec{w}$ for some $\vec{w} \in \ker(C)$.
- Similarly, $C\vec{v} = C\vec{v}'$ iff $\vec{v}' = \vec{v} + \vec{y}$ for some $\vec{y} \in \ker(C)$.
- Each sortable adjacency matrix obtained from $C$ is counted $|\ker(C)|^2$ times.
- Since we’re working over $\mathbb{F}_2$, we have that there are $2^n$ choices for each of $\vec{u}$ and $\vec{v}$ and that $|\ker(C)| = 2^{\text{nullity}(C)}$ and $|\ker(C)|^2 = 4^{\text{nullity}(C)}$.
- Thus, there are $\frac{4^n}{4^{\text{nullity}(C)}} = 4^{\text{rank}(C)}$ distinct MCDS-sortable matrices with central submatrix $C$. 
Counting Formula for GCDS-Sortable Graphs

- We have $C \vec{u} = C \vec{u}'$ iff $C(\vec{u}' - \vec{u}) = \vec{0}$, so $(\vec{u}' - \vec{u}) \in \ker(C)$ and thus $C \vec{u} = C \vec{u}' \vec{u}' = \vec{u} + \vec{w}$ for some $\vec{w} \in \ker(C)$.
- Similarly, $C \vec{v} = C \vec{v}'$ iff $\vec{v}' = \vec{v} + \vec{y}$ for some $\vec{y} \in \ker(C)$.
- Each sortable adjacency matrix obtained from $C$ is counted $|\ker(C)|^2$ times.
- Since we’re working over $\mathbb{F}_2$, we have that there are $2^n$ choices for each of $\vec{u}$ and $\vec{v}$ and that $|\ker(C)| = 2^{\text{nullity}(C)}$ and $|\ker(C)|^2 = 4^{\text{nullity}(C)}$.
- Thus, there are $\frac{4^n}{4^{\text{nullity}(C)}} = 4^{\text{rank}(C)}$ distinct MCDS-sortable matrices with central submatrix $C$. 
Theorem (MacWilliams, 1969)

The number of $n \times n$ adjacency matrices of rank $k$ over the field $\mathbb{F}_2$ is 0 when $k$ is odd, and when $k = 2s$ it is

\[
\prod_{i=1}^{s} \frac{2^{2i-2}}{2^{2i} - 1} \cdot \prod_{i=0}^{2s-1} (2^n - i - 1).
\]
We can combine the above formula with the fact that there are $4^k$ GCDS-sortable graphs on $n$ vertices for each $(n - 2) \times (n - 2)$ adjacency matrix of rank $k$ to count the number of GCDS-sortable graphs on $n$ vertices.
We can combine the above formula with the fact that there are $4^k$ GCDS-sortable graphs on $n$ vertices for each $(n - 2) \times (n - 2)$ adjacency matrix of rank $k$ to count the number of GCDS-sortable graphs on $n$ vertices.

**Theorem**

The number of GCDS-sortable two-rooted graphs on $n$ vertices is

$$\sum_{s=0}^{\lfloor n/2 \rfloor - 1} 2^s(s+3) \left( \frac{\prod_{i=0}^{2s-1} (2^n - 2 - i - 1)}{\prod_{i=1}^{s} (2^{2i} - 1)} \right).$$
A "(r-)simple graph" is an alternating cycle graph where each cycle has length 2.
Definition

A "(r-)simple graph" is an alternating cycle graph where each cycle has length 2.

Theorem

All graphs have a corresponding simple graph with the same sorting distance by reversal.
Hartmin & Verbin used this same logic but with a slightly different definition.

**Definition**

A "(t-)simple graph" is an alternating cycle graph where each cycle has length exactly 3.

There is no corresponding theorem for t-simple graphs.
Consider a t-simple alternating cycle graph. Let each 3-cycle correspond to a row and column in the adjacency matrix $A$, as such:
Consider a t-simple alternating cycle graph. Let each 3-cycle correspond to a row and column in the adjacency matrix $A$, as such:

$$A_{ij} = \begin{bmatrix} v^1_i v^2_j & v^1_i v^1_j \\ v^2_i v^2_j & v^2_i v^1_j \end{bmatrix}$$

This matrix completely encodes the alternating cycle graph.
Consider a t-simple alternating cycle graph. Let each 3-cycle correspond to a row and column in the adjacency matrix $A$, as such:

$$A_{ij} = \begin{bmatrix} v_i^1 v_j^2 & v_i^1 v_j^1 \\ v_i^2 v_j^2 & v_i^2 v_j^1 \end{bmatrix}$$

This matrix completely encodes the alternating cycle graph.
Matrix Clicking

Definition
Hartmin & Verbin Define “clicking” a matrix on a non-zero diagonal element $A_{ii}$ as $A - A_{↓,v} (A - 1) v v^T$. Then, clicking is equivalent to a set of elementary row operations zeroing column $i$, and then zeroing row $i$.

Theorem
Hartmin & Verbin An $n \times n$ matrix is sortable iff exactly $n$ clicking operations result in the zero matrix. But, we don’t have a definition of clicking for swapping.
Matrix Clicking

**Definition**

Hartmin & Verbin Define "clicking" a matrix on a non-zero diagonal element $A_{ii}$ as

\[
A^{(*)v} \triangleq A - A_{\downarrow v} A_{v v}^{-1} A_{v \rightarrow}.
\]
Hartmin & Verbin Define “clicking” a matrix on a non-zero diagonal element $A_{ii}$ as

$$A^{(*v)} \triangleq A - A_{\downarrow v} A_{v v}^{-1} A_{v \rightarrow}.$$  

Then, clicking is equivalent to a set of elementary row operations zeroing column $i$, and then zeroing row $i$. 
Matrix Clicking

**Definition**

Hartmin & Verbin Define "clicking" a matrix on a non-zero diagonal element $A_{ii}$ as

$$A^{(*v)} \triangleq A - A_{\downarrow v} A_{v v}^{-1} A_{v \rightarrow}.$$

Then, clicking is equivalent to a set of elementary row operations zeroing column $i$, and then zeroing row $i$.

**Theorem**

*Hartmin & Verbin* An $n \times n$ matrix is sortable iff exactly $n$ clicking operations result in the zero matrix.
Matrix Clicking

**Definition**

Hartmin & Verbin Define "clicking" a matrix on a non-zero diagonal element $A_{ii}$ as

$$A^{(\star v)} \triangleq A - A_{\downarrow v} A_{v v}^{-1} A_{v \rightarrow}.$$  

Then, clicking is equivalent to a set of elementary row operations zeroing column $i$, and then zeroing row $i$.

**Theorem**

*Hartmin & Verbin* An $n \times n$ matrix is sortable iff exactly $n$ clicking operations result in the zero matrix.

**But**, we don’t have a definition of clicking for swapping.
Relation among different graphs of size four
Relation among different graphs of size four
Relation among different graphs of size four
Relation among different graphs of size four
We suggest future work in the area of matrix tightness:
We suggest future work in the area of matrix tightness:

- Have an \( \left( \frac{n}{2} \right)^2 \times \left( \frac{n}{2} \right)^2 \) matrix of pairs 2-cycles
Looking Forwards

We suggest future work in the area of matrix tightness:

- Have an $\left(\frac{n}{2}\right)^2 \times \left(\frac{n}{2}\right)^2$ matrix of *pairs* 2-cycles
- Diagonal elements represent ”sortable” pairs
Looking Forwards

We suggest future work in the area of matrix tightness:

- Have an \((\frac{n}{2})^2 \times \left(\frac{n}{2}\right)^2\) matrix of pairs 2-cycles
- Diagonal elements represent ”sortable” pairs
- Clicking may still correspond to a sorting process
We suggest future work in the area of matrix tightness:

- Have an \( \left( \frac{n}{2} \right)^2 \times \left( \frac{n}{2} \right)^2 \) matrix of *pairs* 2-cycles
- Diagonal elements represent "sortable" pairs
- Clicking may still correspond to a sorting process
- There may or may not be redundant information
We suggest future work in the area of matrix tightness:

- Have an \( (\frac{n}{2})^2 \times (\frac{n}{2})^2 \) matrix of pairs 2-cycles
- Diagonal elements represent ”sortable” pairs
- Clicking may still correspond to a sorting process
- There may or may not be redundant information

We leave these conjectures for future REU students to investigate further.
Games and
Complexity
The $CDS(\pi, A)$ game was first introduced in *Sorting Permutations: Games, Genomes, and Cycles* where:

- $\pi$ is a non-sortable permutation under CDS
- $A \subseteq \text{SP}(\pi)$

The game is played by players ONE and TWO. ONE starts by applying some valid CDS operation to $\pi$ with some pointers $p$ and $q$. Given $\sigma = CDS_p, q(\pi)$, then TWO applies some valid CDS operation to $\sigma$. Etc. Once a fixed point is reached, ONE wins if such fixed point is in $A$; otherwise, TWO wins. The fixed point $F_k = [k k+1 k+2 ... n]$ is a winning position for ONE if and only if $k-1$ is in $A$: $\text{SP}(F_k) = \{k-1\}$. 
The \( CDS(\pi, A) \) game was first introduced in Sorting Permutations: Games, Genomes, and Cycles where:

- \( \pi \) is a non-sortable permutation under CDS
- \( A \subseteq SP(\pi) \)
The $CDS(\pi, A)$ game was first introduced in *Sorting Permutations: Games, Genomes, and Cycles* where:

- $\pi$ is a non-sortable permutation under CDS
- $A \subseteq SP(\pi)$

It is played by players ONE and TWO.
The \( CDS(\pi, A) \) game was first introduced in *Sorting Permutations: Games, Genomes, and Cycles* where:

- \( \pi \) is a non-sortable permutation under CDS
- \( A \subseteq SP(\pi) \)

It is played by players ONE and TWO.

ONE starts by applying some valid CDS operation to \( \pi \) with some pointers \( p \) and \( q \). Given \( \sigma = CDS_{p,q}(\pi) \), then TWO applies some valid CDS operation to \( \sigma \). Etc.
The \( \text{CDS}(\pi, A) \) game was first introduced in *Sorting Permutations: Games, Genomes, and Cycles* where:

- \( \pi \) is a non-sortable permutation under CDS
- \( A \subseteq \text{SP}(\pi) \)

It is played by players ONE and TWO.

ONE starts by applying some valid CDS operation to \( \pi \) with some pointers \( p \) and \( q \). Given \( \sigma = \text{CDS}_{p,q}(\pi) \), then TWO applies some valid CDS operation to \( \sigma \). Etc.

Once a fixed point is reached, ONE wins if such fixed point is in \( A \); otherwise, TWO wins.
The \( \text{CDS}(\pi, A) \) game was first introduced in *Sorting Permutations: Games, Genomes, and Cycles* where:
- \( \pi \) is a non-sortable permutation under CDS
- \( A \subseteq SP(\pi) \)

It is played by players ONE and TWO.

ONE starts by applying some valid CDS operation to \( \pi \) with some pointers \( p \) and \( q \). Given \( \sigma = \text{CDS}_{p,q}(\pi) \), then TWO applies some valid CDS operation to \( \sigma \). Etc.

Once a fixed point is reached, ONE wins if such fixed point is in \( A \); otherwise, TWO wins.

The fixed point \( F_k = [k \ k+1 \ k+2 \ldots \ n \ 1 \ 2 \ldots \ k-1] \) is a winning position for ONE if and only if \( k-1 \) is in \( A \):
\[
SP(F_k) = \{k - 1\}.
\]
Example CDS Game

- Starting Position: \( \pi_0 = [5, 2, 6, 1, 4, 3] \), \( A = \{1, 2, 3\} \), \( SP(\pi_0) = \{1, 2, 3, 4, 5\} \).
Example CDS Game

- Starting Position: \( \pi_0 = [5, 2, 6, 1, 4, 3] \), \( A = \{1, 2, 3\} \), 
  \( SP(\pi_0) = \{1, 2, 3, 4, 5\} \).

- ONE applies the CDS move with pointers (1, 2) and (5, 6) to \( \pi_0 \), yielding \( \pi_1 = [5, 6, 1, 2, 4, 3] \).
Example CDS Game

- Starting Position: \( \pi_0 = [5, 2, 6, 1, 4, 3] \), \( A = \{1, 2, 3\} \), \( SP(\pi_0) = \{1, 2, 3, 4, 5\} \).
- ONE applies the CDS move with pointers (1, 2) and (5, 6) to \( \pi_0 \), yielding \( \pi_1 = [5, 6, 1, 2, 4, 3] \).
- TWO applies the CDS move with pointers (2, 3) and (3, 4) to \( \pi_1 \), yielding \( \pi_2 = [5, 6, 1, 2, 3, 4] \).

This is a CDS fixed point, so the game ends. The strategic pile is \( SP(\pi_2) = \{4\} \). \( 4 \not\in A \), so TWO wins.
Example CDS Game

- Starting Position: $\pi_0 = [5, 2, 6, 1, 4, 3]$, $A = \{1, 2, 3\}$, $SP(\pi_0) = \{1, 2, 3, 4, 5\}$.

- ONE applies the CDS move with pointers (1, 2) and (5, 6) to $\pi_0$, yielding $\pi_1 = [5, 6, 1, 2, 4, 3]$.

- TWO applies the CDS move with pointers (2, 3) and (3, 4) to $\pi_1$, yielding $\pi_2 = [5, 6, 1, 2, 3, 4]$.

- This is a CDS fixed point, so the game ends.
Example CDS Game

Starting Position: $\pi_0 = [5, 2, 6, 1, 4, 3]$, $A = \{1, 2, 3\}$, $SP(\pi_0) = \{1, 2, 3, 4, 5\}$.

ONE applies the CDS move with pointers (1, 2) and (5, 6) to $\pi_0$, yielding $\pi_1 = [5, 6, 1, 2, 4, 3]$.

TWO applies the CDS move with pointers (2, 3) and (3, 4) to $\pi_1$, yielding $\pi_2 = [5, 6, 1, 2, 3, 4]$.

This is a CDS fixed point, so the game ends.

The strategic pile is $SP(\pi_2) = \{4\}$. $4 \not\in A$, so TWO wins.
Theorem (Adamyk, et al. 2013)

Let $\pi \in S_n$ be a permutation that is not CDS-sortable. If the strategic pile of $\pi$ has at most two elements, then for any nonempty subset $A \subseteq SP(\pi)$ ONE has a winning strategy in the game $CDS(\pi, A)$.

We generalized this result to permutations with strategic piles of size 3 and 4.
Lemma

Let $\pi \in S_n$ be a permutation with ordered strategic pile $(b_1, b_2, b_3)$. Then, if $|A| = 1$, ONE has a winning strategy unless the elements $b_1, b_2, b_3, b_1 + 1, b_2 + 1, b_3 + 1$ are all distinct and appear in $\pi$ in the order $b_3 + 1, b_3, b_2 + 1, b_2, b_1 + 1, b_1$, in which case TWO can win.
Lemma

Let $\pi \in S_n$ be a permutation with ordered strategic pile $(b_1, b_2, b_3)$. Then, if $|A| = 1$, ONE has a winning strategy unless the elements $b_1, b_2, b_3, b_1 + 1, b_2 + 1, b_3 + 1$ are all distinct and appear in $\pi$ in the order $b_3 + 1, b_3, b_2 + 1, b_2, b_1 + 1, b_1$, in which case TWO can win.

\[
\begin{bmatrix}
  r + 1 & \cdots & r & q + 1 & \cdots & q & p + 1 & \cdots & p
\end{bmatrix}
\]

TWO wins for permutations of this form since ONE cannot remove a pair of TWO’s fixed points.
Lemma

Let $\pi \in S_n$ be a permutation with ordered strategic pile $(b_1, b_2, b_3)$. Then, if $|A| = 1$, ONE has a winning strategy unless the elements $b_1, b_2, b_3, b_1 + 1, b_2 + 1, b_3 + 1$ are all distinct and appear in $\pi$ in the order $b_3 + 1, b_3, b_2 + 1, b_2, b_1 + 1, b_1$, in which case TWO can win.

\[
\begin{bmatrix}
  r + 1 & \cdots & r & q + 1 & \cdots & q & p + 1 & \cdots & p
\end{bmatrix}
\]

TWO wins for permutations of this form since ONE cannot remove a pair of TWO’s fixed points.

\[
\begin{bmatrix}
  r + 1 & \cdots & q & p + 1 & \cdots & r & q + 1 & \cdots & p
\end{bmatrix}
\]

ONE wins for permutations of this form by removing a pair of TWO’s fixed points.
The CDS game becomes more difficult to analyze in the case with the 4-element strategic pile since it is no longer necessarily in the players interest to remove strategic pile elements whenever possible.

The winning player can be determined in specific cases, but in general there is no simple condition based only on the ordering of the strategic pile elements.

In the game $CDS(\pi, A)$ with $\pi = [2 \ 5 \ 8 \ 1 \ 10 \ 3 \ 7 \ 9 \ 4 \ 6]$ and $A = \{3, 9\}$, $SP(\pi) = \{1, 3, 6, 9\}$, ONE has a winning strategy but is required to remove no strategic pile elements on the first turn.
Space Complexity of CDS Game

Decision Problem [Jansen, 2014]

**CDS GAME:**
Given a permutation \( \pi \in S_n \) that is not CDS-sortable and a subset \( A \) of the strategic pile of \( \pi \), does player ONE have a winning strategy in the game \( CDS(\pi, A) \)?
Decision Problem [Jansen, 2014]

**CDS GAME:**
Given a permutation $\pi \in S_n$ that is not CDS-sortable and a subset $A$ of the strategic pile of $\pi$, does player ONE have a winning strategy in the game $CDS(\pi, A)$?

**Theorem**

*The solution to this decision problem for a polynomial $\pi$ of length $n$ can be computed in space polynomial in $n$: the problem CDS GAME is in PSPACE.*
Acknowledgements

- Professor Marion Scheepers
- Professor Liljana Babinkostova
- Boise State University Complexity Across Disciplines REU
- DMS-1659872 NSF REU Grant
Expressing Gratitude:
Personal Thank You
References

*Parity Cuts In 2-Rooted Graphs: Unofficial Summary*

*Sorting Permutations: Games, Genomes, and Cycles*, arXiv:1410.2353v3 [math.CO]

C.L. Jansen, M. Scheepers, S.L. Simon, and E. Tatum
*Permutation Sorting and a Game on Graphs*, arXiv:1411.5429v1 [math.CO]

M. Gaetz, B. Molokach, M. Scheepers and M.A. Shanks
*Quantifying CDS Sortability of Permutations Using Strategic Piles: Unofficial Summary*
E. Zermelo

Über eine Anwendung der Mengenlehre auf die Theorie des Schachspiels
Proceedings of the Fifth International Congress of Mathematics
(1913), 501-504

Hartman T. and Verbin E.,
Matrix Tightness: A Linear-Algebraic Framework for Sorting By Transpositions