Communication Complexity

Lecture 24

Computing with remote inputs

Communication Complexity

Communication Complexity

- Setting

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)
  - Alice is given only x. Her friend Bob gets y.

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)
  - Alice is given only x. Her friend Bob gets y.
  - Least amount of communication to achieve this

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)
  - Alice is given only x. Her friend Bob gets y.
  - Least amount of communication to achieve this
- Compare with decision tree complexity

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)
  - Alice is given only x. Her friend Bob gets y.
  - Least amount of communication to achieve this
- Compare with decision tree complexity
  - Trivial upper-bound of |x|

Communication Complexity

- Setting
  - Alice wants to compute f(x,y)
  - Alice is given only x. Her friend Bob gets y.
  - Least amount of communication to achieve this
- Compare with decision tree complexity
  - Trivial upper-bound of |x|
- Interested in proving lower bounds for various f

Examples

Examples

- PARITY(x,y) = $\bigoplus_i (x_i \oplus y_i)$

Examples

- PARITY(x,y) = $\bigoplus_i (x_i \oplus y_i)$
  - CC(PARITY) = 1

Examples

- PARITY(x,y) = $\bigoplus_i (x_i \oplus y_i)$
  - CC(PARITY) = 1
- EQ(x,y) = 1 iff x=y

Examples

- PARITY(x,y) = $\bigoplus_i (x_i \oplus y_i)$
  - CC(PARITY) = 1
- EQ(x,y) = 1 iff x=y
  - Lower-bound?

Examples

- PARITY(x,y) = $\bigoplus_i (x_i \oplus y_i)$
  - CC(PARITY) = 1
- EQ(x,y) = 1 iff x=y
  - Lower-bound?
- DISJ(x,y)=1 iff $x \wedge y = 0^n$

Motivation

Motivation

- Distributed computing

Motivation

- Distributed computing
- Lower-bounds for parallel computation/circuit complexity

Motivation

- Distributed computing
- Lower-bounds for parallel computation/circuit complexity
  - Amount of communication across a cut in the circuit

Motivation

- Distributed computing
- Lower-bounds for parallel computation/circuit complexity
  - Amount of communication across a cut in the circuit
  - Variants tightly related to circuit complexity

Motivation

- Distributed computing
- Lower-bounds for parallel computation/circuit complexity
  - Amount of communication across a cut in the circuit
  - Variants tightly related to circuit complexity
- Proving optimality of algorithms and data-structures

Protocol

Protocol

- We'll consider deterministic protocols

Protocol

- We'll consider deterministic protocols
- Fixed number of rounds (Alice to Bob, then Bob to Alice), each party sends a fixed number of bits in each round

Protocol

- We'll consider deterministic protocols
- Fixed number of rounds (Alice to Bob, then Bob to Alice), each party sends a fixed number of bits in each round
  - Can even consider protocol to have Alice and Bob alternately exchanging single bits (since not considering number of rounds)

Protocol

- We'll consider deterministic protocols
- Fixed number of rounds (Alice to Bob, then Bob to Alice), each party sends a fixed number of bits in each round
  - Can even consider protocol to have Alice and Bob alternately exchanging single bits (since not considering number of rounds)
    - At most doubles the communication complexity

Protocol Execution

Protocol Execution

- i^{th} message from Alice is a function of her input and previous messages

Protocol Execution

- i^{th} message from Alice is a function of her input and previous messages
- Her output is a function of the final "transcript" and her own input (her "view")

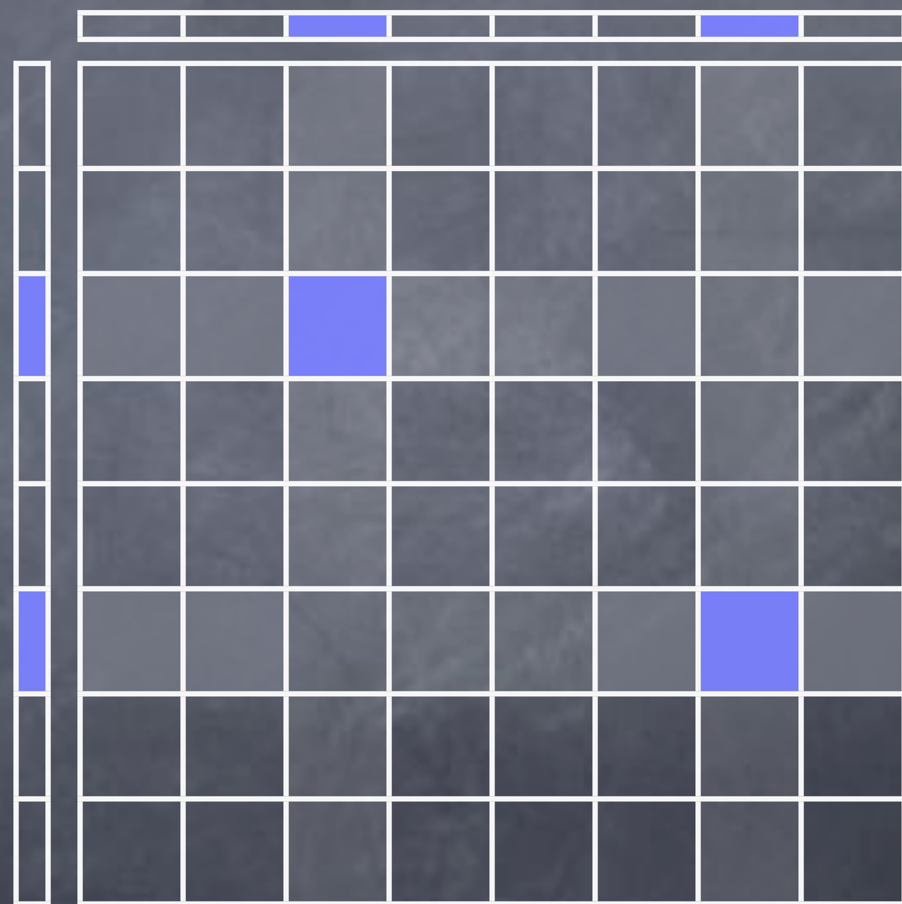
Protocol Execution

- i^{th} message from Alice is a function of her input and previous messages
- Her output is a function of the final "transcript" and her own input (her "view")
    - Similarly for Bob. His view = transcript + his input

Protocol Execution

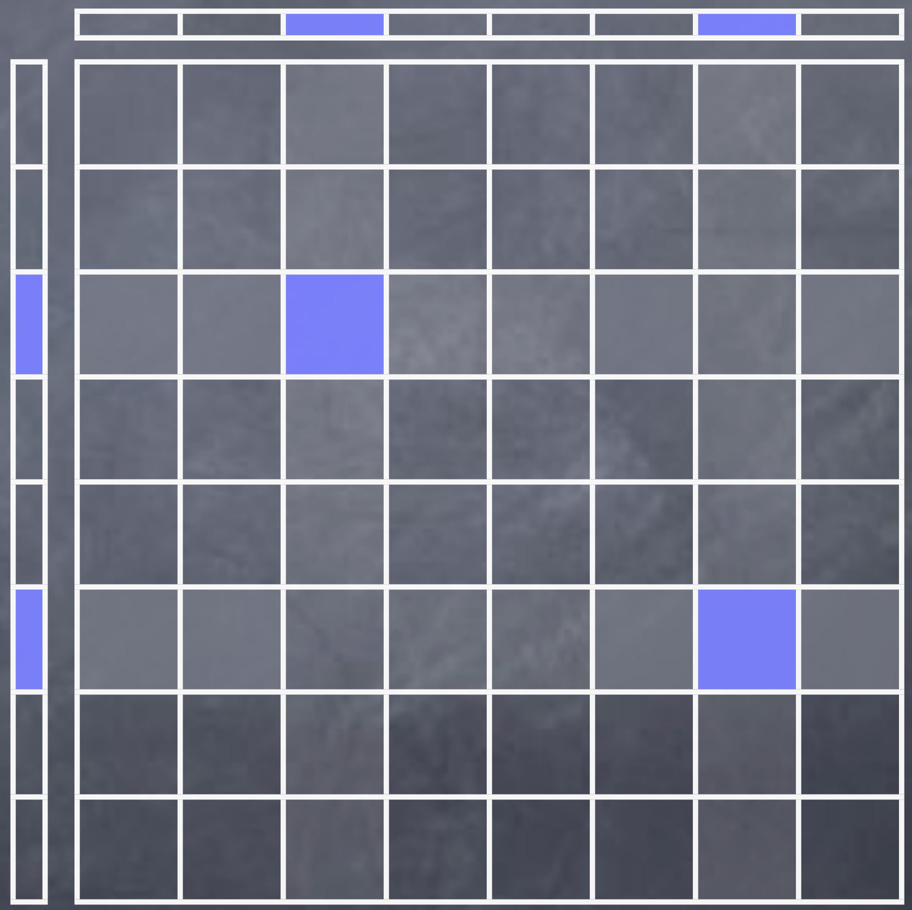
- i^{th} message from Alice is a function of her input and previous messages
- Her output is a function of the final "transcript" and her own input (her "view")
  - Similarly for Bob. His view = transcript + his input
- #transcripts $\leq 2^{CC}$. i.e. $CC \geq \log(\#transcripts)$

Transcript Table



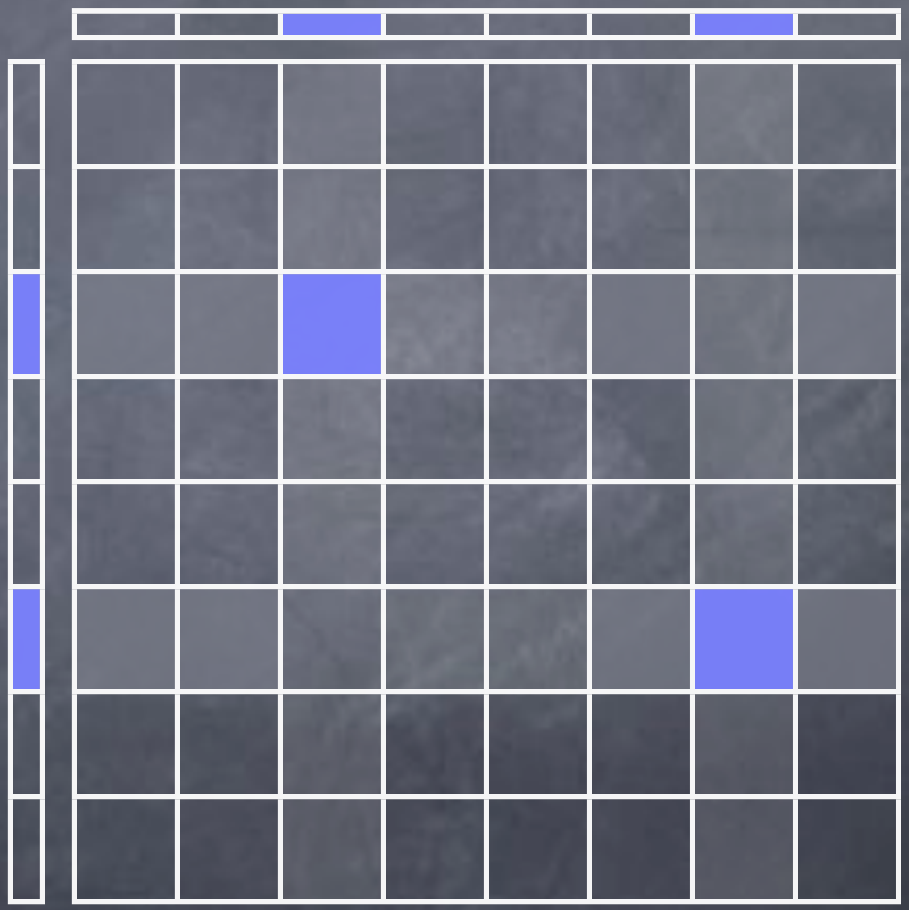
Transcript Table

- Consider the transcript table



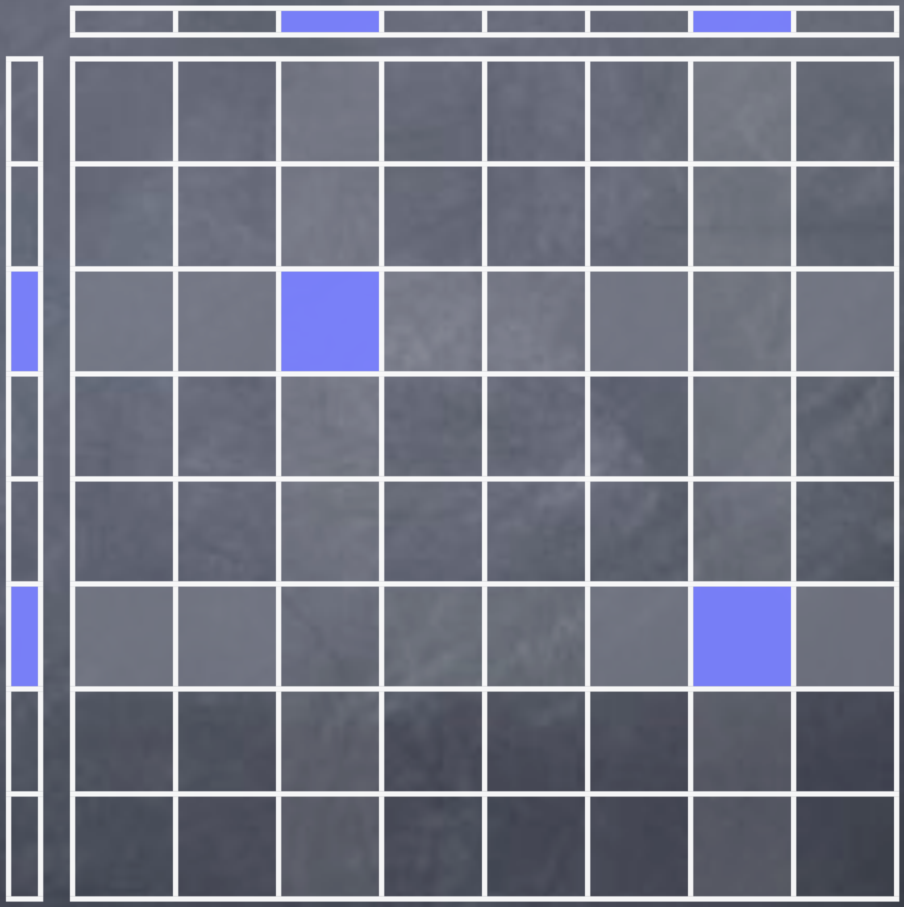
Transcript Table

- Consider the transcript table
  - If on (a_1,b_1) and (a_2,b_2) same transcript



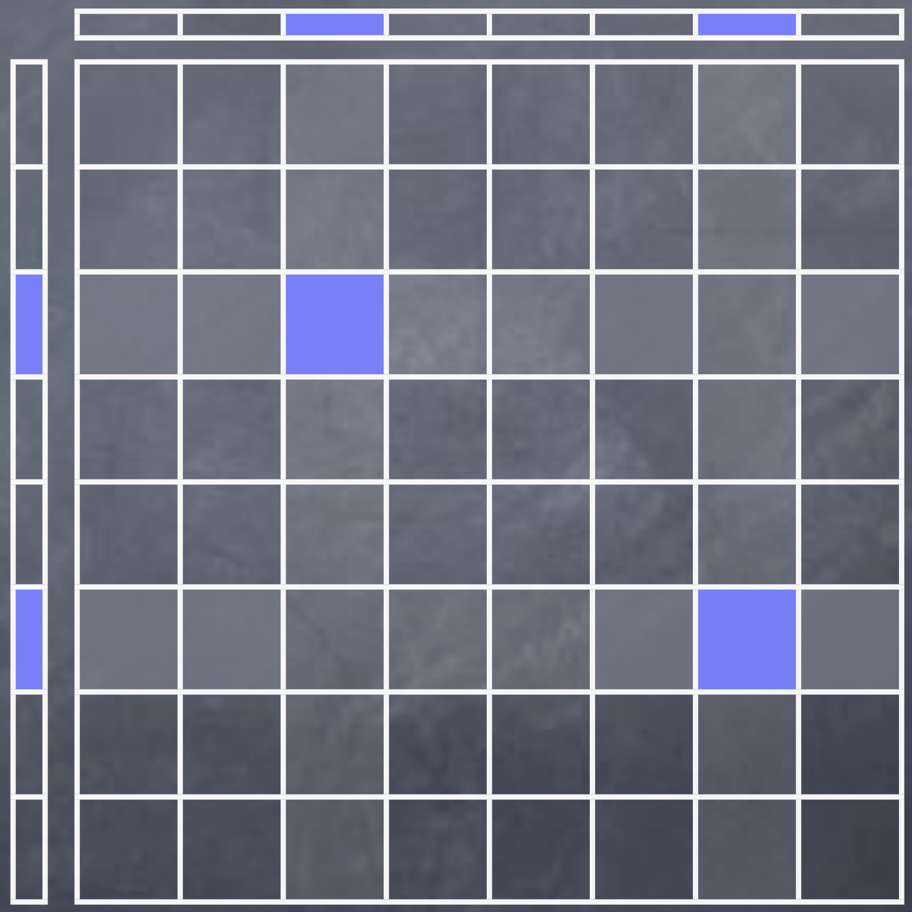
Transcript Table

- Consider the transcript table
    - If on (a_1,b_1) and (a_2,b_2) same transcript
        - Then same transcript on (a_1,b_2) also!

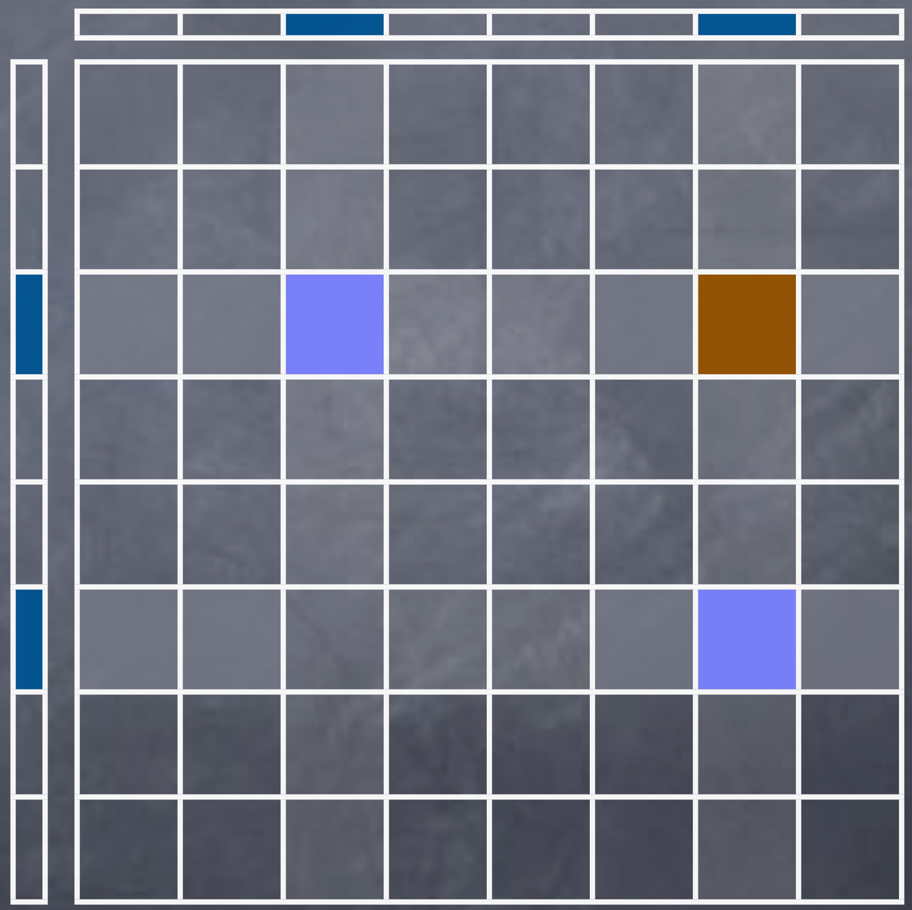


Transcript Table

- Consider the transcript table
  - If on (a_1,b_1) and (a_2,b_2) same transcript
    - Then same transcript on (a_1,b_2) also!
    - Alice and Bob never realize the difference through out the protocol

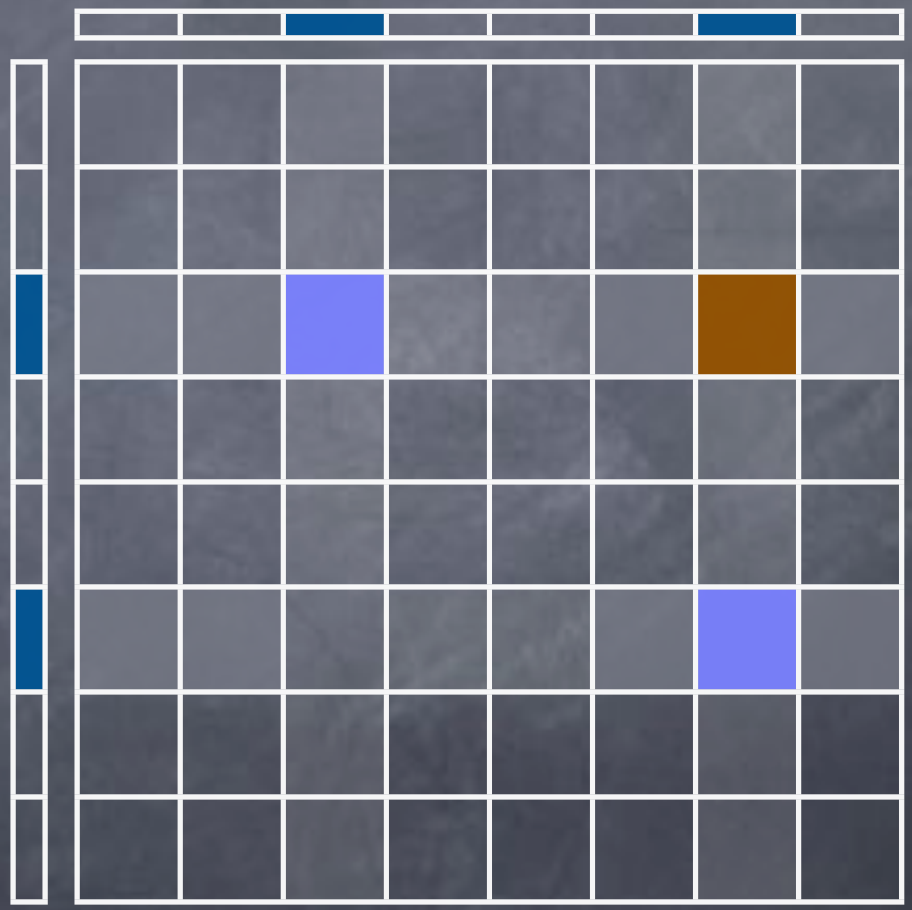


Fooling Set



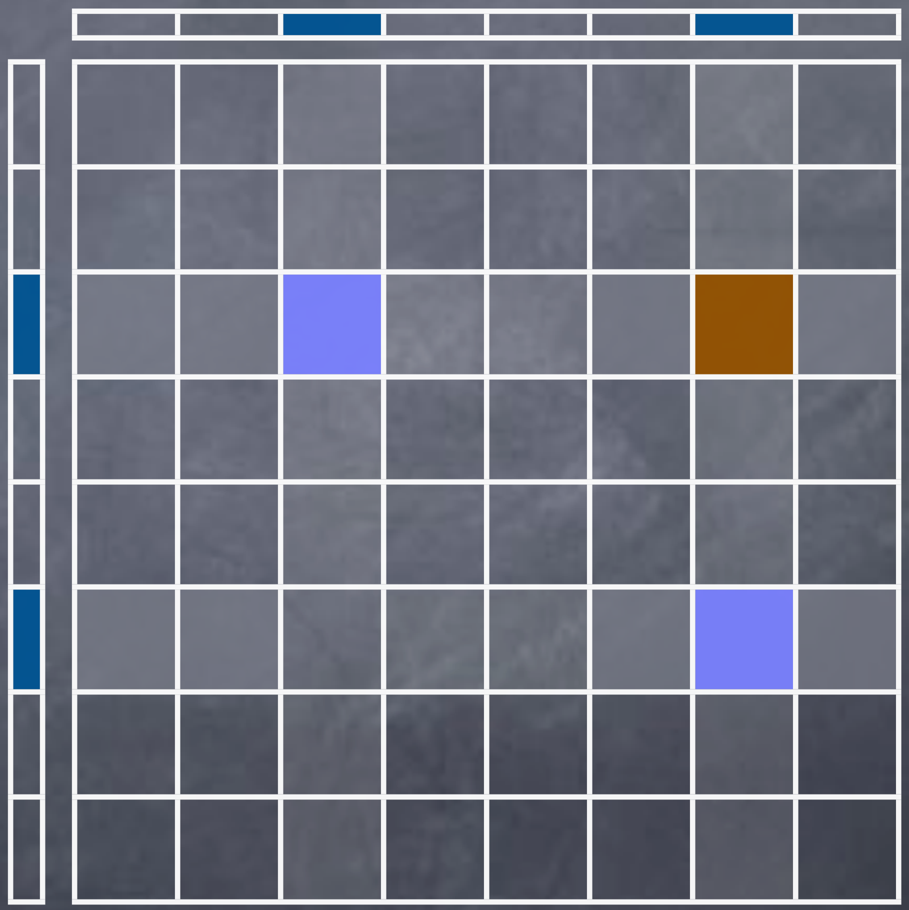
Fooling Set

- If on (a_1,b_1) and (a_2,b_2) same transcript, then same transcript on (a_1,b_2) also



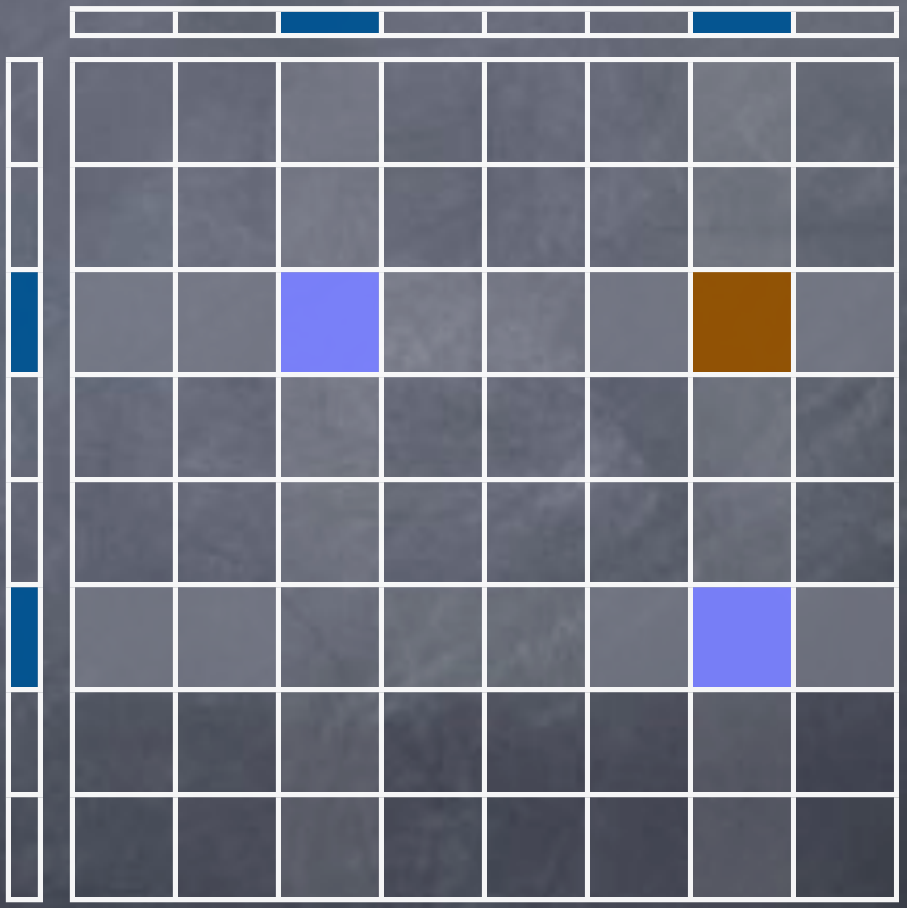
Fooling Set

- If on (a_1,b_1) and (a_2,b_2) same transcript, then same transcript on (a_1,b_2) also
- Show a set S of input-pairs that must have distinct transcripts



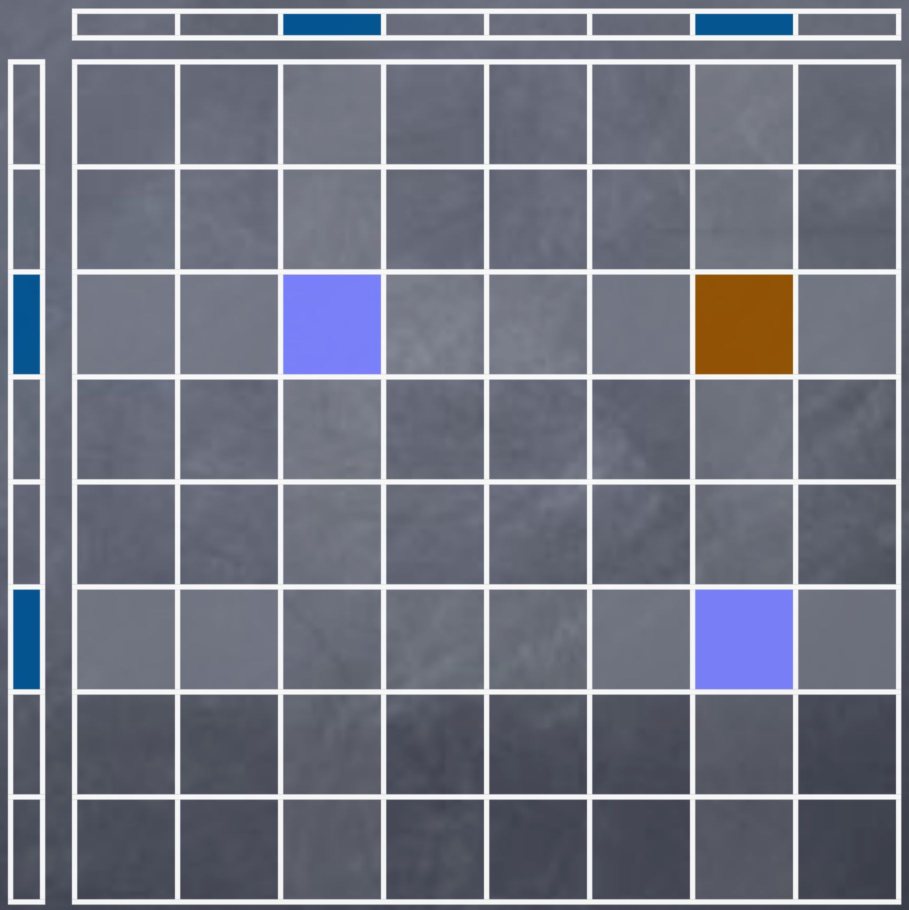
Fooling Set

- If on (a_1,b_1) and (a_2,b_2) same transcript, then same transcript on (a_1,b_2) also
- Show a set S of input-pairs that must have distinct transcripts
  - Even though all pairs have same output



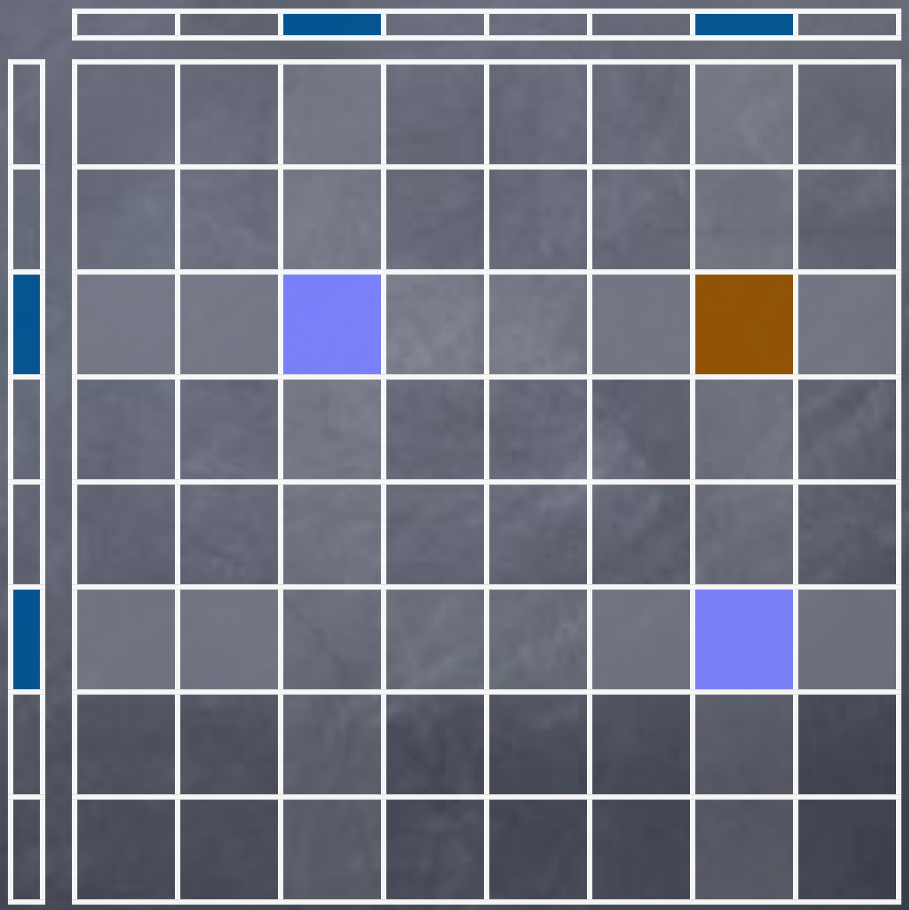
Fooling Set

- If on (a_1,b_1) and (a_2,b_2) same transcript, then same transcript on (a_1,b_2) also
- Show a set S of input-pairs that must have distinct transcripts
  - Even though all pairs have same output
  - "Cross" of no two pairs has the same output

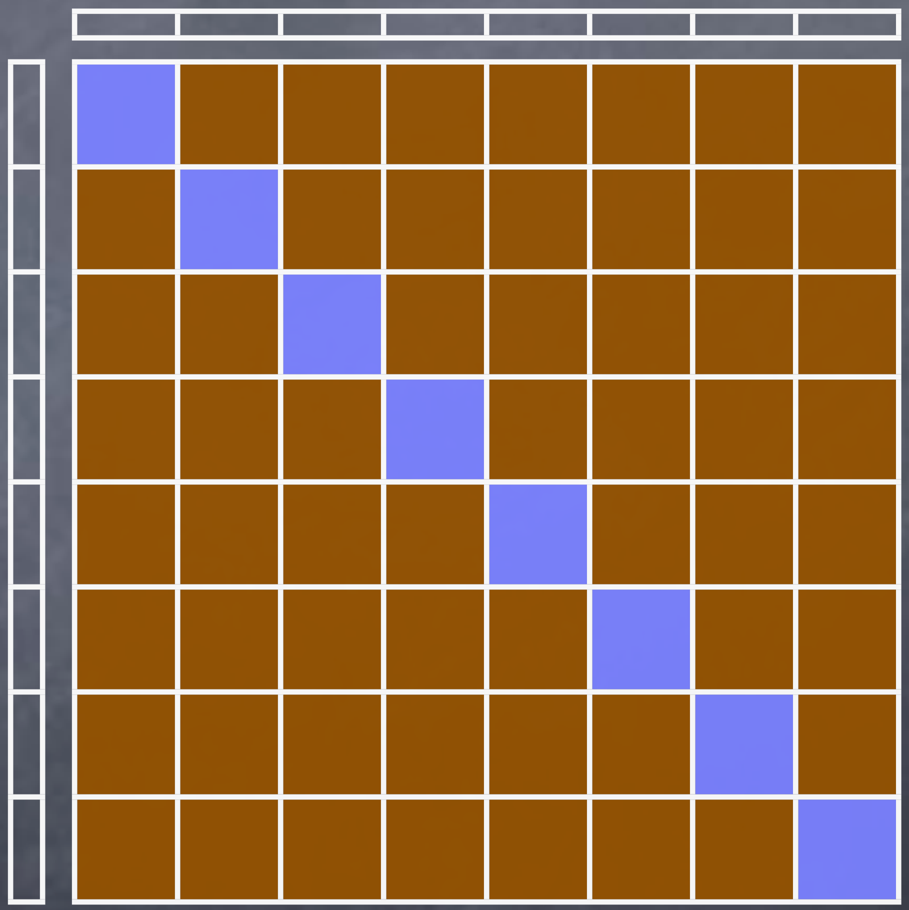


Fooling Set

- If on (a_1,b_1) and (a_2,b_2) same transcript, then same transcript on (a_1,b_2) also
- Show a set S of input-pairs that must have distinct transcripts
  - Even though all pairs have same output
  - "Cross" of no two pairs has the same output
- If S is a set of such pairs, CC ≥ log(|S|)

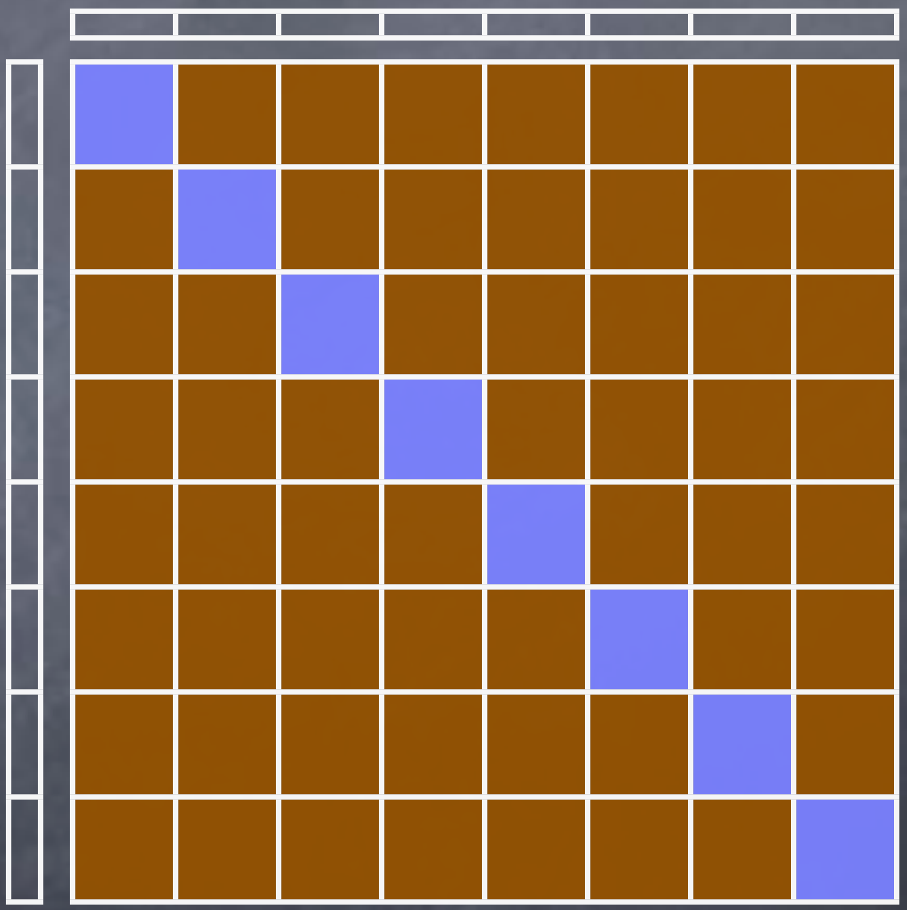


Fooling Set for EQ



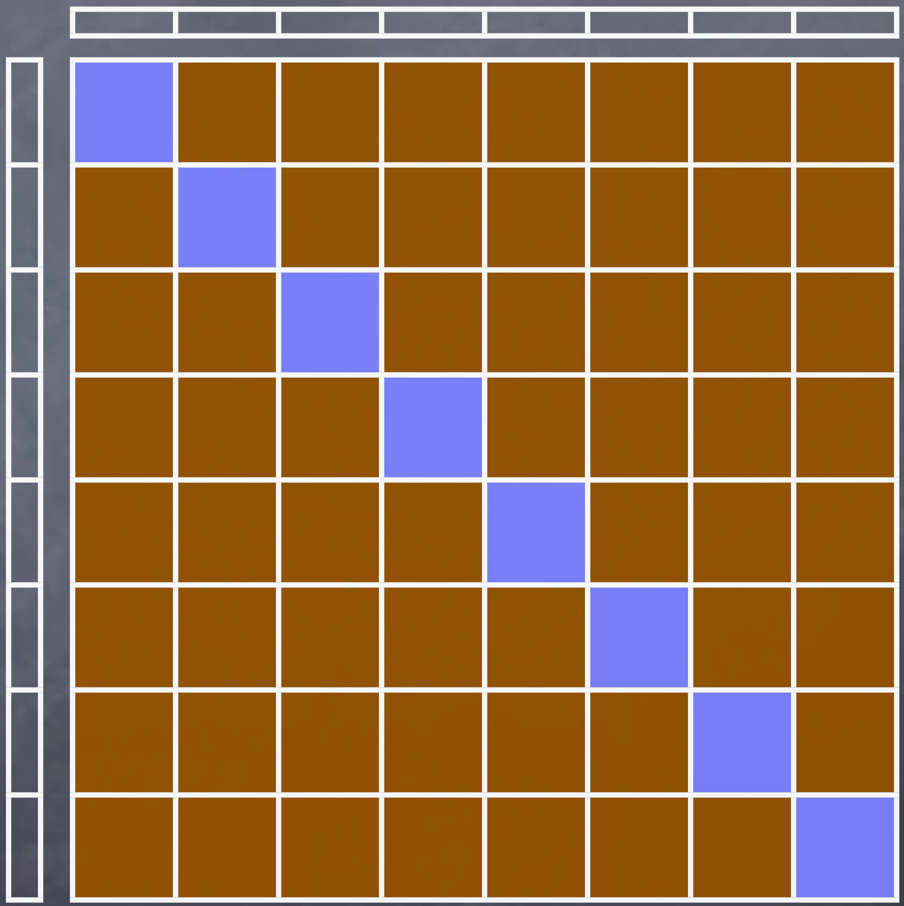
Fooling Set for EQ

- S = set of all pairs (x,x)



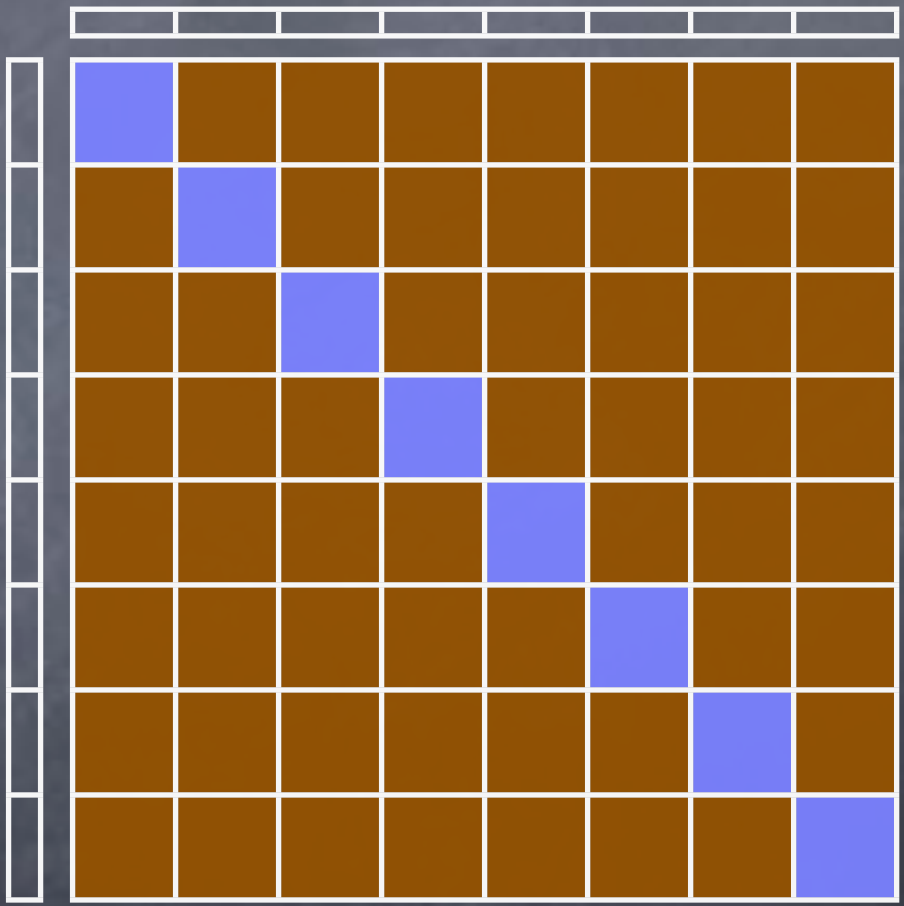
Fooling Set for EQ

- S = set of all pairs (x,x)
  - CC(EQ) ≥ log(|S|) ≥ n



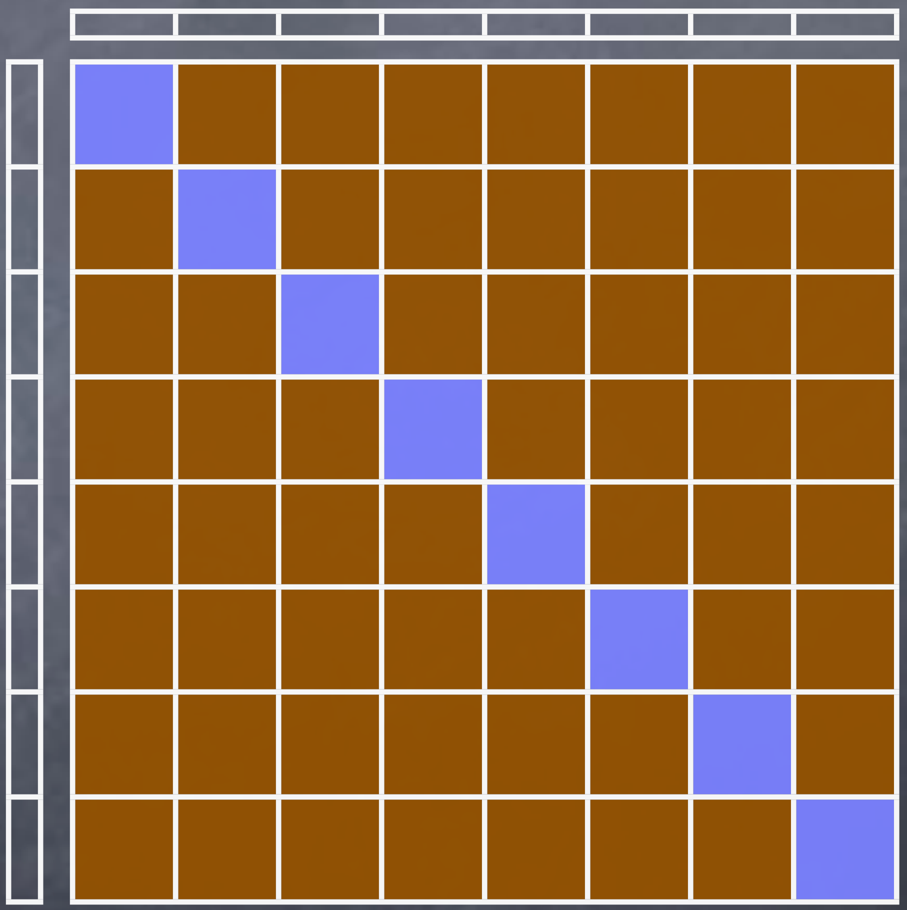
Fooling Set for EQ

- S = set of all pairs (x,x)
  - CC(EQ) ≥ log(|S|) ≥ n
- True for any function in which each row and column has exactly one 1



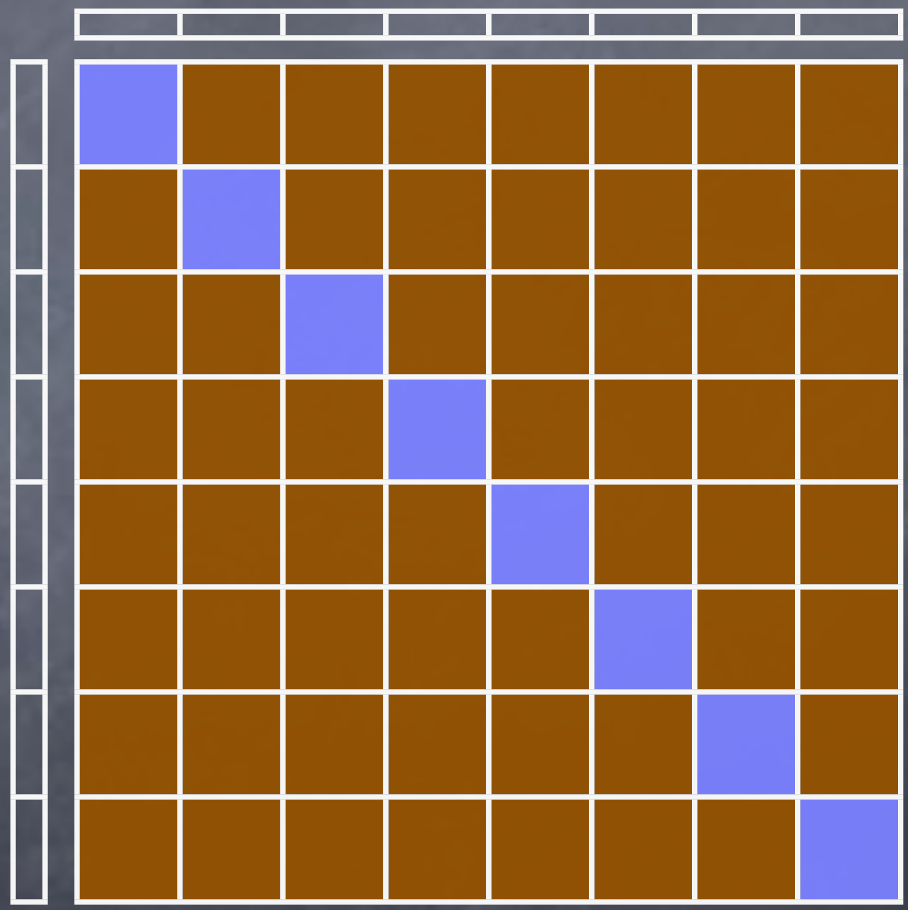
Fooling Set for EQ

- S = set of all pairs (x,x)
  - CC(EQ) ≥ log(|S|) ≥ n
- True for any function in which each row and column has exactly one 1
- Other functions too



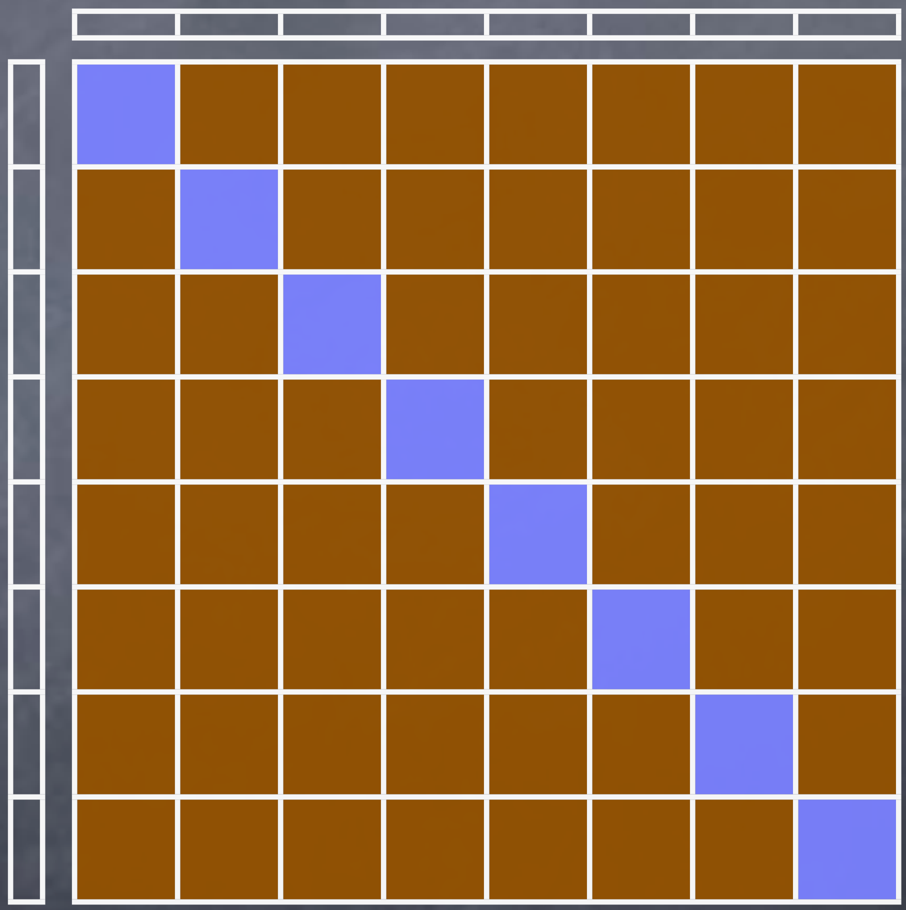
Fooling Set for EQ

- S = set of all pairs (x,x)
  - CC(EQ) ≥ log(|S|) ≥ n
- True for any function in which each row and column has exactly one 1
- Other functions too
  - e.g.: DISJ(x,y) if $x \wedge y = 0^n$

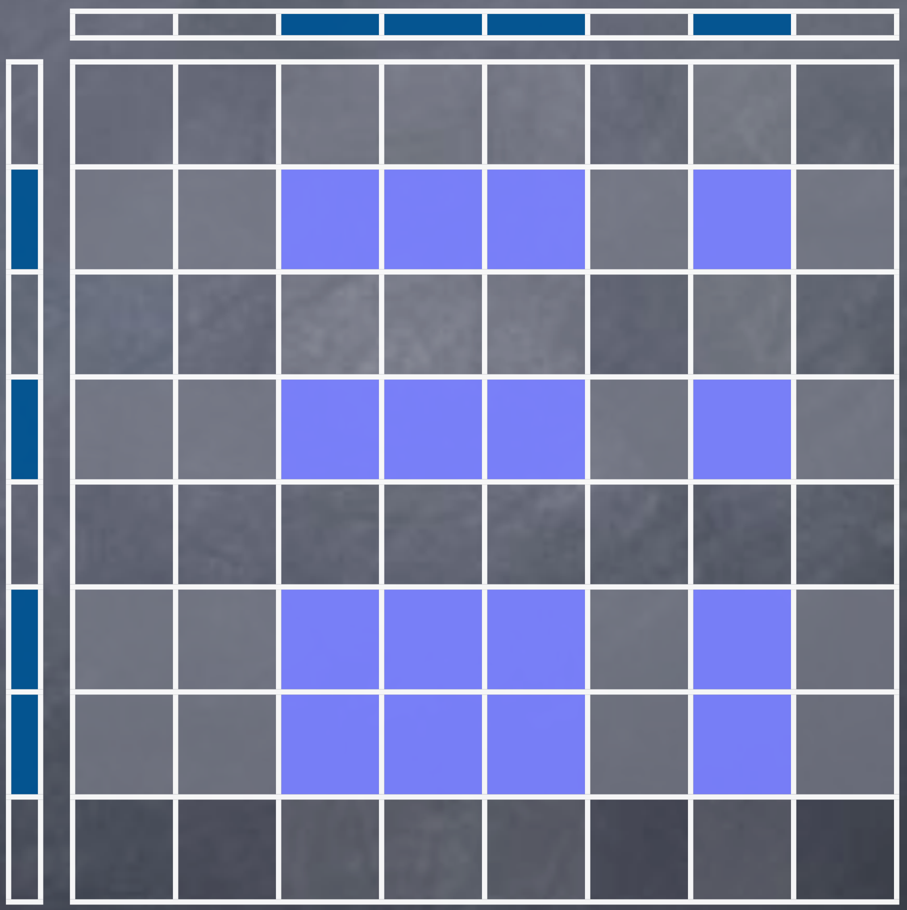


Fooling Set for EQ

- S = set of all pairs (x,x)
  - CC(EQ) ≥ log(|S|) ≥ n
- True for any function in which each row and column has exactly one 1
- Other functions too
  - e.g.: DISJ(x,y) if $x \wedge y = 0^n$
  - S = set of complementary pairs, $(x, \neg x)$

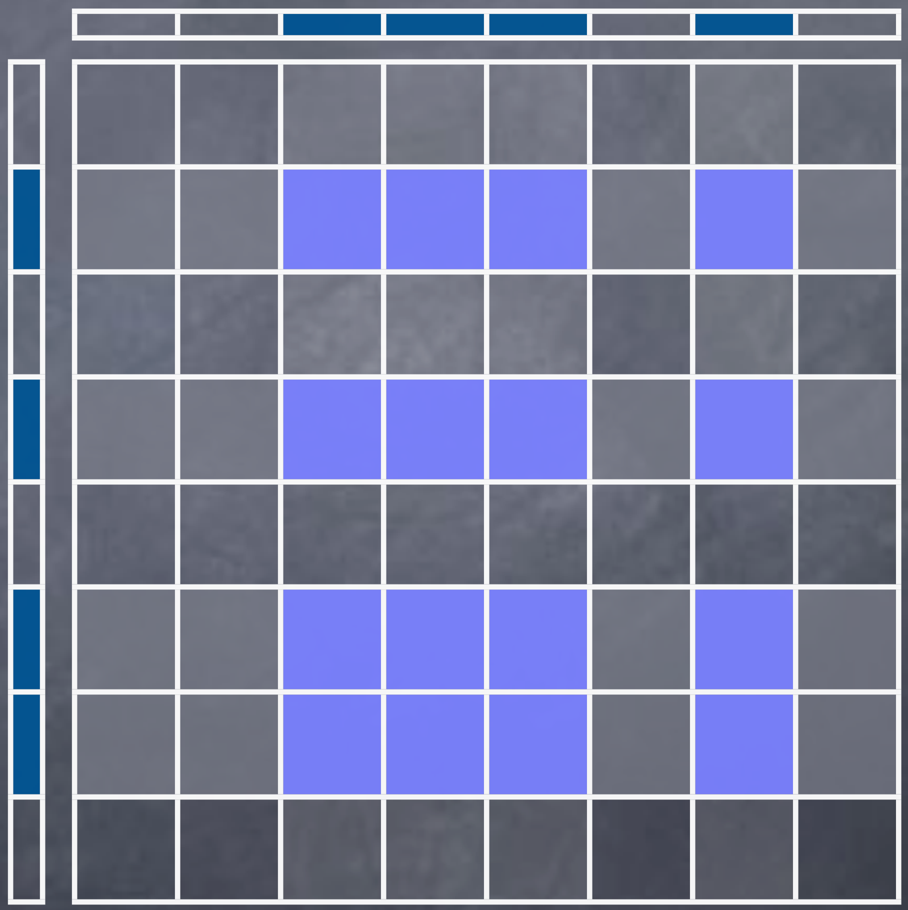


Monochromatic Rectangles



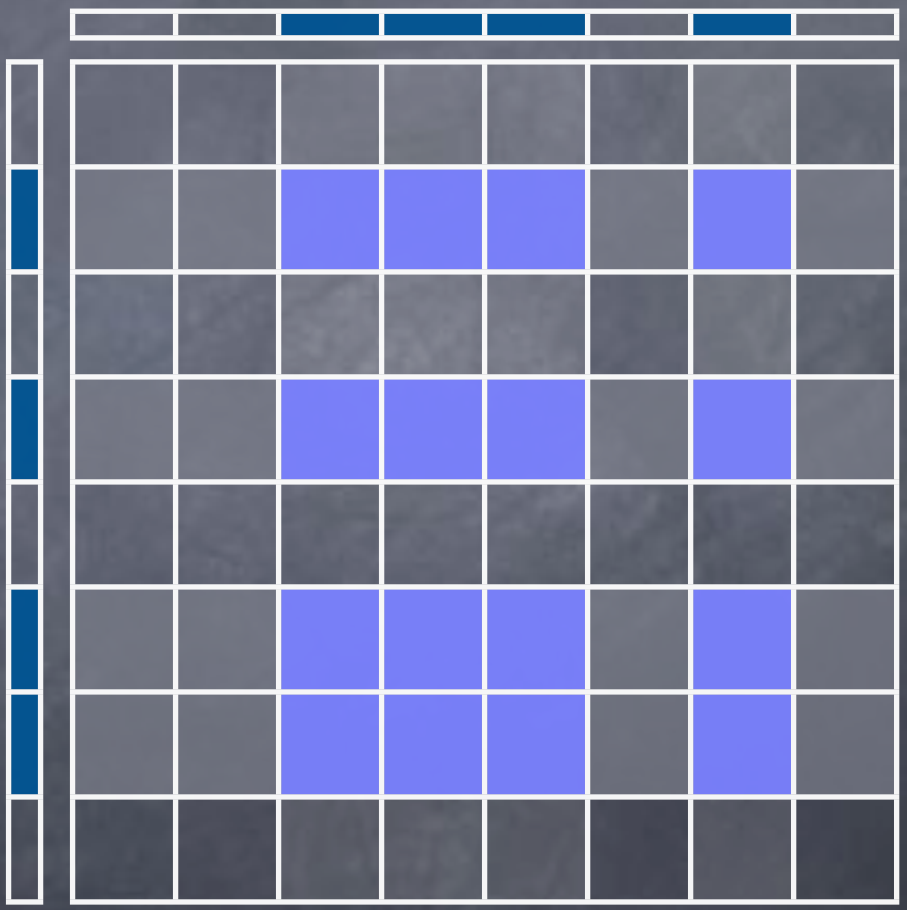
Monochromatic Rectangles

- Rectangle: a subset of $D_1 \times D_2$ of the form $S_1 \times S_2$



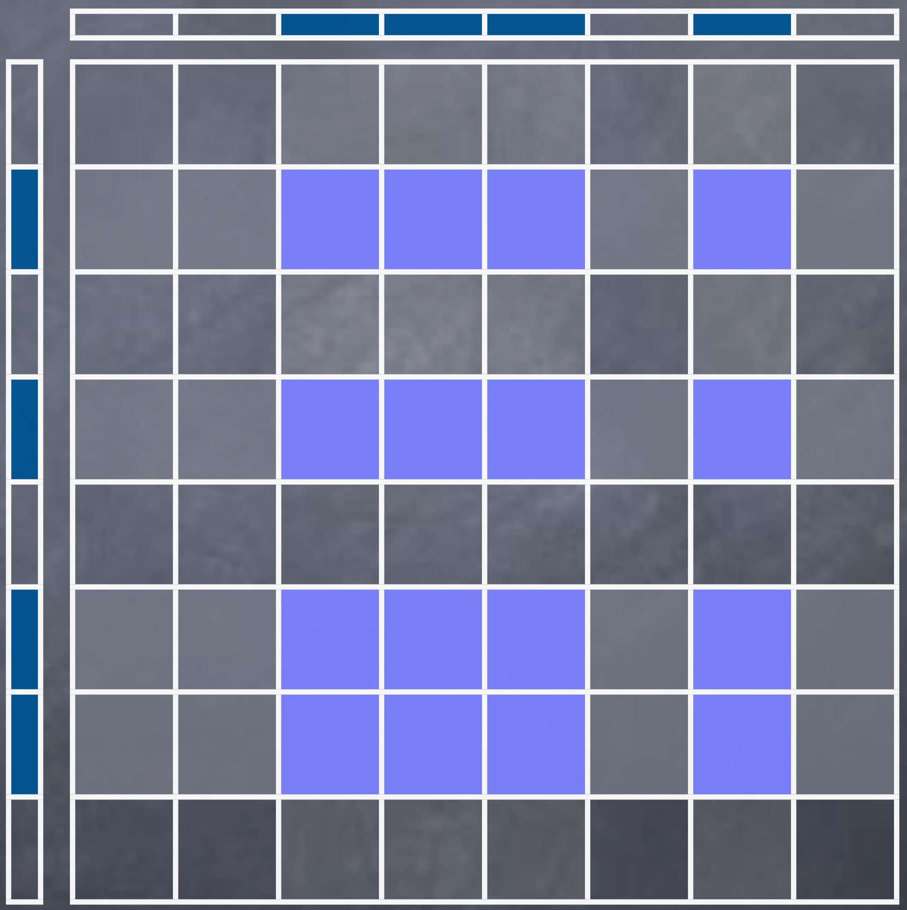
Monochromatic Rectangles

- Rectangle: a subset of $D_1 \times D_2$ of the form $S_1 \times S_2$
- Monochromatic: same f-value



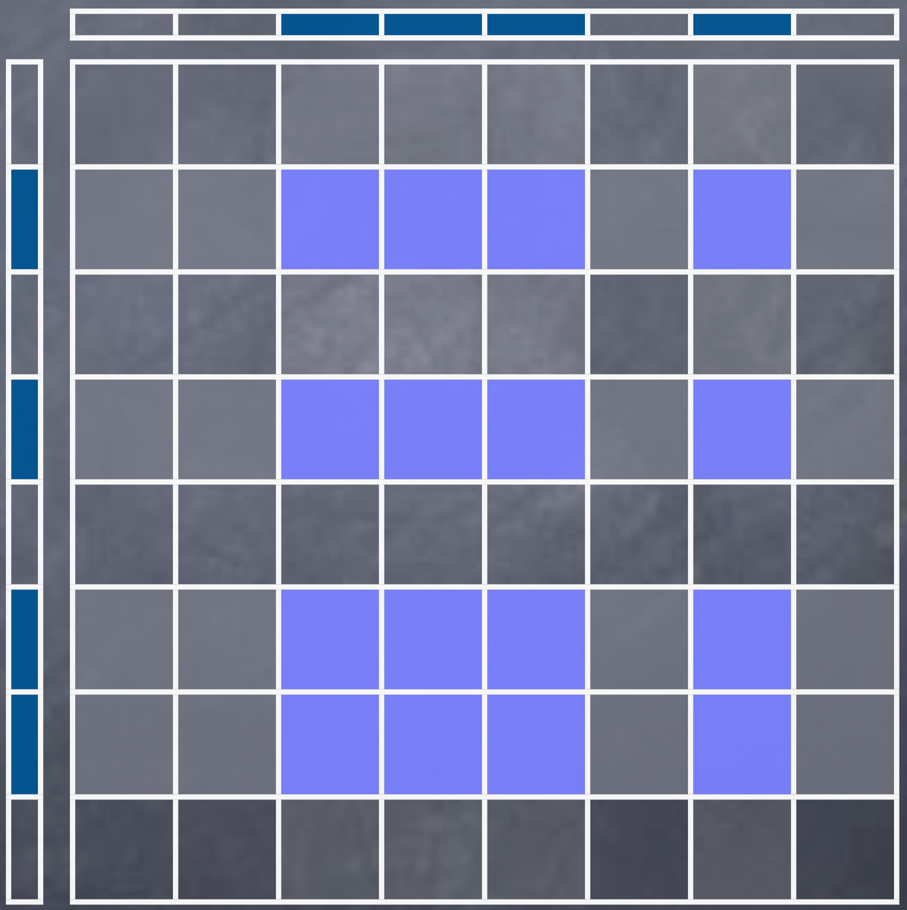
Monochromatic Rectangles

- Rectangle: a subset of $D_1 \times D_2$ of the form $S_1 \times S_2$
- Monochromatic: same f-value
- Recall: for any protocol, set of all input-pairs with the same transcript is a rectangle

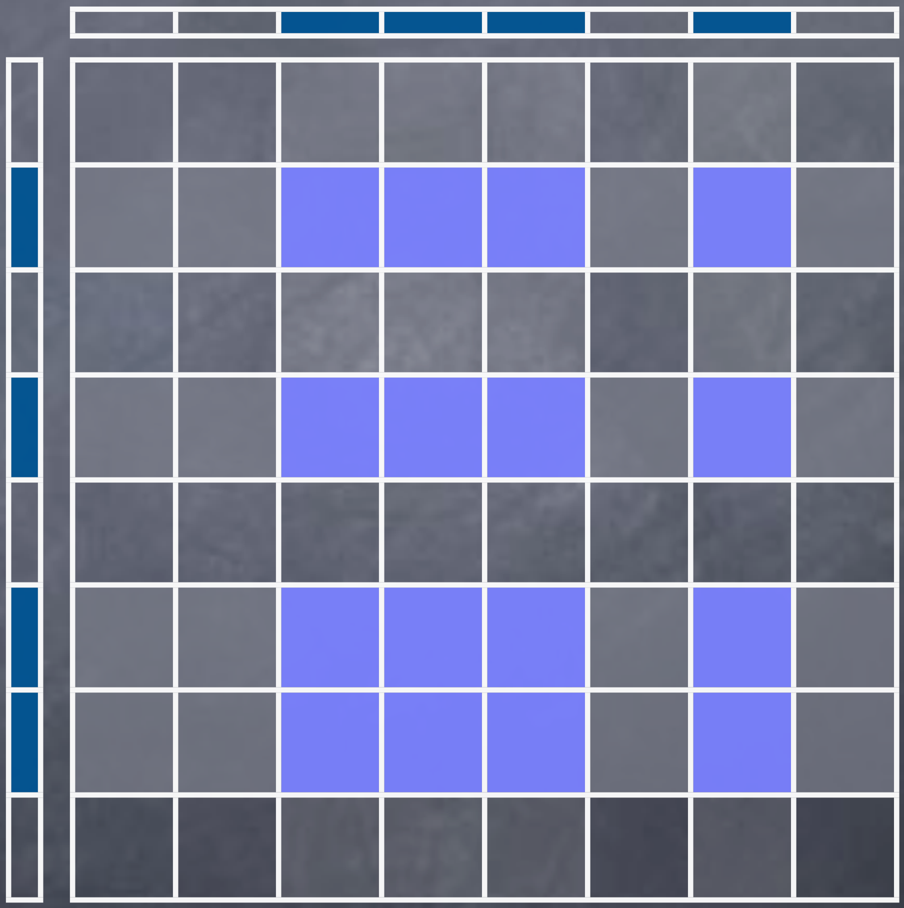


Monochromatic Rectangles

- Rectangle: a subset of $D_1 \times D_2$ of the form $S_1 \times S_2$
- Monochromatic: same f-value
- Recall: for any protocol, set of all input-pairs with the same transcript is a rectangle
- For protocol to be correct, the rectangles should be monochromatic

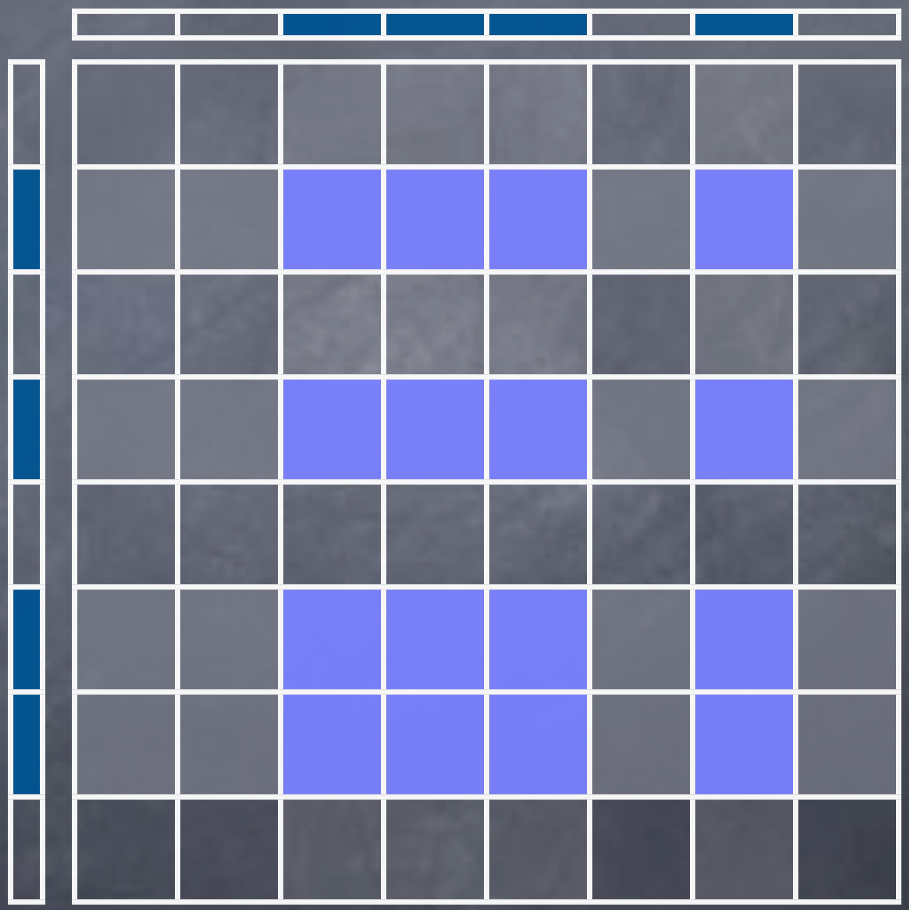


Tiling Lower-Bound



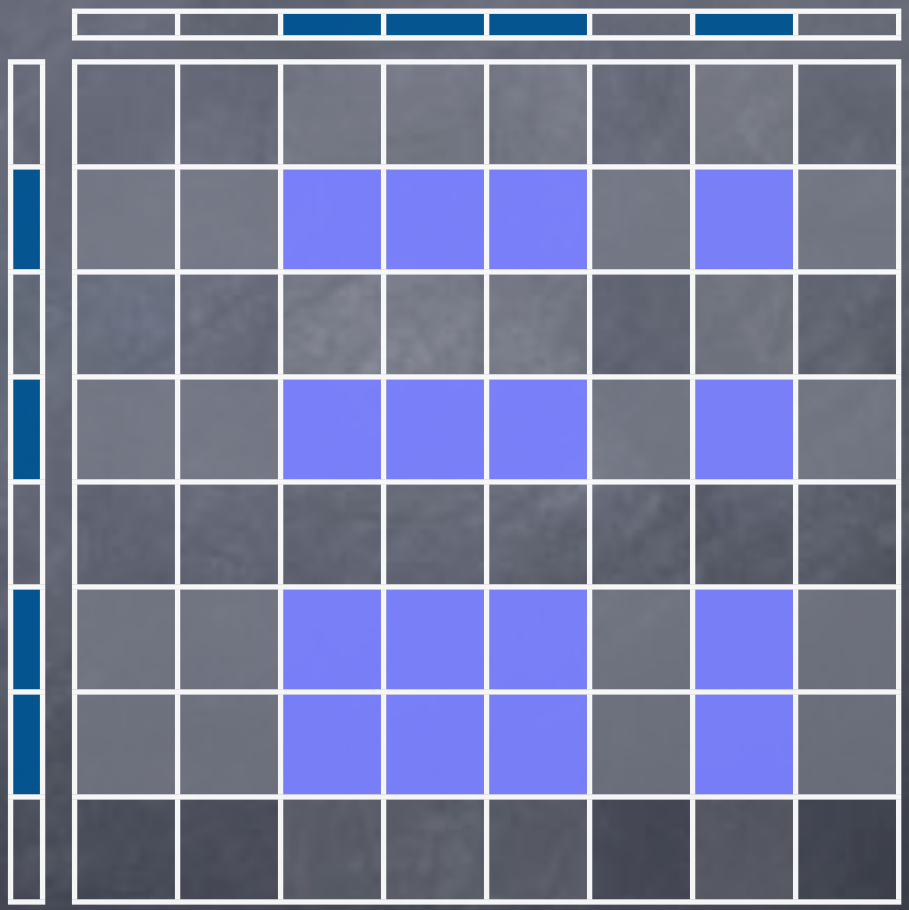
Tiling Lower-Bound

- For protocol to be correct, same-transcript rectangles should be monochromatic



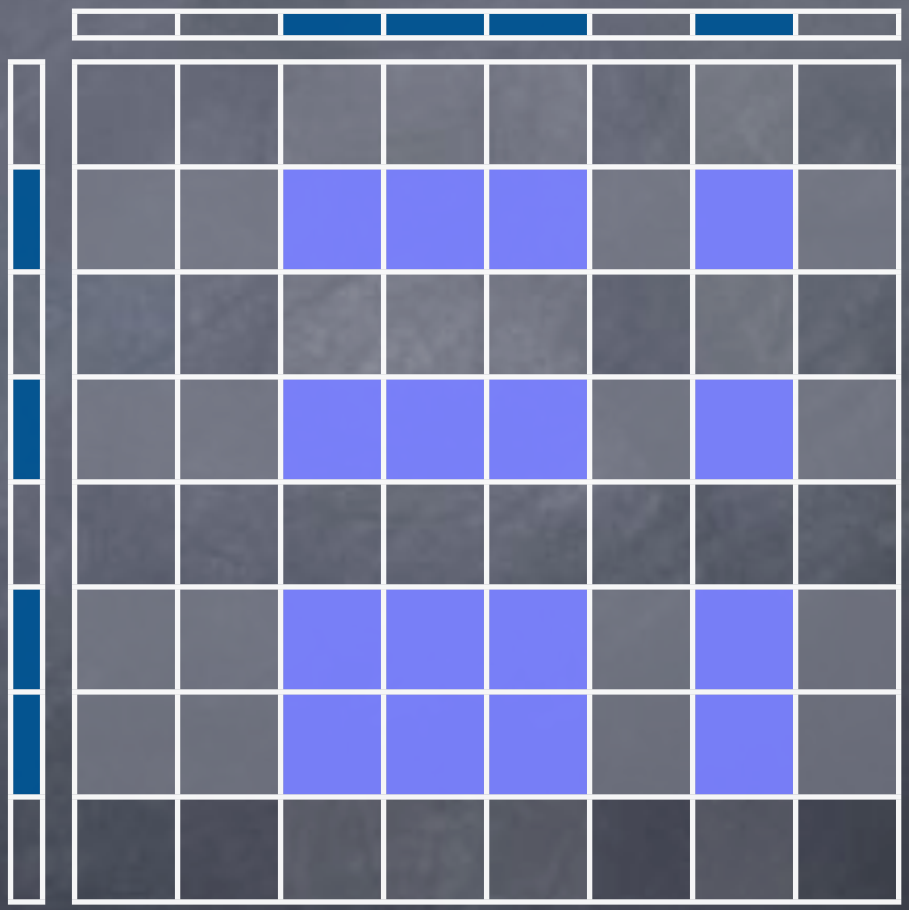
Tiling Lower-Bound

- For protocol to be correct, same-transcript rectangles should be monochromatic
- Find the least number of monochromatic rectangles that can tile the function, χ(f)



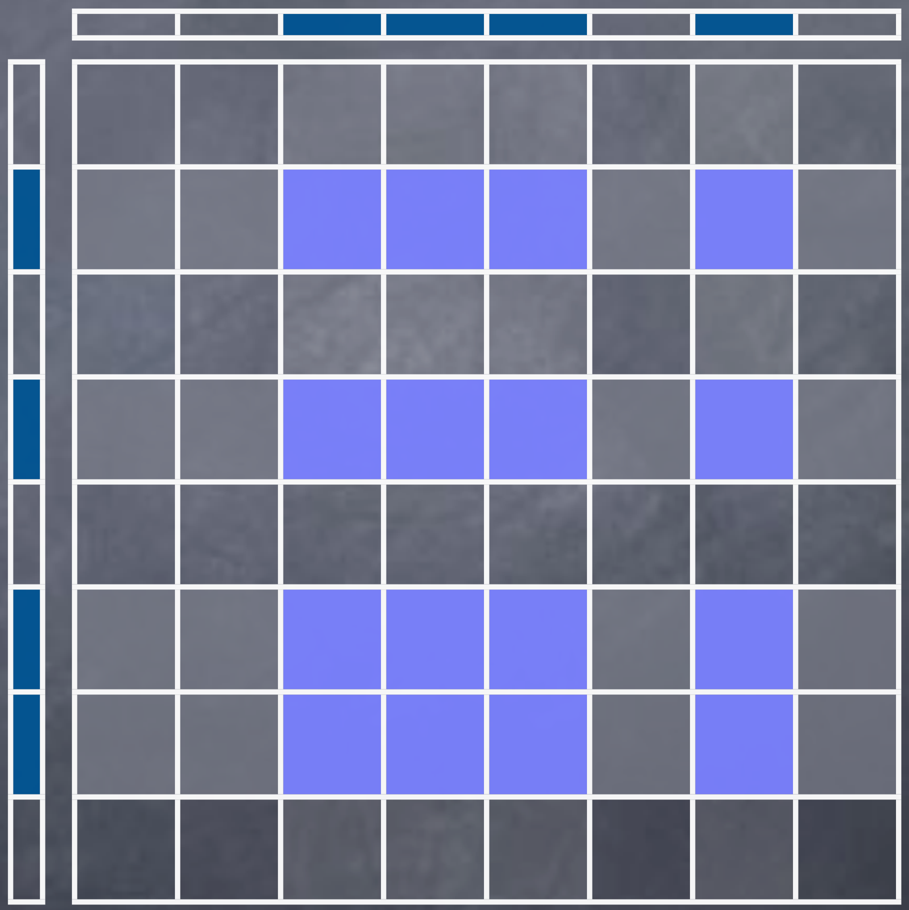
Tiling Lower-Bound

- For protocol to be correct, same-transcript rectangles should be monochromatic
- Find the least number of monochromatic rectangles that can tile the function, $\chi(f)$
  - #transcripts $\geq \chi(f)$



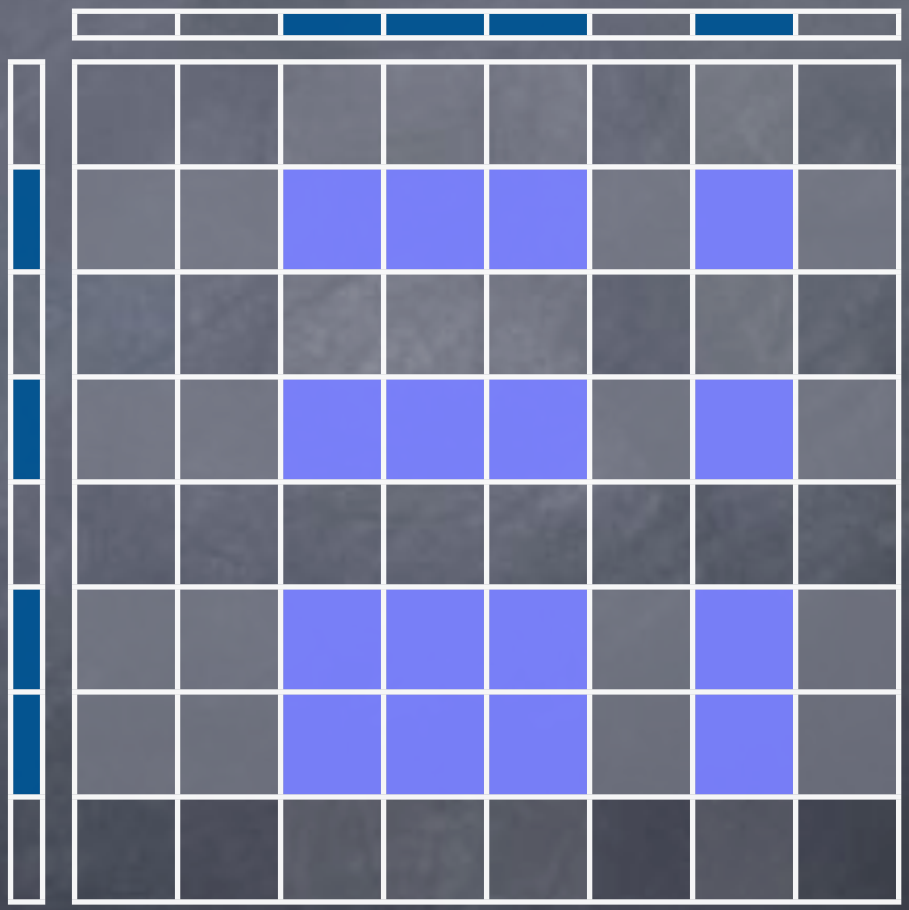
Tiling Lower-Bound

- For protocol to be correct, same-transcript rectangles should be monochromatic
- Find the least number of monochromatic rectangles that can tile the function, $\chi(f)$
  - #transcripts $\geq \chi(f)$
  - $CC(f) \geq \log(\chi(f))$

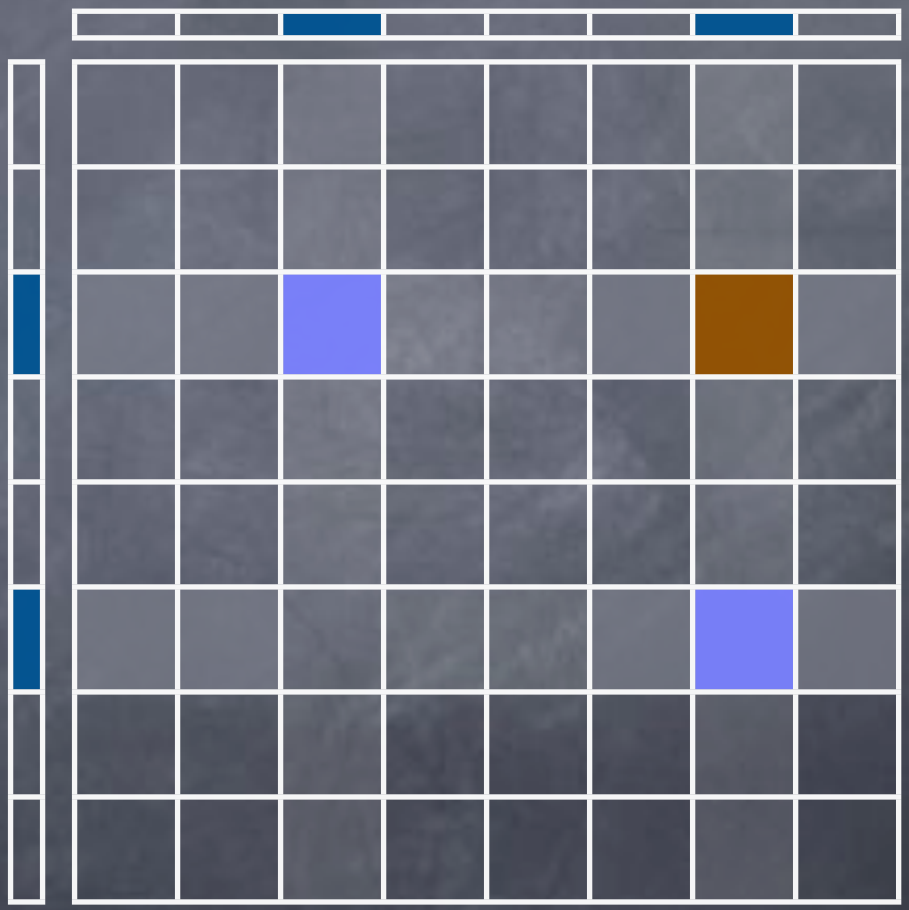


Tiling Lower-Bound

- For protocol to be correct, same-transcript rectangles should be monochromatic
- Find the least number of monochromatic rectangles that can tile the function, $\chi(f)$
  - #transcripts $\geq \chi(f)$
  - $CC(f) \geq \log(\chi(f))$
- How to lower-bound $\chi(f)$?

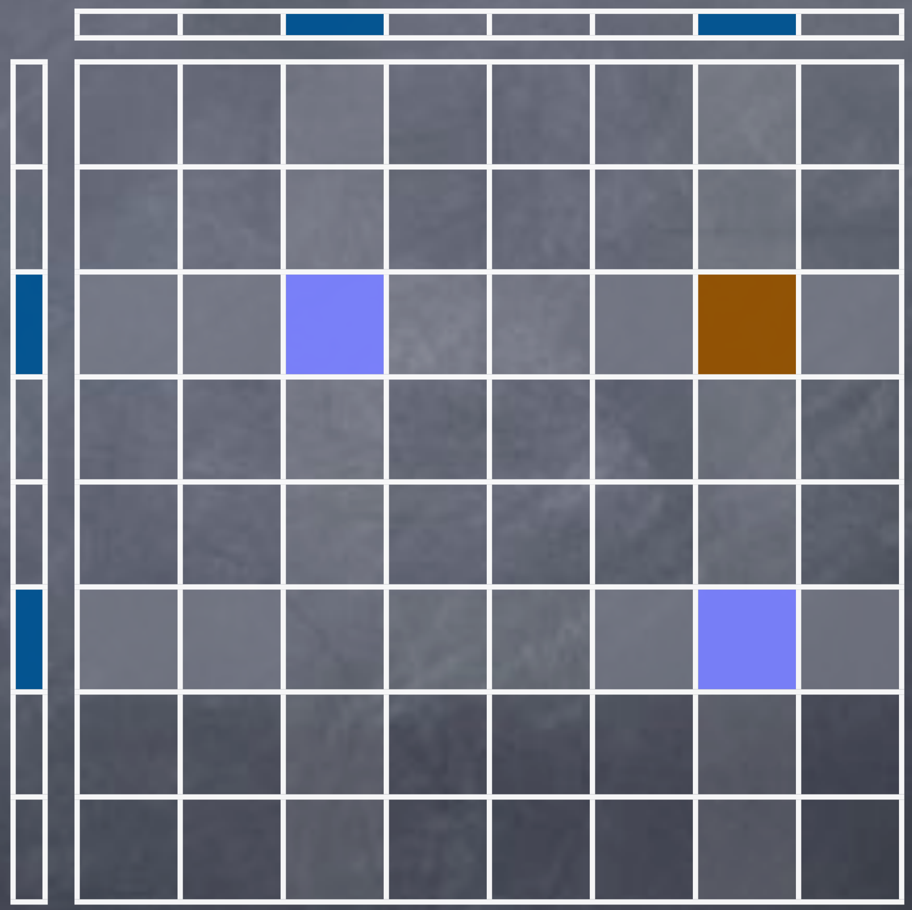


Lower-Bounding $\chi(f)$



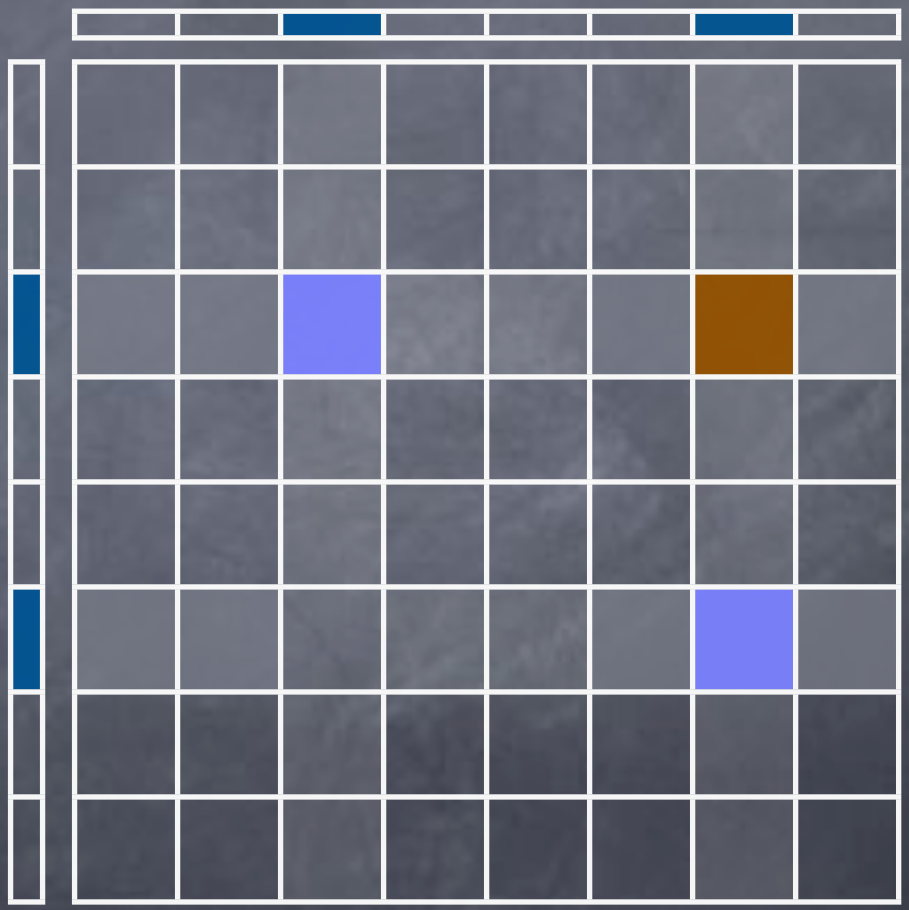
Lower-Bounding χ(f)

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling



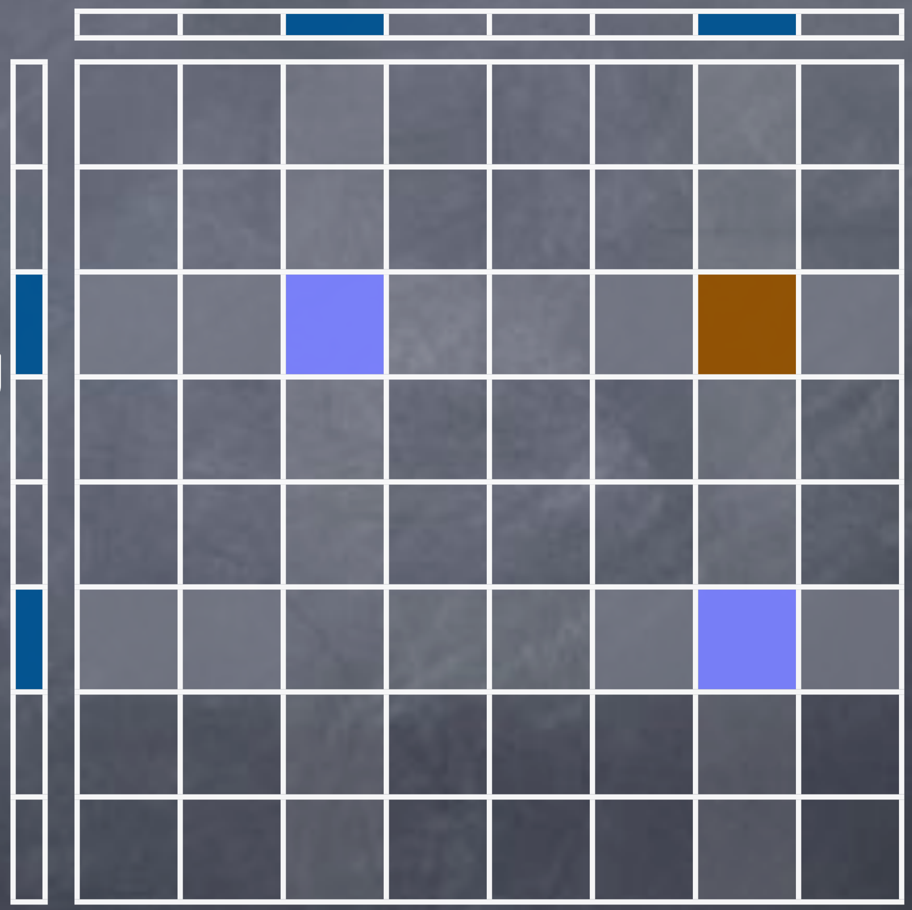
Lower-Bounding χ(f)

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling
  - χ(f) ≥ |S| for every fooling set S



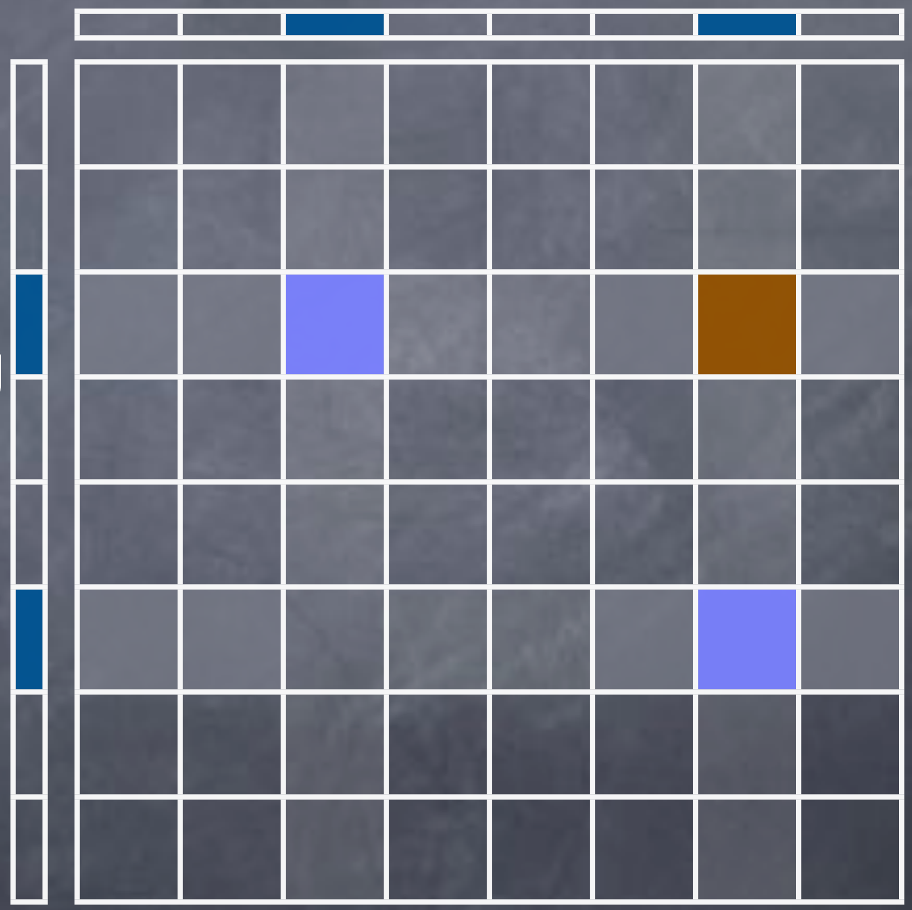
Lower-Bounding $\chi(f)$

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling
  - $\chi(f) \geq |S|$ for every fooling set S
- Rank lower-bound



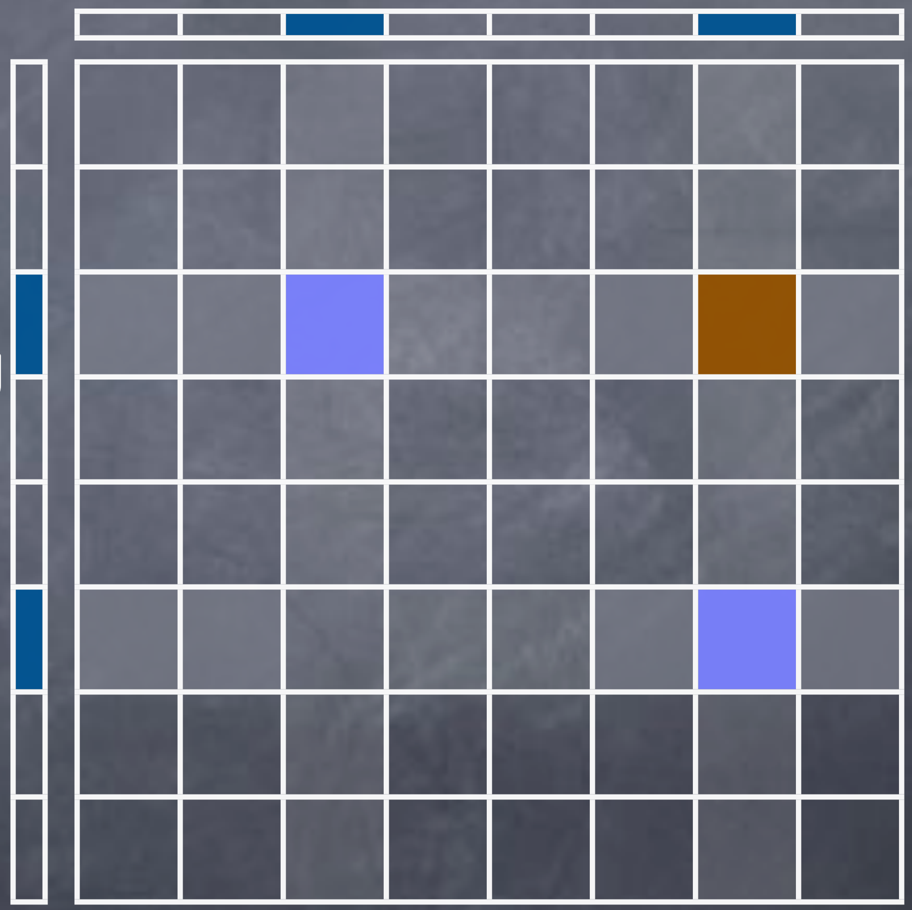
Lower-Bounding χ(f)

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling
  - χ(f) ≥ |S| for every fooling set S
- Rank lower-bound
  - χ(f) ≥ Rank(M_f)



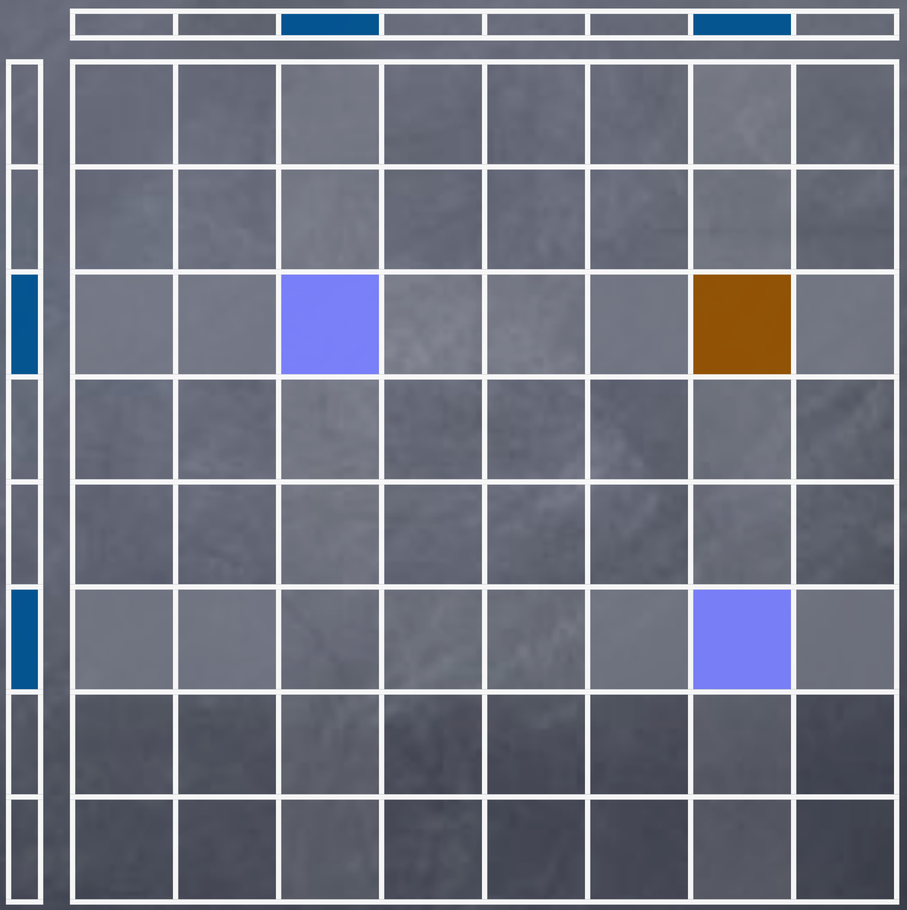
Lower-Bounding $\chi(f)$

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling
  - $\chi(f) \geq |S|$ for every fooling set S
- Rank lower-bound
  - $\chi(f) \geq \text{Rank}(M_f)$
- Discrepancy lower-bound



Lower-Bounding $\chi(f)$

- If a fooling set of size S, no two input-pairs from S can be on the same tile in a monochromatic tiling
  - $\chi(f) \geq |S|$ for every fooling set S
- Rank lower-bound
  - $\chi(f) \geq \text{Rank}(M_f)$
- Discrepancy lower-bound
  - $\chi(f) \geq \text{Discrepancy}(f)$



Rank(M)

Rank(M)

- Rank of a matrix

Rank(M)

- Rank of a matrix
  - Maximum number of linearly independent rows (or equivalently, columns)

Rank(M)

- Rank of a matrix
  - Maximum number of linearly independent rows (or equivalently, columns)
  - Linear independence: operations in a field

Rank(M)

- Rank of a matrix
  - Maximum number of linearly independent rows (or equivalently, columns)
  - Linear independence: operations in a field
- Rank-r matrix: after row & column reductions $D_{(m \times n)}$ diagonal, with r 1's, rest 0's. M = UDV

Rank(M)

- Rank of a matrix
  - Maximum number of linearly independent rows (or equivalently, columns)
  - Linear independence: operations in a field
- Rank-r matrix: after row & column reductions $D_{(m \times n)}$ diagonal, with r 1's, rest 0's. M = UDV
- Rank(M) $\leq$ r, iff M can be written as sum of $\leq$ r rank 1 matrices

Rank(M)

- Rank of a matrix
  - Maximum number of linearly independent rows (or equivalently, columns)
  - Linear independence: operations in a field
- Rank-r matrix: after row & column reductions $D_{(m \times n)}$ diagonal, with r 1's, rest 0's. M = UDV
- Rank(M) $\leq$ r, iff M can be written as sum of $\leq$ r rank 1 matrices
  - $M = UDV = \Sigma_{i \leq r} D_{ii} U_{i(m \times 1)} V_{i(1 \times n)} = \Sigma_{i \leq r} B_i$, where $\text{Rank}(B_i)=1$

$\chi(f) \geq \text{Rank}(M_f)$

$\chi(f) \geq \text{Rank}(M_f)$

- If $M = \Sigma_{i \leq r} B_i$ with $\text{Rank}(B_i)=1$, then $\text{Rank}(M) \leq r$

$\chi(f) \geq \text{Rank}(M_f)$

- If $M = \Sigma_{i \leq r} B_i$ with $\text{Rank}(B_i)=1$, then $\text{Rank}(M) \leq r$
- $M_f = \Sigma_{i \leq \chi(f)} \text{Tile}_i$, where Tile_i has a monochromatic rectangle and 0's elsewhere

$\chi(f) \geq \text{Rank}(M_f)$

- If $M = \Sigma_{i \leq r} B_i$ with $\text{Rank}(B_i)=1$, then $\text{Rank}(M) \leq r$
- $M_f = \Sigma_{i \leq \chi(f)} \text{Tile}_i$, where Tile_i has a monochromatic rectangle and 0's elsewhere
  - $\text{Rank}(\text{Tile}_i)=1$

$\chi(f) \geq \text{Rank}(M_f)$

- If $M = \Sigma_{i \leq r} B_i$ with $\text{Rank}(B_i)=1$, then $\text{Rank}(M) \leq r$
- $M_f = \Sigma_{i \leq \chi(f)} \text{Tile}_i$, where Tile_i has a monochromatic rectangle and 0's elsewhere
  - $\text{Rank}(\text{Tile}_i)=1$
  - $\text{Rank}(M_f) \leq \chi(f)$

$\chi(f) \geq \text{Rank}(M_f)$

- If $M = \Sigma_{i\leq r} B_i$ with $\text{Rank}(B_i)=1$, then $\text{Rank}(M) \leq r$
- $M_f = \Sigma_{i\leq\chi(f)} \text{Tile}_i$, where Tile_i has a monochromatic rectangle and 0's elsewhere
  - $\text{Rank}(\text{Tile}_i)=1$
  - $\text{Rank}(M_f) \leq \chi(f)$
- $CC(f) \geq \log(\chi(f)) \geq \log(\text{Rank}(M_f))$

Discrepancy

Discrepancy

- Discrepancy of a 0-1 matrix

Discrepancy

- Discrepancy of a 0-1 matrix
  - max “imbalance” in any rectangle

Discrepancy

- Discrepancy of a 0-1 matrix
  - max “imbalance” in any rectangle
  - Imbalance = | #1’s - #0’s |

Discrepancy

- Discrepancy of a 0-1 matrix
  - max "imbalance" in any rectangle
  - Imbalance = | #1's - #0's |
  - Disc(M) = 1/(mn) $\max_{rect}$ imbalance(rect)

Discrepancy

- Discrepancy of a 0-1 matrix
  - max "imbalance" in any rectangle
  - Imbalance = | #1's - #0's |
  - Disc(M) = $1/(mn) \max_{rect}$ imbalance(rect)
- $\chi(f) \geq 1/\text{Disc}(M_f)$

Discrepancy

- Discrepancy of a 0-1 matrix
  - max "imbalance" in any rectangle
  - Imbalance = | #1's - #0's |
  - $Disc(M) = 1/(mn) \max_{rect} imbalance(rect)$
- $\chi(f) \geq 1/Disc(M_f)$
  - $Disc(M_f) \geq 1/(mn)$ (size of largest monochromatic tile)

Discrepancy

- Discrepancy of a 0-1 matrix
  - max "imbalance" in any rectangle
  - Imbalance = | #1's - #0's |
  - $\text{Disc}(M) = 1/(mn) \max_{rect} \text{imbalance}(rect)$
- $\chi(f) \geq 1/\text{Disc}(M_f)$
  - $\text{Disc}(M_f) \geq 1/(mn)$ (size of largest monochromatic tile)
  - $\chi(f) \geq (mn)/($size of largest monochromatic tile$)$

CC Lower-bounds Summary

CC Lower-bounds Summary

- $CC(f) \geq \log(\#\text{transcripts})$

CC Lower-bounds Summary

- CC(f) ≥ log(#transcripts)
  - Tiling Lower-bound: #transcripts ≥ χ(f)

CC Lower-bounds Summary

- $CC(f) \geq \log(\#transcripts)$
  - Tiling Lower-bound: $\#transcripts \geq \chi(f)$
  - Both fairly tight: $CC(f) = O(\log^2(\chi(f)))$

CC Lower-bounds Summary

- CC(f) ≥ log(#transcripts)
  - Tiling Lower-bound: #transcripts ≥ χ(f)
  - Both fairly tight: CC(f) = O($\log^2(\chi(f))$)
- To lower-bound χ(f): fooling-set, rank, 1/Disc

CC Lower-bounds Summary

- $CC(f) \geq \log(\#\text{transcripts})$
  - Tiling Lower-bound: $\#\text{transcripts} \geq \chi(f)$
  - Both fairly tight: $CC(f) = O(\log^2(\chi(f)))$
- To lower-bound $\chi(f)$: fooling-set, rank, 1/Disc
  - $\chi(f) \geq |\text{max fooling-set}| \geq (\text{Rank}(M_f))^2$

CC Lower-bounds Summary

- CC(f) ≥ log(#transcripts)
  - Tiling Lower-bound: #transcripts ≥ $\chi(f)$
  - Both fairly tight: $CC(f) = O(\log^2(\chi(f)))$
- To lower-bound $\chi(f)$: fooling-set, rank, 1/Disc
  - $\chi(f) \geq |\max \text{ fooling-set}| \geq (\text{Rank}(M_f))^2$
  - 1/Discrepancy lower-bounds can be very loose

CC Lower-bounds Summary

- $CC(f) \geq \log(\#\text{transcripts})$
  - Tiling Lower-bound: $\#\text{transcripts} \geq \chi(f)$
  - Both fairly tight: $CC(f) = O(\log^2(\chi(f)))$
- To lower-bound $\chi(f)$: fooling-set, rank, 1/Disc
  - $\chi(f) \geq |\text{max fooling-set}| \geq (\text{Rank}(M_f))^2$
  - 1/Discrepancy lower-bounds can be very loose
  - Conjecture: $\text{Rank}(M_f)$ (and hence fooling set) is fairly tight

CC Lower-bounds Summary

- $CC(f) \geq \log(\#transcripts)$
  - Tiling Lower-bound: $\#transcripts \geq \chi(f)$
  - Both fairly tight: $CC(f) = O(\log^2(\chi(f)))$
- To lower-bound $\chi(f)$: fooling-set, rank, 1/Disc
  - $\chi(f) \geq |\max \text{ fooling-set}| \geq (\text{Rank}(M_f))^2$
  - 1/Discrepancy lower-bounds can be very loose
  - Conjecture: $\text{Rank}(M_f)$ (and hence fooling set) is fairly tight
    - i.e., $CC(f) = O(\text{polylog}(\text{Rank}(M_f)))$

Many Variants

Many Variants

- Randomized protocols: significant savings in expectation

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess
- Multi-party: Input split across multiple parties. Broadcast channels for communication.

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess
- Multi-party: Input split across multiple parties. Broadcast channels for communication.
  - Number on the forehead version

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess
- Multi-party: Input split across multiple parties. Broadcast channels for communication.
    - Number on the forehead version
- Non-boolean output

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess
- Multi-party: Input split across multiple parties. Broadcast channels for communication.
    - Number on the forehead version
- Non-boolean output
- Multi-valued functions: agree on one value

Many Variants

- Randomized protocols: significant savings in expectation
- Non-deterministic: Alice and Bob are non-deterministic. "Communication" now includes shared guess
- Multi-party: Input split across multiple parties. Broadcast channels for communication.
  - Number on the forehead version
- Non-boolean output
- Multi-valued functions: agree on one value
- Different costs: asymmetric communication, average-case complexity