# Group Testing:
## Something old, Something new, Something borrowed

Nikhil Karamchandani
IIT Bombay

CNI Seminar Series, IISc

Feb. 18, 2025

# Group testing: origin

- First studied by Robert Dorfman in US in the 1940s for syphilis testing amongst soldiers.


"BY THE WAY, WHAT ARE WE IN LINE FOR?"

Img. source: https://www.cartoonstock.com/

# Group testing: origin

- First studied by Robert Dorfman in US in the 1940s for syphilis testing amongst soldiers.
- Can do individual testing, inefficient since most tests will be negative.


"BY THE WAY, WHAT ARE WE IN LINE FOR?"

Img. source: https://www.cartoonstock.com/

# Group testing: origin

- First studied by Robert Dorfman in US in the 1940s for syphilis testing amongst soldiers.
- Can do individual testing, inefficient since most tests will be negative.
- *Key idea*: 'pool' samples from many soldiers and test it

"BY THE WAY, WHAT ARE WE IN LINE FOR?"

Img. source: https://www.cartoonstock.com/

# Group testing: origin

- First studied by Robert Dorfman in US in the 1940s for syphilis testing amongst soldiers.
- Can do individual testing, inefficient since most tests will be negative.
- *Key idea*: ‘pool’ samples from many soldiers and test it
  - *Negative test*: all in the pool are uninfected
  - *Positive test*: at least one soldier is infected

"BY THE WAY, WHAT ARE WE IN LINE FOR?"

Img. source: https://www.cartoonstock.com/

# Group testing: origin

- First studied by Robert Dorfman in US in the 1940s for syphilis testing amongst soldiers.
- Can do individual testing, inefficient since most tests will be negative.
- *Key idea*: 'pool' samples from many soldiers and test it
  - *Negative test*: all in the pool are uninfected
  - *Positive test*: at least one soldier is infected
- *Goal*: design pooling strategies to minimize number of tests.


"BY THE WAY, WHAT ARE WE IN LINE FOR?"

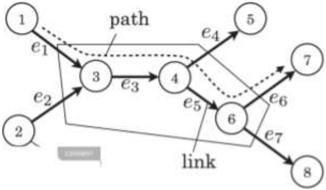
---

Img. source: https://www.cartoonstock.com/

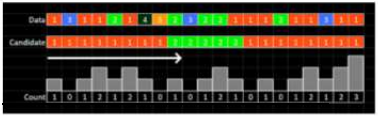
# Group testing: applications


Imgs: online sources

# Group testing: applications


"BY THE WAY, WHAT ARE WE IN LINE FOR?"



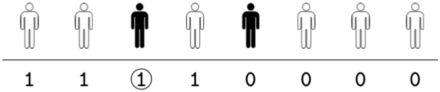

Imgs: online sources

# Model

- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

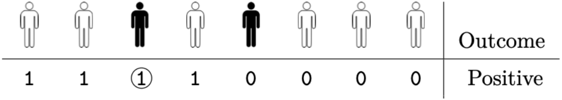
# Model




- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ can be represented by $\mathbf{x} \in \{0, 1\}^n$.
  - $\mathbf{x}_i^t = 1$ if item $i$ included in test.

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Model




- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ can be represented by $\mathbf{x} \in \{0, 1\}^n$.
  - $\mathbf{x}_i^t = 1$ if item $i$ included in test.
- Outcome $y_t = \bigvee_{i \in \mathcal{K}} \mathbf{x}_i^t$.

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Problem

| | | | | | | | | Outcome |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | ① | 1 | 0 | 0 | 0 | 0 | Positive |
| 0 | 0 | 0 | 0 | ① | 1 | 1 | 1 | Positive |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Negative |
| 0 | 0 | ① | 0 | 0 | 0 | 0 | 0 | Positive |
| 0 | 0 | ① | 0 | ① | 1 | 0 | 0 | Positive |
| 0 | 0 | 0 | 0 | ① | 0 | 0 | 0 | Positive |

- Test design $\mathbf{X} \in \{0,1\}^{T \times n}$, output $\mathbf{y} = \bigvee_{i \in \mathcal{K}} \mathbf{X}_i$.

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Problem

| ? | ? | ? | ? | ? | ? | ? | ? | **y** |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Testing design $\mathbf{X} \in \{0,1\}^{T \times n}$, output $\mathbf{y} = \bigvee_{i \in \mathcal{K}} \mathbf{X}_i$.
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Problem

| ? | ? | ? | ? | ? | ? | ? | ? | y |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Testing design $\mathbf{X} \in \{0,1\}^{T \times n}$, output $\mathbf{y} = \bigvee_{i \in \mathcal{K}} \mathbf{X}_i$.
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.
- *Goal*: Given $n, k$, find feasible testing designs of minimum size.

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Problem

| ? | ? | ? | ? | ? | ? | ? | ? | **y** |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Testing design $\mathbf{X} \in \{0,1\}^{T \times n}$, output $\mathbf{y} = \bigvee_{i \in \mathcal{K}} \mathbf{X}_i$.
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.
- *Goal*: Given $n, k$, find feasible testing designs of minimum size.
  - Explicit constructions, efficient decoding rules

Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Lower bound

| ? | ? | ? | ? | ? | ? | ? | ? | y |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Feasible testing design $\Longrightarrow \exists$ injective function from set of possible defective sets to the set of possible outputs

# Lower bound

| ? | ? | ? | ? | ? | ? | ? | ? | y |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Feasible testing design $\Longrightarrow \exists$ injective function from set of possible defective sets to the set of possible outputs

$$2^T \geq \sum_{i=0}^{k} \binom{n}{i}$$

# Lower bound

| ? | ? | ? | ? | ? | ? | ? | ? | y |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |

- Feasible testing design $\implies \exists$ injective function from set of possible defective sets to the set of possible outputs

$$2^T \geq \sum_{i=0}^{k} \binom{n}{i} \implies T \geq \Omega\left(k \log \frac{n}{k}\right)$$

# Achievable strategies: adaptive testing

- Sequential design of tests

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$
  - Conduct binary search.

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$
  - Conduct binary search. Needs at most $\lceil \log n \rceil$ tests.

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$
  - Conduct binary search. Needs at most $\lceil \log n \rceil$ tests.
- $k > 1$
  - Repeat above process, removing one defective in each round.

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$
  - Conduct binary search. Needs at most $\lceil \log n \rceil$ tests.
- $k > 1$
  - Repeat above process, removing one defective in each round.
  - Needs at most $O(k \log n)$ tests.

# Achievable strategies: adaptive testing

- Sequential design of tests
- $k = 1$
  - Conduct binary search. Needs at most $\lceil \log n \rceil$ tests.
- $k > 1$
  - Repeat above process, removing one defective in each round.
  - Needs at most $O(k \log n)$ tests.
  - More sophisticated algorithms achieve $O\left(k \log \frac{n}{k}\right)$ tests.
- Order-optimal w.r.t lower bound.

# Group testing: bounds

| | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log \left(\frac{n}{k}\right)$ | $k \log \left(\frac{n}{k}\right)$ |

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

Not 3-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

Not 3-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies & (k-1)\text{-disjunct} \\ \impliedby & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

2-disjunct

- *$t$-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

$\mathcal{K} = \{1, 2\}$, O/p is $\bigvee_{i \in [1:2]} \mathbf{X}_i$

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies (k-1)\text{-disjunct} \\ \impliedby k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

$\mathcal{K} = \{1, 2\}$, O/p is $\bigvee_{i \in [1:2]} \mathbf{X}_i$

$\mathbf{X}_3 \not\preceq \bigvee_{i \in [1:2]} \mathbf{X}_i$

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

$\mathcal{K} = \{1, 2\}$, O/p is $\bigvee_{i \in [1:2]} \mathbf{X}_i$

$\mathbf{X}_3 \not\preceq \bigvee_{i \in [1:2]} \mathbf{X}_i$

$\implies \exists$ witness test for item 3

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies & (k-1)\text{-disjunct} \\ \impliedby & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

$\mathcal{K} = \{1, 2\}$, O/p is $\bigvee_{i \in [1:2]} \mathbf{X}_i$

$\mathbf{X}_4 \not\preceq \bigvee_{i \in [1:2]} \mathbf{X}_i$

$\implies \exists$ witness test for item 4

2-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies & (k-1)\text{-disjunct} \\ \impliedby & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

2-disjunct

Say $k = 2$, $\mathbf{X}$ is $k$-disjunct

$\mathcal{K} = \{1, 2\}$, O/p is $\bigvee_{i \in [1:2]} \mathbf{X}_i$

$\mathbf{X}_4 \not\preceq \bigvee_{i \in [1:2]} \mathbf{X}_i$

$\implies \exists$ witness test for item 4

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies (k-1)\text{-disjunct} \\ \impliedby k\text{-disjunct} \end{cases}$$

- Simple decoding algorithm: if all tests involving an item o/p positive, mark defective.

# Non-adaptive testing: bounds

- *Lower bound*: $\Omega(k^2 \log_k n)$ tests; connection to $k$-cover families [D'yachkov & Rykov'82, Furedi'96]

# Non-adaptive testing: bounds

- *Lower bound*: $\Omega(k^2 \log_k n)$ tests; connection to $k$-cover families [D'yachkov & Rykov'82, Furedi'96]
- *Random construction*: $O\left(k^2 \log \frac{n}{k}\right)$ tests; choose each entry i.i.d. $\sim Ber(1/(k+1))$.

# Non-adaptive testing: bounds

- *Lower bound*: $\Omega(k^2 \log_k n)$ tests; connection to $k$-cover families [D'yachkov & Rykov'82, Furedi'96]
- *Random construction*: $O\left(k^2 \log \frac{n}{k}\right)$ tests; choose each entry i.i.d. $\sim Ber(1/(k+1))$.
- *Explicit construction*: $O\left(k^2 \min\{\log_k^2 n, \log n\}\right)$ tests; based on a concatenated code construction [Kautz & Singleton'64, Porat & Rotschild'08]

# Group testing: bounds

| | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log \left(\frac{n}{k}\right)$ | $k \log \left(\frac{n}{k}\right)$ |
| Non-adaptive | $k^2 \log_k n$ | $k^2 \min\{\log_k^2 n, \log n\}$ |

# Cascaded Group Testing

with Waqar Mirza and Niranjan Balachandran

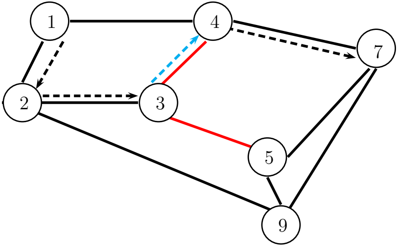
Information Theory Workshop (ITW), Nov. 2024

https://arxiv.org/abs/2405.17917

# Motivation

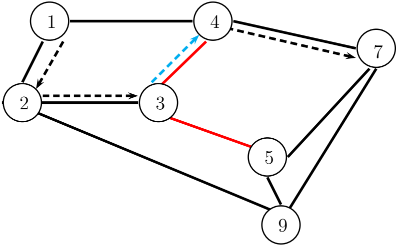
# Motivation

- Network tomography



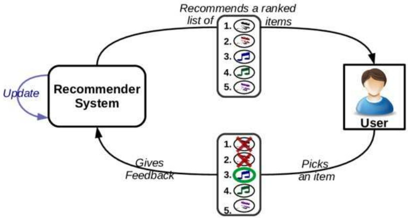
# Motivation

- Network tomography



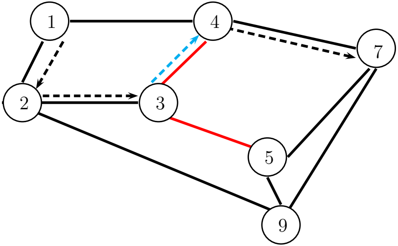

- Recommendation systems

[Img. source: “On Recommendation Systems in a Sequential Context", Frederic Guillou]

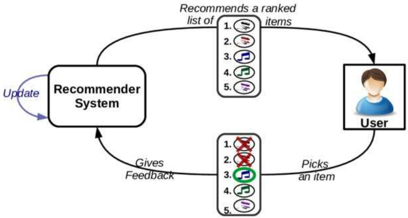


# Motivation

- Network tomography




- Recommendation systems
  [Img. source: “On Recommendation Systems in a Sequential Context", Frederic Guillou]




- Cascading bandits / OLTR

# Model

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$

# Model

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | |

- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ is associated with an ordered subset of items $(i_1, i_2, \ldots, i_{|t|})$.

# Model

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |

- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ is associated with an ordered subset of items $(i_1, i_2, \ldots, i_{|t|})$.
- Test $t$ returns first defective item in the sequence.

# Model

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |

- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ is associated with an ordered subset of items $(i_1, i_2, \ldots, i_{|t|})$.
- Test $t$ returns first defective item in the sequence.

# Model

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |

- $n$ items $\mathcal{V}$, unknown subset $\mathcal{K}$ of defectives with size at most $k$.
  - $k \ll n$
- Each test $t$ is associated with an ordered subset of items $(i_1, i_2, \ldots, i_{|t|})$.
- Test $t$ returns first defective item in the sequence.
  - 0 if no defective in test.

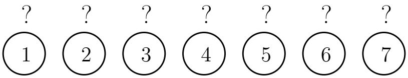
# Problem

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |
| $t_4$ | 6 3 4 7 | 0 |
| $t_5$ | 7 5 4 6 | 5 |

- Testing design $\mathbf{X} = \{t_1, t_2, ..., t_T\}$, output $\mathbf{y} = (y_1, y_2, \ldots, y_T)$

# Problem

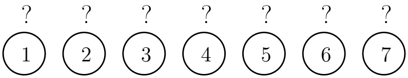



$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |
| $t_4$ | 6 3 4 7 | 0 |
| $t_5$ | 7 5 4 6 | 5 |

- Testing design $\mathbf{X} = \{t_1, t_2, ..., t_T\}$, output $\mathbf{y} = (y_1, y_2, \ldots, y_T)$
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.

# Problem

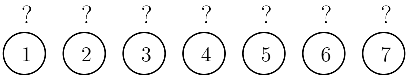



$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |
| $t_4$ | 6 3 4 7 | 0 |
| $t_5$ | 7 5 4 6 | 5 |

- Testing design $\mathbf{X} = \{t_1, t_2, ..., t_T\}$, output $\mathbf{y} = (y_1, y_2, \ldots, y_T)$
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.
- *Goal*: Given $n, k$, find feasible testing designs of minimum size.

# Problem




$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |
| $t_4$ | 6 3 4 7 | 0 |
| $t_5$ | 7 5 4 6 | 5 |

- Testing design $\mathbf{X} = \{t_1, t_2, ..., t_T\}$, output $\mathbf{y} = (y_1, y_2, \ldots, y_T)$
- $\mathbf{X}$ is *feasible* if we can recover any $\mathcal{K}$ from $\mathbf{y}$, $|\mathcal{K}| \leq k$.
- *Goal*: Given $n, k$, find feasible testing designs of minimum size.
  - Explicit constructions, efficient decoding rules

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 5 |
| $t_2$ | 1 2 3 4 5 6 | 2 |
| $t_3$ | 1 3 4 6 7 | 0 |
| $t_4$ | 6 3 4 7 | 0 |
| $t_5$ | 7 5 4 6 | 5 |

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome | BGT outcome |
|---|---|---|---|
| $t_1$ | 3 1 5 7 | 5 | Yes |
| $t_2$ | 1 2 3 4 5 6 | 2 | Yes |
| $t_3$ | 1 3 4 6 7 | 0 | No |
| $t_4$ | 6 3 4 7 | 0 | No |
| $t_5$ | 7 5 4 6 | 5 | Yes |

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome | BGT outcome |
|---|---|---|---|
| $t_1$ | 3 1 5 7 | 5 | Yes |
| $t_2$ | 1 2 3 4 5 6 | 2 | Yes |
| $t_3$ | 1 3 4 6 7 | 0 | No |
| $t_4$ | 6 3 4 7 | 0 | No |
| $t_5$ | 7 5 4 6 | 5 | Yes |

- CGT test provides at least as much information as BGT test.

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome | BGT outcome |
|---|---|---|---|
| $t_1$ | 3 1 5 7 | 5 | Yes |
| $t_2$ | 1 2 3 4 5 6 | 2 | Yes |
| $t_3$ | 1 3 4 6 7 | 0 | No |
| $t_4$ | 6 3 4 7 | 0 | No |
| $t_5$ | 7 5 4 6 | 5 | Yes |

- CGT test provides at least as much information as BGT test.
- Feasible design under BGT $\Longrightarrow$ Feasible design under CGT

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome | BGT outcome |
|---|---|---|---|
| $t_1$ | 3 1 5 7 | 5 | Yes |
| $t_2$ | 1 2 3 4 5 6 | 2 | Yes |
| $t_3$ | 1 3 4 6 7 | 0 | No |
| $t_4$ | 6 3 4 7 | 0 | No |
| $t_5$ | 7 5 4 6 | 5 | Yes |

- CGT test provides at least as much information as BGT test.
- Feasible design under BGT $\Longrightarrow$ Feasible design under CGT
  - Upper bounds for BGT are also upper bounds for CGT

# Cascaded GT vs Binary GT

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome | BGT outcome |
|---|---|---|---|
| $t_1$ | 3 1 5 7 | 5 | Yes |
| $t_2$ | 1 2 3 4 5 6 | 2 | Yes |
| $t_3$ | 1 3 4 6 7 | 0 | No |
| $t_4$ | 6 3 4 7 | 0 | No |
| $t_5$ | 7 5 4 6 | 5 | Yes |

- CGT test provides at least as much information as BGT test.
- Feasible design under BGT $\Longrightarrow$ Feasible design under CGT
  - Upper bounds for BGT are also upper bounds for CGT
- How much can the additional information help?

# Cascaded GT vs Binary GT: bounds

| *BGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log\left(\frac{n}{k}\right)$ | $k \log\left(\frac{n}{k}\right)$ |
| Non-adaptive | $k^2 \log_k n$ | $k^2 \min\{\log_k^2 n, \log n\}$ |

# Cascaded GT vs Binary GT: bounds

| *BGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log \left(\frac{n}{k}\right)$ | $k \log \left(\frac{n}{k}\right)$ |
| Non-adaptive | $k^2 \log_k n$ | $k^2 \min\{\log_k^2 n, \log n\}$ |
| *CGT* | Lower bound | Upper bound |
| Adaptive | | $k \log \left(\frac{n}{k}\right)$ |
| Non-adaptive | | $k^2 \min\{\log_k^2 n, \log n\}$ |

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


- Sequential design of tests

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 1 2 3 4 5 6 7 | |

- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 1 2 3 4 5 6 7 | 2 |

- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | | | | | | | CGT outcome |
|---|---|---|---|---|---|---|---|---|
| $t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 2 |
| $t_2$ | 1 | 3 | 4 | 5 | 6 | 7 | | 5 |

- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | | | | | | | CGT outcome |
|---|---|---|---|---|---|---|---|---|
| $t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 2 |
| $t_2$ | 1 | 3 | 4 | 5 | 6 | 7 | | 5 |

- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.
- Needs at most $k$ tests,

# Adaptive testing

$\mathcal{V} = \{1, 2, \ldots, 7\}, \mathcal{K} = \{2, 5\}$


| | Test | | | | | | | CGT outcome |
|---|---|---|---|---|---|---|---|---|
| $t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 2 |
| $t_2$ | 1 | 3 | 4 | 5 | 6 | 7 | | 5 |

- Sequential design of tests
- Initialise $\mathcal{V} = \{1, 2, \ldots, n\}$, $\hat{\mathcal{K}} \leftarrow \emptyset$, $i \leftarrow 1$ and run the loop:
  1. Run a test with items in $\mathcal{V} \backslash \hat{\mathcal{K}}$ in an arbitrary order.
  2. If the test returns 0, terminate and return $\hat{\mathcal{K}}$.
  3. If the test returns $v$, then update $\hat{\mathcal{K}} \leftarrow \hat{\mathcal{K}} \cup \{v\}$.
  4. Update $i \leftarrow i + 1$. If $i > k$, terminate and return $\hat{\mathcal{K}}$.
- Needs at most $k$ tests, optimal in the worst-case.

# Cascaded group testing: bounds

| *BGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log \left(\frac{n}{k}\right)$ | $k \log \left(\frac{n}{k}\right)$ |
| Non-adaptive | $k^2 \log_k n$ | $k^2 \min\{\log_k^2 n, \log n\}$ |

| *CGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k$ | $k$ |
| Non-adaptive | | $k^2 \min\{\log_k^2 n, \log n\}$ |

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$:

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$
- $k = 2$:

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$
- $k = 2$: two tests suffice,

$$t_1 = (1, 2, ..., n), \ t_2 = (n, n - 1, ..., 1)$$

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$
- $k = 2$: two tests suffice,

$$t_1 = (1, 2, ..., n),\ t_2 = (n, n-1, ..., 1)$$

- Optimal for $k = 1, 2$.

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$
- $k = 2$: two tests suffice,

$$t_1 = (1, 2, ..., n), \ t_2 = (n, n-1, ..., 1)$$

- Optimal for $k = 1, 2$. BGT would need $\Omega(\log n)$ tests.

# Non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$: one test suffices, $t_1 = (1, 2, ..., n)$
- $k = 2$: two tests suffice,

$$t_1 = (1, 2, ..., n), \; t_2 = (n, n-1, ..., 1)$$

- Optimal for $k = 1, 2$. BGT would need $\Omega(\log n)$ tests.
- What about larger $k$?

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$


| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

- Testing design $\mathbf{X}$ is *feasible* if we can recover $\mathcal{K}$ from $\mathbf{y}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | |
| $t_2$ | 1 2 3 4 5 6 | |
| $t_3$ | 1 3 4 6 7 | |
| $t_4$ | 6 3 4 7 | |
| $t_5$ | 7 5 4 6 | |

- Testing design $\mathbf{X}$ is *feasible* if we can recover $\mathcal{K}$ from $\mathbf{y}$.
- Distinct outputs for each $\mathcal{K}_1 \neq \mathcal{K}_2$, s.t. $|\mathcal{K}_1|, |\mathcal{K}_2| \leq k$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

$\mathcal{K}_1 = \{1, 2, 3\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 3 |
| $t_2$ | 1 2 3 4 5 6 | 1 |
| $t_3$ | 1 3 4 6 7 | 1 |
| $t_4$ | 6 3 4 7 | 3 |
| $t_5$ | 7 5 4 6 | 0 |

- Testing design $\mathbf{X}$ is *feasible* if we can recover $\mathcal{K}$ from $\mathbf{y}$.
- Distinct outputs for each $\mathcal{K}_1 \neq \mathcal{K}_2$, s.t. $|\mathcal{K}_1|, |\mathcal{K}_2| \leq k$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

$\mathcal{K}_1 = \{1, 2, 3\}$
$\mathcal{K}_2 = \{1, 3\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 3 |
| $t_2$ | 1 2 3 4 5 6 | 1 |
| $t_3$ | 1 3 4 6 7 | 1 |
| $t_4$ | 6 3 4 7 | 3 |
| $t_5$ | 7 5 4 6 | 0 |

- Testing design **X** is *feasible* if we can recover $\mathcal{K}$ from **y**.
- Distinct outputs for each $\mathcal{K}_1 \neq \mathcal{K}_2$, s.t. $|\mathcal{K}_1|, |\mathcal{K}_2| \leq k$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

$\mathcal{K}_1 = \{1, 2, 3\}$
$\mathcal{K}_2 = \{1, 3\}$

| | Test | CGT outcome |
|---|---|---|
| $t_1$ | 3 1 5 7 | 3 |
| $t_2$ | 1 2 3 4 5 6 | 1 |
| $t_3$ | 1 3 4 6 7 | 1 |
| $t_4$ | 6 3 4 7 | 3 |
| $t_5$ | 7 5 4 6 | 0 |

- Testing design **X** is *feasible* if we can recover $\mathcal{K}$ from **y**.
- Distinct outputs for each $\mathcal{K}_1 \neq \mathcal{K}_2$, s.t. $|\mathcal{K}_1|, |\mathcal{K}_2| \leq k$.
- Analogue of disjunctness property under BGT.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$


| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

$\mathcal{K} = \{1, 2, 3\}$

$v = 1$

$v = 2$

$v = 3$

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

(1) (2) (3) (4) (5) (6) (7)

| | Test |
|---|---|
| $t_1$ | 3 1 5 7 |
| $t_2$ | 1 2 3 4 5 6 |
| $t_3$ | 1 3 4 6 7 |
| $t_4$ | 6 3 4 7 |
| $t_5$ | 7 5 4 6 |

$\mathcal{K} = \{1, 2, 3\}$
$v = 1$ ✓
$v = 2$ ×
$v = 3$ ✓

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$


| | Test | | | |
|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 |
| $t_2$ | 2 | 6 | 5 | 7 |
| $t_3$ | 3 | 7 | 5 | 6 |
| $t_4$ | 4 | 7 | 6 | 5 |

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| Test | | | | |
|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 |
| $t_2$ | 2 | 6 | 5 | 7 |
| $t_3$ | 3 | 7 | 5 | 6 |
| $t_4$ | 4 | 7 | 6 | 5 |

$\mathcal{K} = \{1, 2, 3\}$

$v = 1$ ✓

$v = 2$ ✓

$v = 3$ ✓

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| Test | | | | |
|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 |
| $t_2$ | 2 | 6 | 5 | 7 |
| $t_3$ | 3 | 7 | 5 | 6 |
| $t_4$ | 4 | 7 | 6 | 5 |

$\mathcal{K} = \{2, 5, 7\}$

$v = 2$ ✓

$v = 5$ ✓

$v = 7$ ✓

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

Test

| | | | | |
|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 |
| $t_2$ | 2 | 6 | 5 | 7 |
| $t_3$ | 3 | 7 | 5 | 6 |
| $t_4$ | 4 | 7 | 6 | 5 |

$\mathcal{K} = \{3, 4, 6\}$
$v = 3$ ✓
$v = 4$ ✓
$v = 6$ ✓

- *Feasibility condition*: $\forall\, \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test | | | |
|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 |
| $t_2$ | 2 | 6 | 5 | 7 |
| $t_3$ | 3 | 7 | 5 | 6 |
| $t_4$ | 4 | 7 | 6 | 5 |

- *Feasibility condition*: $\forall\, \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 7 6 7

| | Test | | | | CGT outcome |
|---|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 | 6 |
| $t_2$ | 2 | 6 | 5 | 7 | 6 |
| $t_3$ | 3 | 7 | 5 | 6 | 3 |
| $t_4$ | 4 | 7 | 6 | 5 | 4 |

$\mathcal{K} = \{3, 4, 6\}$

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.
- *Reconstruction*: $\hat{\mathcal{K}} = \{y_i : i \in [T], y_i \neq 0\}$

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test | | | | CGT outcome |
|---|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 | 6 |
| $t_2$ | 2 | 6 | 5 | 7 | 6 |
| $t_3$ | 3 | 7 | 5 | 6 | 3 |
| $t_4$ | 4 | 7 | 6 | 5 | 4 |

$\mathcal{K} = \{3, 4, 6\}$

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.
- *Reconstruction*: $\hat{\mathcal{K}} = \{y_i : i \in [T], y_i \neq 0\}$
- *Lower bound*: Any feasible design has at least $\lfloor \frac{k+1}{2} \rfloor \lceil \frac{k+1}{2} \rceil$ tests.

# Non-adaptive testing: feasibility

$\mathcal{V} = \{1, 2, \ldots, 7\}, k = |\mathcal{K}| = 3$

1 2 3 4 5 6 7

| | Test | | | | CGT outcome |
|---|---|---|---|---|---|
| $t_1$ | 1 | 5 | 6 | 7 | 6 |
| $t_2$ | 2 | 6 | 5 | 7 | 6 |
| $t_3$ | 3 | 7 | 5 | 6 | 3 |
| $t_4$ | 4 | 7 | 6 | 5 | 4 |

$\mathcal{K} = \{3, 4, 6\}$

- *Feasibility condition*: $\forall\ \mathcal{K} \subset V$ with $|\mathcal{K}| = k$, and for every $v \in \mathcal{K}$, $\exists$ test $t \in \mathbf{X}$ where $v$ appears before every other item in $\mathcal{K}$.
- *Reconstruction*: $\hat{\mathcal{K}} = \{y_i : i \in [T], y_i \neq 0\}$
- *Lower bound*: Any feasible design has at least $\lfloor \frac{k+1}{2} \rfloor \lceil \frac{k+1}{2} \rceil$ tests. *Erdős-Szekeres theorem* gives $\lfloor \log_2 \log_2 (n-1) \rfloor$ lower bound.

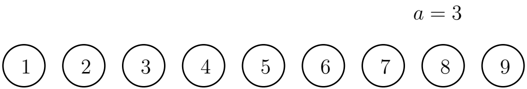
# Cascaded group testing: bounds

| *BGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log \left(\frac{n}{k}\right)$ | $k \log \left(\frac{n}{k}\right)$ |
| Non-adaptive | $k^2 \log_k n$ | $k^2 \min\{\log_k^2 n, \log n\}$ |

| *CGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k$ | $k$ |
| Non-adaptive | $\max\{k^2, \log\log n\}$ | $k^2 \min\{\log_k^2 n, \log n\}$ |

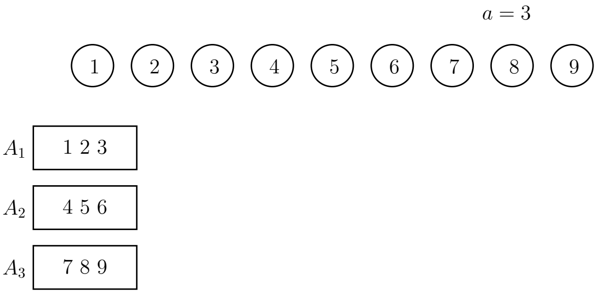
# Non-adaptive testing: $k = 3$

# Non-adaptive testing: $k = 3$




- Use feasible design $\mathbf{X}_1$ for $a$ items to create feasible design $\mathbf{X}_2$ for $a^2$ items.

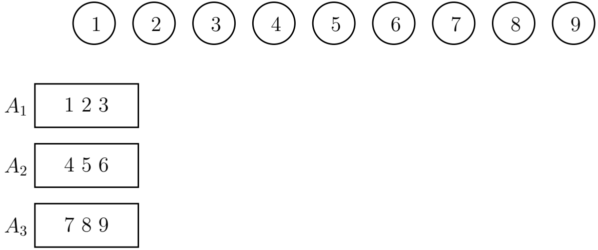
# Non-adaptive testing: $k = 3$




- Use feasible design $\mathbf{X}_1$ for $a$ items to create feasible design $\mathbf{X}_2$ for $a^2$ items.
- Partition $a^2$ items into disjoint sets $A_1, A_2, \ldots, A_a$ of size $a$ each.

# Non-adaptive testing: $k = 3$

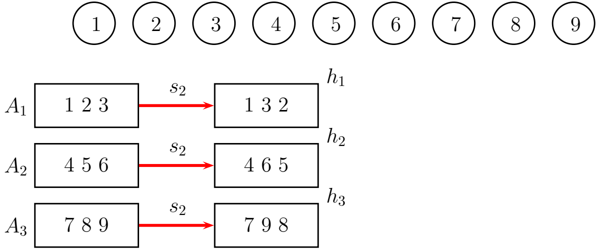
$a = 3, s_1 = (2, 3, 1), s_2 = (1, 3, 2)$




- Given permutations $s_1, s_2$ on $a$ items, permutation $s_3 = s_1 \circ s_2$ on $a^2$ items is given by:

# Non-adaptive testing: $k = 3$

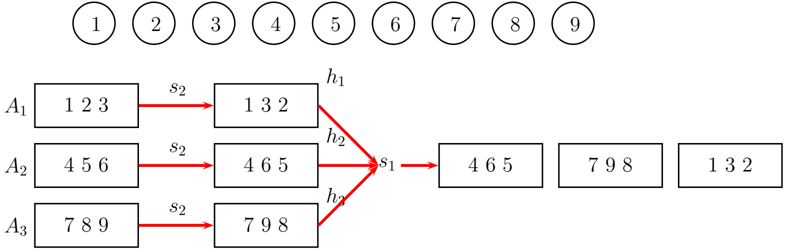
$a = 3, s_1 = (2,3,1), s_2 = (1,3,2)$




- Given permutations $s_1, s_2$ on $a$ items, permutation $s_3 = s_1 \circ s_2$ on $a^2$ items is given by:
  - For each $i$, arrange items of $A_i$ according to $s_2$. Call result $h_i$.

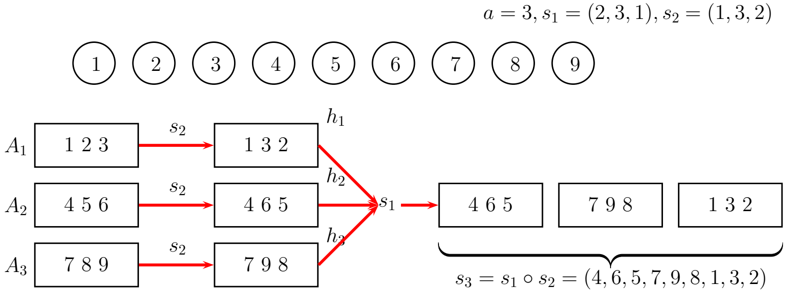
# Non-adaptive testing: $k = 3$

$a = 3, s_1 = (2, 3, 1), s_2 = (1, 3, 2)$




- Given permutations $s_1, s_2$ on $a$ items, permutation $s_3 = s_1 \circ s_2$ on $a^2$ items is given by:
  - For each $i$, arrange items of $A_i$ according to $s_2$. Call result $h_i$.
  - Arrange $h_1, h_2, ..., h_a$ according to $s_1$ to obtain $s_3$

# Non-adaptive testing: $k = 3$




- Given permutations $s_1, s_2$ on $a$ items, permutation $s_3 = s_1 \circ s_2$ on $a^2$ items is given by:
  - For each $i$, arrange items of $A_i$ according to $s_2$. Call result $h_i$.
  - Arrange $h_1, h_2, ..., h_a$ according to $s_1$ to obtain $s_3$

# Non-adaptive testing: $k = 3$

$a = 3$

1 2 3 4 5 6 7 8 9

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

- Start with feasible design $\mathbf{X}_1 = \{t_1, t_2, \ldots, t_{|\mathbf{X}_1|}\}$ for $a$ items.

# Non-adaptive testing: $k = 3$

$a = 3$


$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- Start with feasible design $\mathbf{X}_1 = \{t_1, t_2, \ldots, t_{|\mathbf{X}_1|}\}$ for $a$ items.
- Consider $\mathcal{F} := \{t_i \circ t_i : i \in [|\mathbf{X}_1|]\}$.

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

- Start with feasible design $\mathbf{X}_1 = \{t_1, t_2, \ldots, t_{|\mathbf{X}_1|}\}$ for $a$ items.
- Consider $\mathcal{F} := \{t_i \circ t_i : i \in [|\mathbf{X}_1|]\}$.
- Take $g_1 = (1, 2, ..., a)$ and $g_2 = (a, a-1, ..., 1)$.

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- Start with feasible design $\mathbf{X}_1 = \{t_1, t_2, \ldots, t_{|\mathbf{X}_1|}\}$ for $a$ items.
- Consider $\mathcal{F} := \{t_i \circ t_i : i \in [|\mathbf{X}_1|]\}$.
- Take $g_1 = (1, 2, ..., a)$ and $g_2 = (a, a-1, ..., 1)$. Consider $\mathcal{H} := \{g_i \circ g_j : i, j \in [2]\}$.

# Non-adaptive testing: $k = 3$

$a = 3$

1 2 3 4 5 6 7 8 9

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- Start with feasible design $\mathbf{X}_1 = \{t_1, t_2, \ldots, t_{|\mathbf{X}_1|}\}$ for $a$ items.
- Consider $\mathcal{F} := \{t_i \circ t_i : i \in [|\mathbf{X}_1|]\}$.
- Take $g_1 = (1, 2, ..., a)$ and $g_2 = (a, a-1, ..., 1)$. Consider $\mathcal{H} := \{g_i \circ g_j : i, j \in [2]\}$.
- Finally, design for $a^2$ items given by $\mathbf{X}_2 := \mathcal{F} \cup \mathcal{H}$.

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design;

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$
- Recursive design for $n$ items,

# Non-adaptive testing: $k = 3$

$a = 3$

1 2 3 4 5 6 7 8 9

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$
- Recursive design for $n$ items, with at most $O(\log \log n)$ tests.

# Non-adaptive testing: $k = 3$

$a = 3$

1 2 3 4 5 6 7 8 9

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$
- Recursive design for $n$ items, with at most $O(\log \log n)$ tests.
- Idea generalizes to any constant $k$,

# Non-adaptive testing: $k = 3$

$a = 3$

1 2 3 4 5 6 7 8 9

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | |
|---|---|
| $t_1 \circ t_1$ | 1 2 3 4 5 6 7 8 9 |
| $t_2 \circ t_2$ | 5 4 6 2 1 3 8 7 9 |
| $t_3 \circ t_3$ | 9 8 7 6 5 4 3 2 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | |
|---|---|
| $g_1 \circ g_1$ | 1 2 3 4 5 6 7 8 9 |
| $g_1 \circ g_2$ | 3 2 1 6 5 4 9 8 7 |
| $g_2 \circ g_1$ | 7 8 9 4 5 6 1 2 3 |
| $g_2 \circ g_2$ | 9 8 7 6 5 4 3 2 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$
- Recursive design for $n$ items, with at most $O(\log \log n)$ tests.
- Idea generalizes to any constant $k$, with at most $O((\log \log n)^{c_k})$ tests, where $c_k = 2^{(k-2)} - 1$.

# Non-adaptive testing: $k = 3$

$a = 3$

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$\mathbf{X}_1$

| | | | |
|---|---|---|---|
| $t_1$ | 1 | 2 | 3 |
| $t_2$ | 2 | 1 | 3 |
| $t_3$ | 3 | 2 | 1 |

$\mathcal{F}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $t_1 \circ t_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $t_2 \circ t_2$ | 5 | 4 | 6 | 2 | 1 | 3 | 8 | 7 | 9 |
| $t_3 \circ t_3$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

$g_1 = (1, 2, 3)$
$g_2 = (3, 2, 1)$

$\mathcal{H}$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $g_1 \circ g_1$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $g_1 \circ g_2$ | 3 | 2 | 1 | 6 | 5 | 4 | 9 | 8 | 7 |
| $g_2 \circ g_1$ | 7 | 8 | 9 | 4 | 5 | 6 | 1 | 2 | 3 |
| $g_2 \circ g_2$ | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |

- $\mathbf{X}_2$ is a feasible design; $|\mathbf{X}_2| \leq |\mathbf{X}_1| + 4$
- Recursive design for $n$ items, with at most $O(\log \log n)$ tests.
- Idea generalizes to any constant $k$, with at most $O((\log \log n)^{c_k})$ tests, where $c_k = 2^{(k-2)} - 1$.
- Can be much smaller than BGT which needs $\Omega(k^2 \log_k n)$ tests.

# Cascaded group testing: bounds for $k = O(1)$

| *BGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k \log n$ | $k \log n$ |
| Non-adaptive | $k^2 \log n$ | $k^2 \log n$ |

| *CGT* | Lower bound | Upper bound |
|---|---|---|
| Adaptive | $k$ | $k$ |
| Non-adaptive | $\max\{k^2, \log\log n\}$ | $\min\{(\log\log n)^{c_k}, k^2 \log n\}$ |

# Summary

- New variant of group testing

# Summary

- New variant of group testing
- Derived bounds under adaptive and non adaptive testing

# Summary

- New variant of group testing
- Derived bounds under adaptive and non adaptive testing
- Further directions:

# Summary

- New variant of group testing
- Derived bounds under adaptive and non adaptive testing
- Further directions:
  - General achievable strategies for any $k$
  - Close gap between upper and lower bounds
  - Noisy and constrained testing

Thanks

https://sites.google.com/site/nikhilkaram/

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

$\uparrow \; \uparrow \; \uparrow \; \uparrow$

2-disjunct

Not 3-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Say $k = 4$, $\mathbf{X}$ not $(k-1)$-disjunct

2-disjunct

Not 3-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Say $k = 4$, $\mathbf{X}$ not $(k-1)$-disjunct

$$\mathbf{X}_1 \preceq \bigvee_{i \in [2:4]} \mathbf{X}_i \Longrightarrow \bigvee_{i \in [2:4]} \mathbf{X}_i = \bigvee_{i \in [1:4]} \mathbf{X}_i$$

2-disjunct

Not 3-disjunct

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \Longrightarrow & (k-1)\text{-disjunct} \\ \Longleftarrow & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: disjunct testing matrix

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2-disjunct

Not 3-disjunct

Say $k = 4$, $\mathbf{X}$ not $(k-1)$-disjunct

$\mathbf{X}_1 \preceq \bigvee_{i\in[2:4]} \mathbf{X}_i \implies \bigvee_{i\in[2:4]} \mathbf{X}_i = \bigvee_{i\in[1:4]} \mathbf{X}_i$

O/p for $\mathcal{K} = \{2,3,4\}$ same as for $\mathcal{K} = \{1,2,3,4\}$

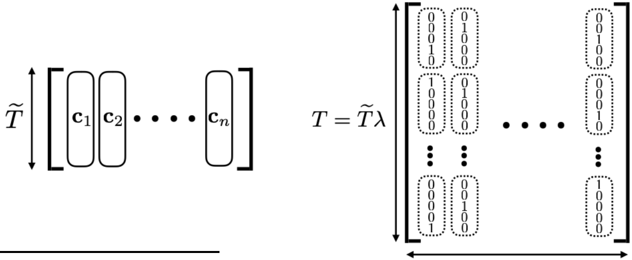
$\implies \mathbf{X}$ not feasible for $k = 4$.

- *t-disjunct matrix*: Union of any $t$ columns does not contain any other single column.
- With at most $k$ defectives,

$$\text{Feasible testing design matrix} \begin{cases} \implies & (k-1)\text{-disjunct} \\ \impliedby & k\text{-disjunct} \end{cases}$$

# Non-adaptive testing: bounds

- *Lower bound*: $\Omega(k^2 \log_k n)$ tests; connection to $k$-cover families [D'yachkov & Rykov'82, Furedi'96]
- *Random construction*: $O\left(k^2 \log \frac{n}{k}\right)$ tests; choose each entry i.i.d. $\sim Ber(1/(k+1))$.
- *Explicit construction*: $O\left(k^2 \min\{\log_k^2 n, \log n\}\right)$ tests; based on a concatenated code construction [Kautz & Singleton'64, Porat & Rotschild'08]




Img. source: Group Testing: An Information Theory Perspective, Now Publishers, 2019

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.

$$\mathbf{H}_{3,7}$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.

$$\mathbf{H}_{3,7}$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.
  - Item $j$ defective $\Rightarrow$ output $\mathbf{y} = \bigvee_{i \in \mathcal{K}} \mathbf{H}_i = \mathbf{H}_j$

$$\mathbf{H}_{3,7}$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.
  - Item $j$ defective $\Rightarrow$ output $\mathbf{y} = \mathbf{H}_j$

$$\mathbf{H}_{3,7} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$\mathbf{H}_5$ $\quad j = 5$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.
  - Item $j$ defective $\Rightarrow$ output $\mathbf{y} = \mathbf{H}_j$ $\Longrightarrow$ $\mathbf{H}$ is a feasible testing design.

$$\mathbf{H}_{3,7}$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$\mathbf{H}_5$ $\qquad j = 5$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.
  - Item $j$ defective $\Rightarrow$ output $\mathbf{y} = \mathbf{H}_j$ $\Longrightarrow$ $\mathbf{H}$ is a feasible testing design.
  - Works for $m$ s.t. $2^m - 1 \geq n \Longrightarrow$ needs $\lceil \log(n+1) \rceil$ tests.

$\mathbf{H}_{3,7}$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$\mathbf{H}_5$ $\quad j = 5$

# Achievable strategies: non-adaptive testing

- Testing design matrix has to be specified beforehand.
- $k = 1$
  - For $m \geq 1$, parity check matrix $\mathbf{H}$ of a binary Hamming code has dimension $m \times 2^m - 1$.
  - All columns are distinct non-zero binary vectors.
  - Item $j$ defective $\Rightarrow$ output $\mathbf{y} = \mathbf{H}_j$ $\Longrightarrow$ $\mathbf{H}$ is a feasible testing design.
  - Works for $m$ s.t. $2^m - 1 \geq n \Longrightarrow$ needs $\lceil \log(n+1) \rceil$ tests.
  - Near-optimal for $k = 1$.

$$\mathbf{H}_{3,7}$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$\mathbf{H}_5$ $\quad j = 5$

# Extensions and variants

# Extensions and variants

- *Decoding criterion*: small error probability, partial recovery

# Extensions and variants

- *Decoding criterion*: small error probability, partial recovery
- *Defectives prior*: combinatorial, i.i.d., bursty

# Extensions and variants

- *Decoding criterion*: small error probability, partial recovery
- *Defectives prior*: combinatorial, i.i.d., bursty
- *Noise model*: symmetric, Z channel, dilution, erasure

# Extensions and variants

- *Decoding criterion*: small error probability, partial recovery
- *Defectives prior*: combinatorial, i.i.d., bursty
- *Noise model*: symmetric, Z channel, dilution, erasure
- *Testing model*: threshold, quantitative, concomitant, tropical, graph-constrained

# Cascaded group testing: lower bound

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2 + 1$, there is a monotone subsequence of length $a + 1$.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2 + 1$, there is a monotone subsequence of length $a + 1$.
- Say $k = 3$, $n = 2^{2^r} + 1$; and we have a feasible design with $T \leq r = \log_2 \log_2(n-1)$ tests.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2+1$, there is a monotone subsequence of length $a+1$.
- Say $k=3$, $n = 2^{2^r}+1$; and we have a feasible design with $T \leq r = \log_2 \log_2(n-1)$ tests.
- $\exists$ a subset of $n_1 = 2^{2^{(r-1)}}+1$ items whose relative ordering in $t_1$ is monotone.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2 (n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2 + 1$, there is a monotone subsequence of length $a + 1$.
- Say $k = 3$, $n = 2^{2^r} + 1$; and we have a feasible design with $T \leq r = \log_2 \log_2 (n-1)$ tests.
- $\exists$ a subset of $n_1 = 2^{2^{(r-1)}} + 1$ items whose relative ordering in $t_1$ is monotone.
- Amongst these, $\exists$ a monotone subsequence of size $n_2 = 2^{2^{(r-2)}} + 1$ in $t_2$.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2 + 1$, there is a monotone subsequence of length $a + 1$.
- Say $k = 3$, $n = 2^{2^r} + 1$; and we have a feasible design with $T \leq r = \log_2 \log_2(n-1)$ tests.
- $\exists$ a subset of $n_1 = 2^{2^{(r-1)}} + 1$ items whose relative ordering in $t_1$ is monotone.
- Amongst these, $\exists$ a monotone subsequence of size $n_2 = 2^{2^{(r-2)}} + 1$ in $t_2$.
- Proceeding inductively, we get $n_T = 2^{2^{(r-T)}} + 1 \geq 3$ items, such that in each $t_i$ they appear in increasing or decreasing order.

# Cascaded group testing: lower bound

- For $k \geq 3$, any feasible design has more than $\lfloor \log_2 \log_2(n-1) \rfloor$ tests.
- *Erdős-Szekeres theorem*: For any sequence of length $a^2 + 1$, there is a monotone subsequence of length $a + 1$.
- Say $k = 3$, $n = 2^{2^r} + 1$; and we have a feasible design with $T \leq r = \log_2 \log_2(n-1)$ tests.
- $\exists$ a subset of $n_1 = 2^{2^{(r-1)}} + 1$ items whose relative ordering in $t_1$ is monotone.
- Amongst these, $\exists$ a monotone subsequence of size $n_2 = 2^{2^{(r-2)}} + 1$ in $t_2$.
- Proceeding inductively, we get $n_T = 2^{2^{(r-T)}} + 1 \geq 3$ items, such that in each $t_i$ they appear in increasing or decreasing order.
- Feasibility condition not satisfied $\implies T > r = \log_2 \log_2(n-1)$.