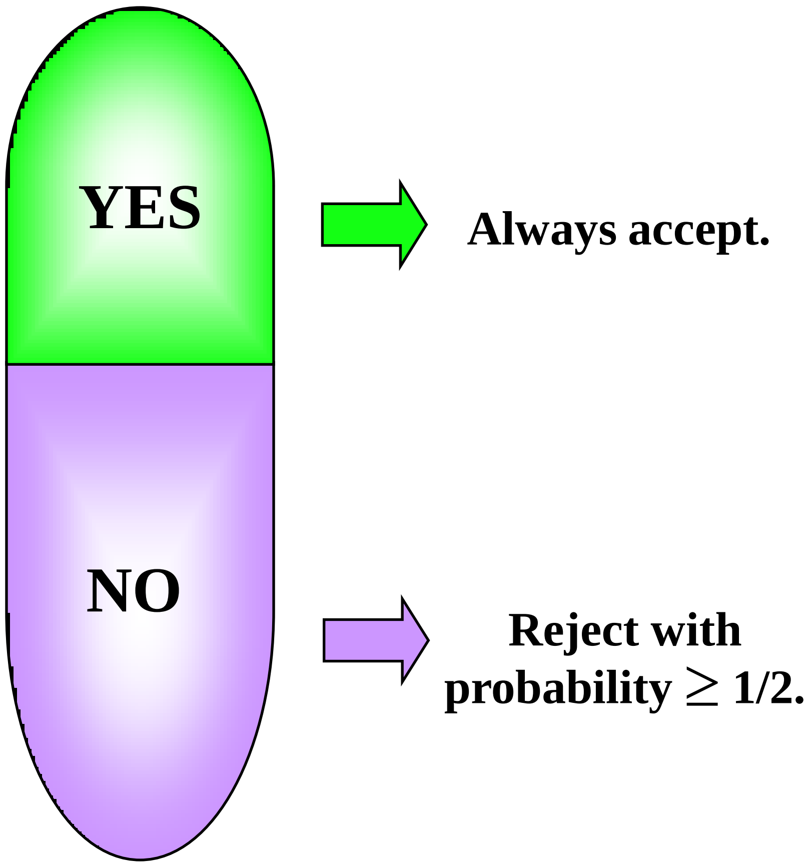
Monotonicity Testing

Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron and Alex Samorodnitsky

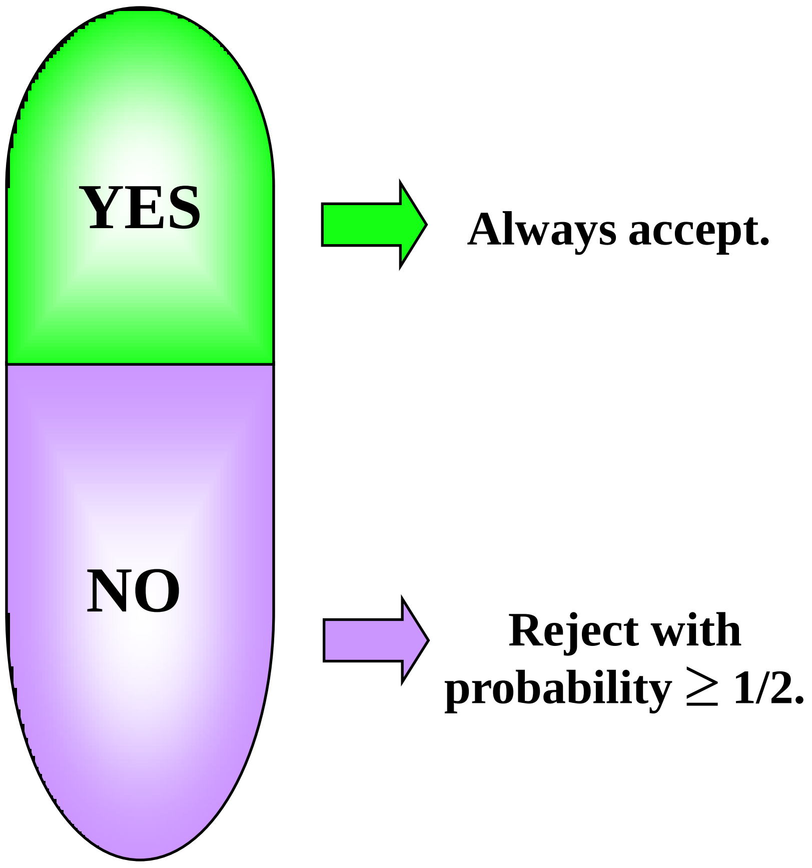
Probabilistic Property Testing

Probabilistic Algorithm

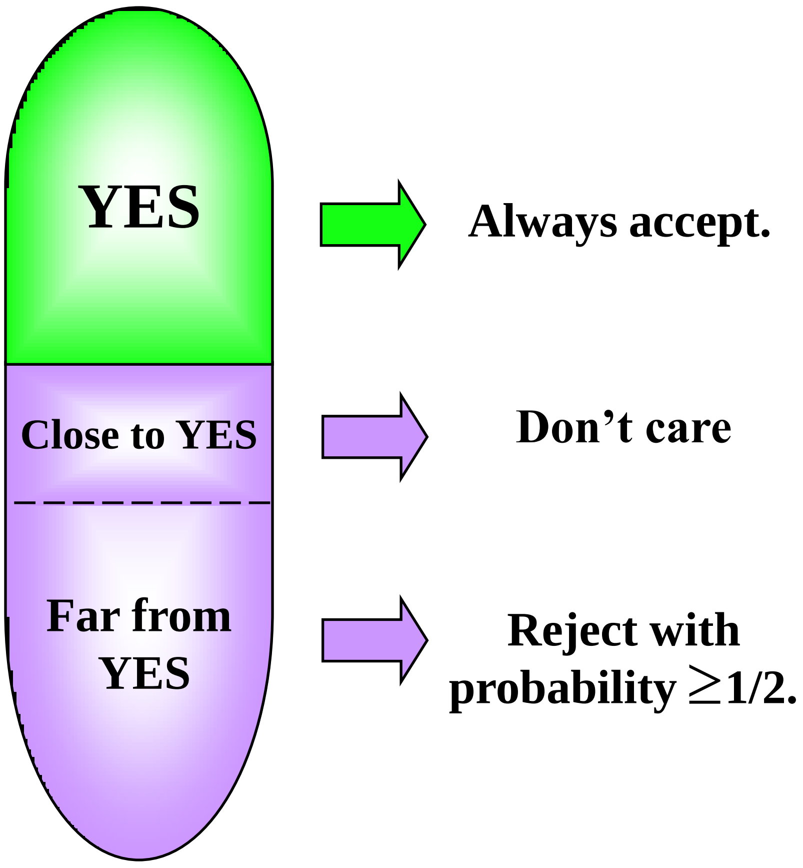


Probabilistic Property Testing

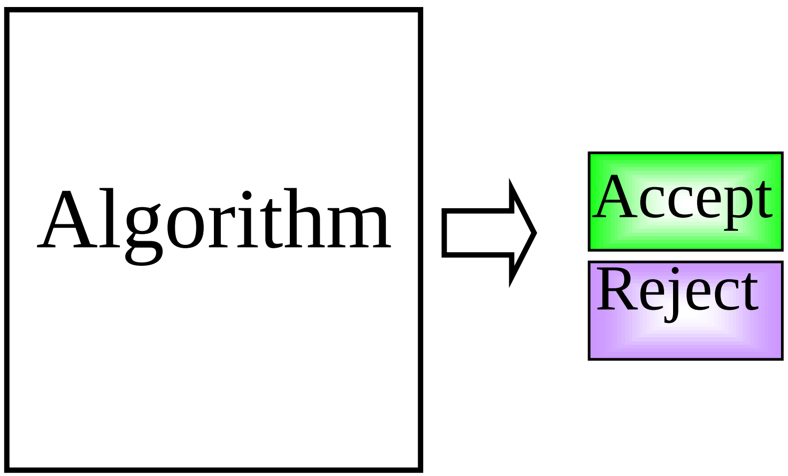
Probabilistic Algorithm



Probabilistic Property Tester



Probabilistic Property Tester

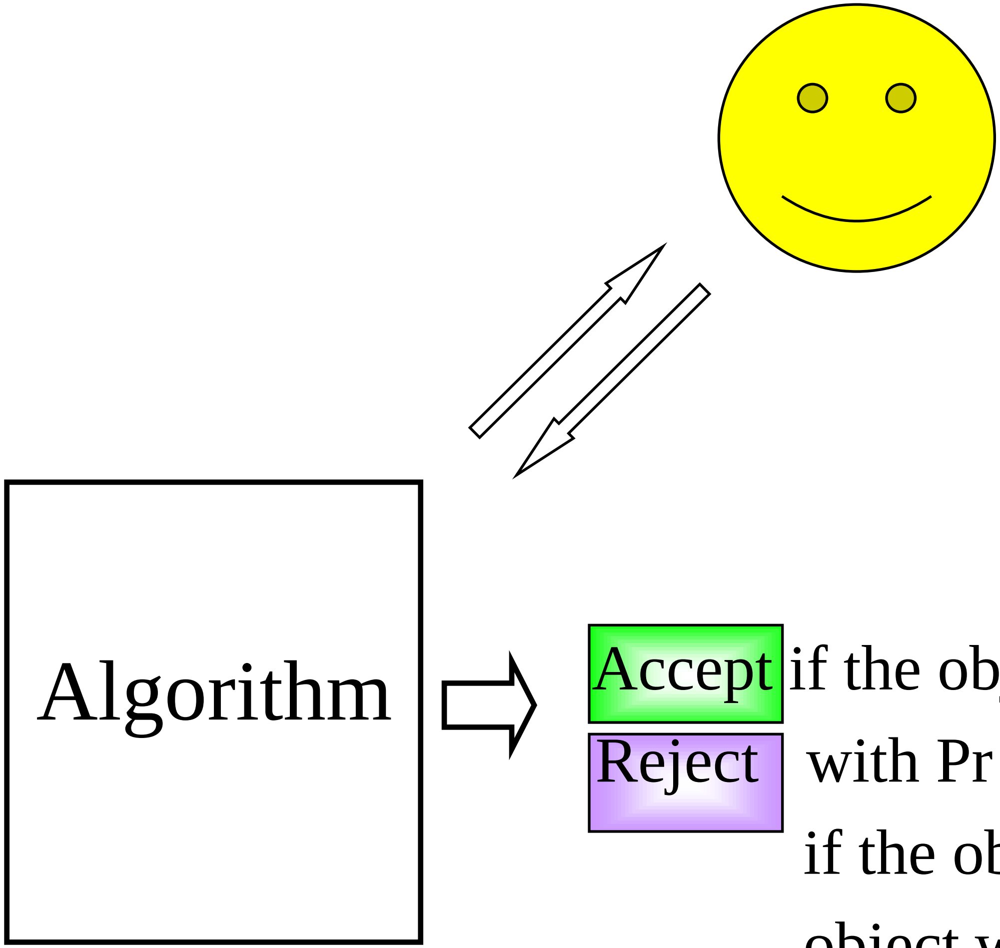


if the object has the property.

with Pr $\geq$ 1/2

if the object is far from any object with the property.

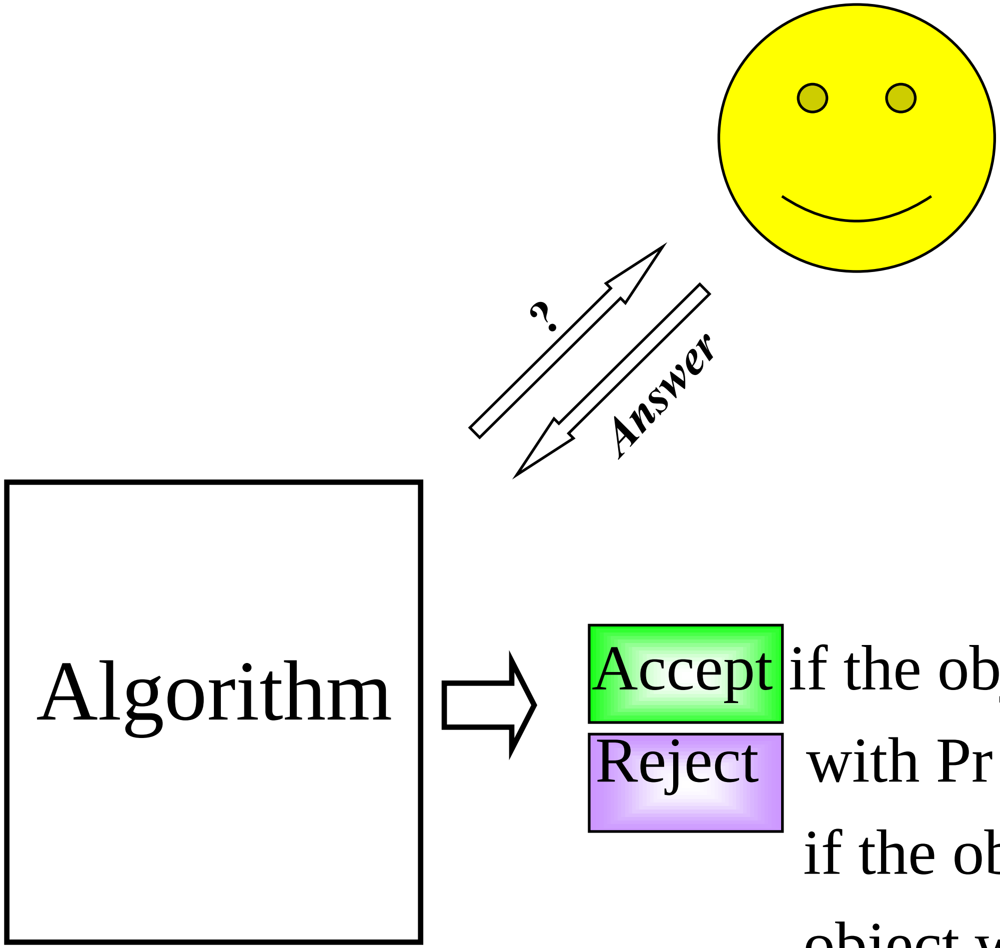
Probabilistic Property Tester



Accept if the object has the property.

Reject with Pr ≥ 1/2 if the object is far from any object with the property.

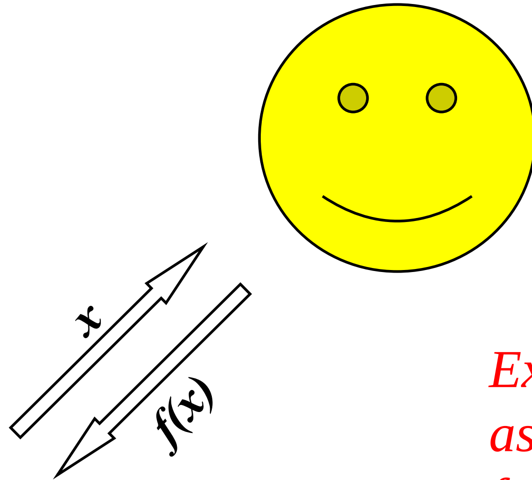
Probabilistic Property Tester



Accept if the object has the property.

Reject with Pr ≥ 1/2 if the object is far from any object with the property.

Probabilistic Property Tester



Express property testing as testing properties of functions.

Algorithm ⇒

Accept if the object has the property.

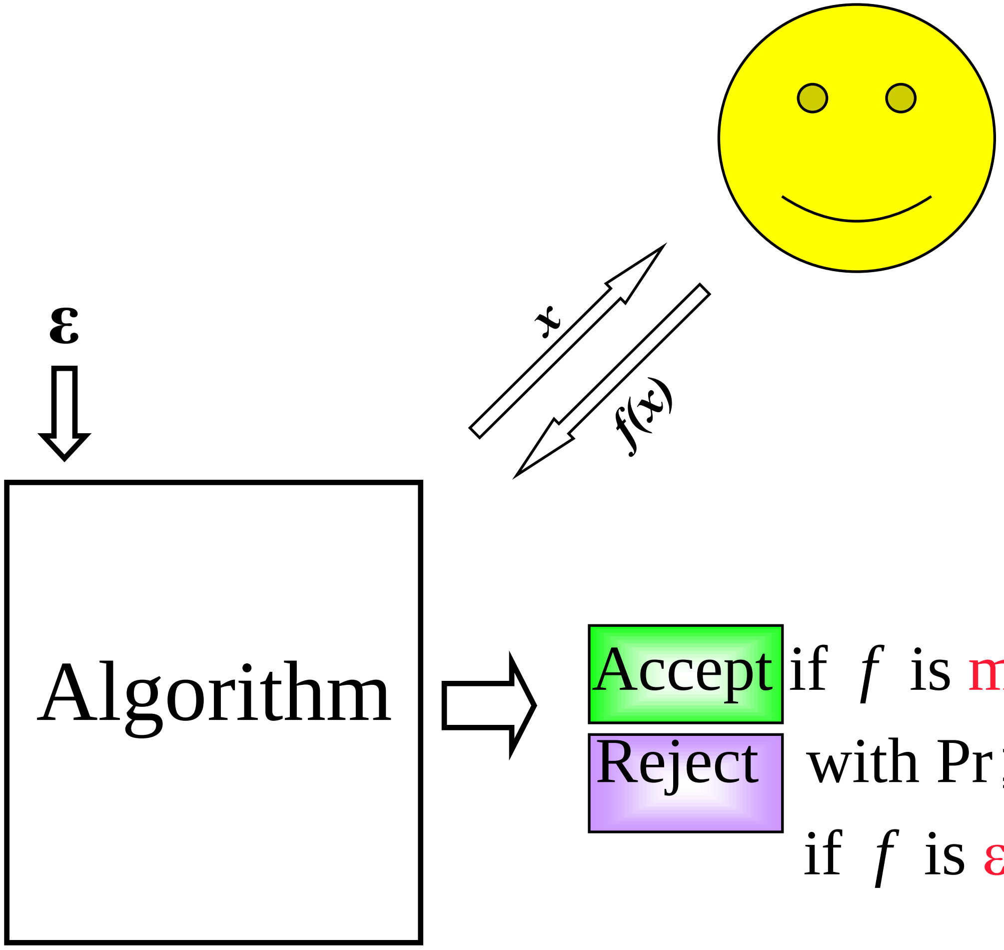
Reject with Pr $\geq$ 1/2 if the object is far from any object with the property.

Motivation

Probabilistic Property Tester **can be**

- much faster than an exact algorithm;
- the only option when the exact problem is not decidable;
- used for preprocessing;
- good enough in application where some errors are tolerable.

Problem Statement



Accept if f is monotone.

Reject with $\Pr \geq 1/2$ if f is ε-far from monotone.

Definitions for $f : \Sigma^n \mapsto R$

- For two n-symbol strings x and y we say $x \prec y$ if

  y is formed from x by increasing one or more symbols.

examples

001001
011001
011111

? 100000

Definitions for $f : \Sigma^n \mapsto R$

- For two n-symbol strings x and y we say $x \prec y$ if
  y is formed from x by increasing one or more symbols.

- f is monotone if
  $f(x) \le f(y)$ for all $x \prec y$.

examples

001001
011001
011111
?
100000

x	0	1	2	3	4	5
$f(x)$	0	3	7	8	8	9

Definitions for $f : \Sigma^n \mapsto R$

examples

- For two n-symbol strings x and y we say $x \prec y$ if
  y is formed from x by increasing one or more symbols.

001001
011001
011111
? 100000

- f is monotone if
  $f(x) \le f(y)$ for all $x \prec y$.

x	0	1	2	3	4	5
$f(x)$	0	3	7	8	8	9

- f is ε-far from monotone if
  every monotone function disagrees with f on at least an ε-fraction of the domain.

x	0	1	2	3	4	5
$f(x)$	0	3	9	8	7	9

1/3-far from monotone

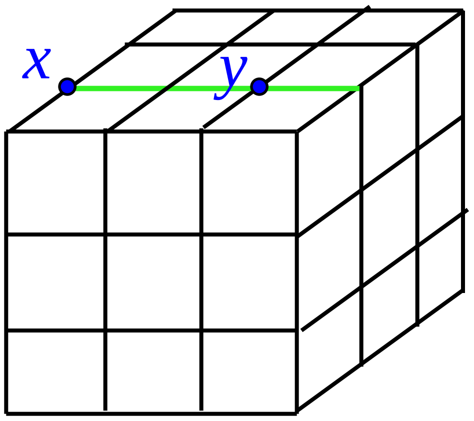
Results

Q = Query Complexity of Monotonicity Tests

[GGLR98] $$Q = O\left(\frac{n^2}{\varepsilon} \cdot |\Sigma|^2 \cdot |R|\right)$$

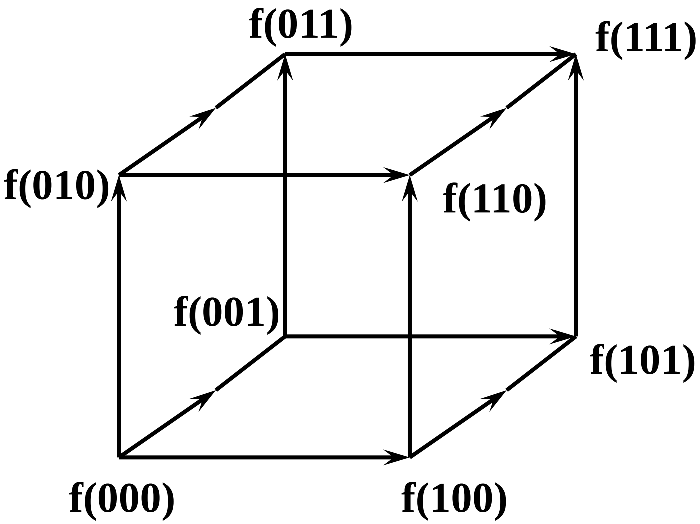
This work $$Q = O\left(\frac{n}{\varepsilon} \cdot \log|\Sigma| \cdot \log|R|\right)$$

Algorithm (Reduction to a simpler case)



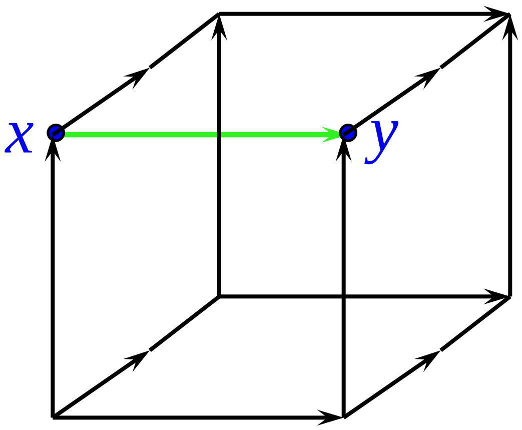
- INPUT:
  - ε and $f : \Sigma^n \mapsto R$
- Repeat several times:
  - Pick a **line** along the axes of the hyper-grid uniformly at random.
  - Use your favorite algorithm to test if the **line** is monotone [our paper, EKKRV98, Noga Alon] .
  - If a pair (x, y) of points on the **line** with $x \prec y$ and $f(x) > f(y)$ is found, then REJECT.
- Otherwise, ACCEPT.

Special case: $f : \{0,1\}^n \mapsto R$



- Edge $x \rightarrow y$ iff $x \prec y$ and
  x and y differ in one coordinate
- Edge $x \rightarrow y$ is a violated edge of f
  if $f(x) > f(y)$.

Algorithm for $f : \{0,1\}^n \mapsto R$



- INPUT:
  - ε and $f : \{0,1\}^n \mapsto R$
- Repeat ***Q/2*** times:
  - Pick an **edge** $x \rightarrow y$ uniformly at random.
  - If $x \rightarrow y$ is violated (i.e. $f(x) > f(y)$), then REJECT.
- Otherwise, ACCEPT.

Intuition for Analysis

- If f is monotone, the algorithm always accepts.

YES

- If f is not monotone:
  - If f has few violated edges, we can make f monotone by changing its value at a few points.

Don't care

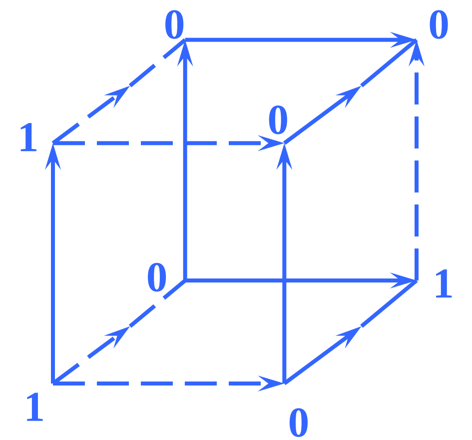
  - If f has many violated edges, the algorithm succeeds with high probability.

NO

Proof Plan

- BINARY RANGE ($f:\{0,1\}^n \mapsto \{0,1\}$)

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges}$$



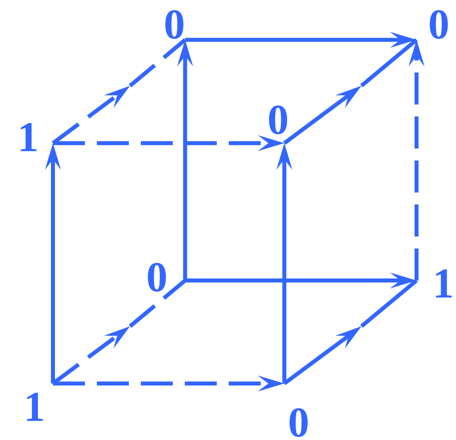
THEOREM: If f is ε-far from monotone, then a random edge is violated with probability

$$\frac{\#\,\text{violated edges}}{n2^n} \ge \frac{\#\,\text{altered points}}{2 \cdot n2^n} \ge \frac{\varepsilon 2^n}{2 \cdot \text{n}2^n} \ge \frac{\varepsilon}{2\text{n}}.$$

Proof Plan

- BINARY RANGE ($f:\{0,1\}^n \mapsto \{0,1\}$)

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges}$$



THEOREM: If f is ε-far from monotone, then a random edge is violated with probability

$$\frac{\#\,\text{violated edges}}{n2^n} \ge \frac{\#\,\text{altered points}}{2 \cdot n2^n} \ge \frac{\varepsilon 2^n}{2 \cdot \mathrm{n}2^n} \ge \frac{\varepsilon}{2\mathrm{n}}.$$

- RANGE REDUCTION ($f:\{0,1\}^n \mapsto R$)

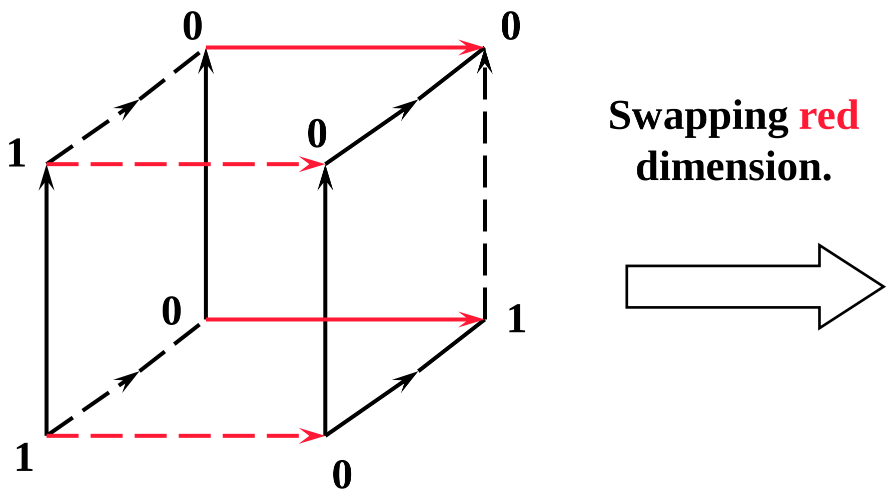
$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot \log|R|$$

THEOREM: If f is ε-far from monotone, then a random edge is violated with probability

$$\frac{\#\,\text{violated edges}}{n2^n} \ge \frac{\#\,\text{altered points}}{n2^n \cdot 2\log|R|} \ge \frac{\varepsilon 2^n}{\mathrm{n}2^n \cdot 2\log|R|} \ge \frac{\varepsilon}{2\mathrm{n}\log|R|}.$$

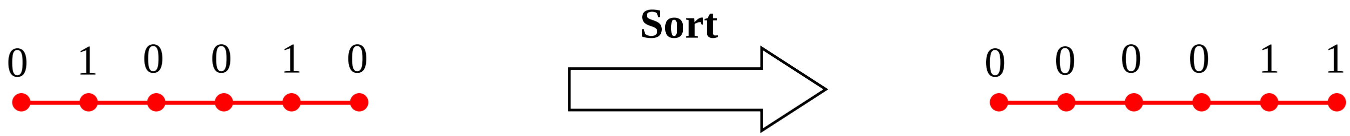
Repairing Violated Edges in One Dimension

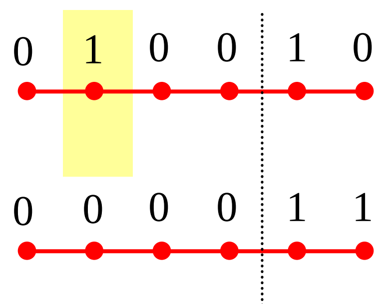
Swap violated edges $1 \rightarrow 0$ **in red dimension to** $0 \rightarrow 1$**.**



Repairing Violated Edges in One Dimension

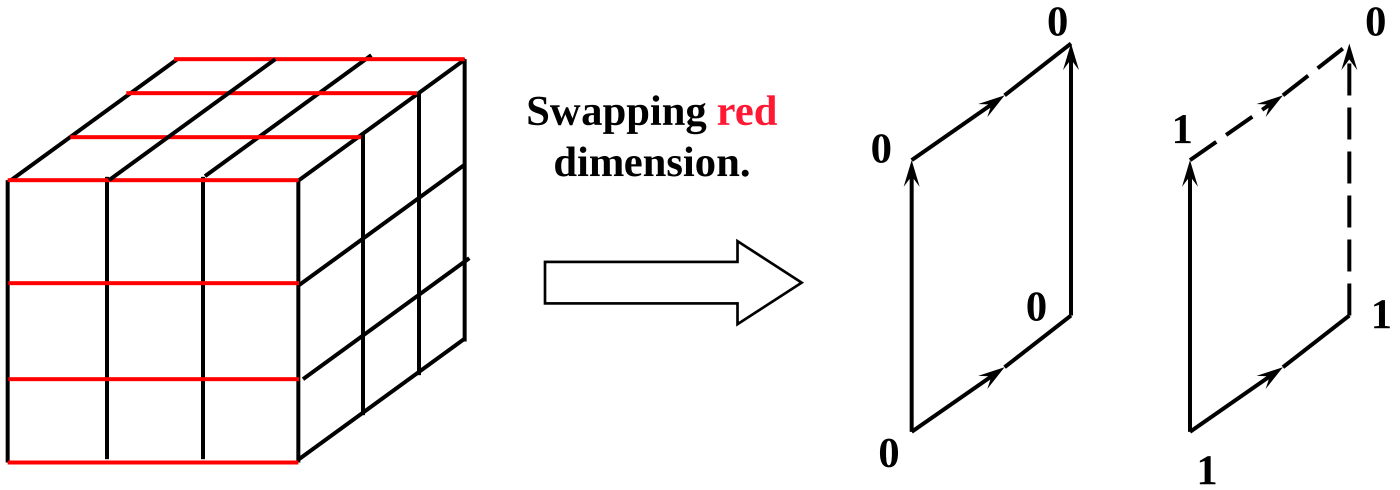
Swap violated edges $1\rightarrow 0$ **in red dimension to** $0\rightarrow 1$**.**





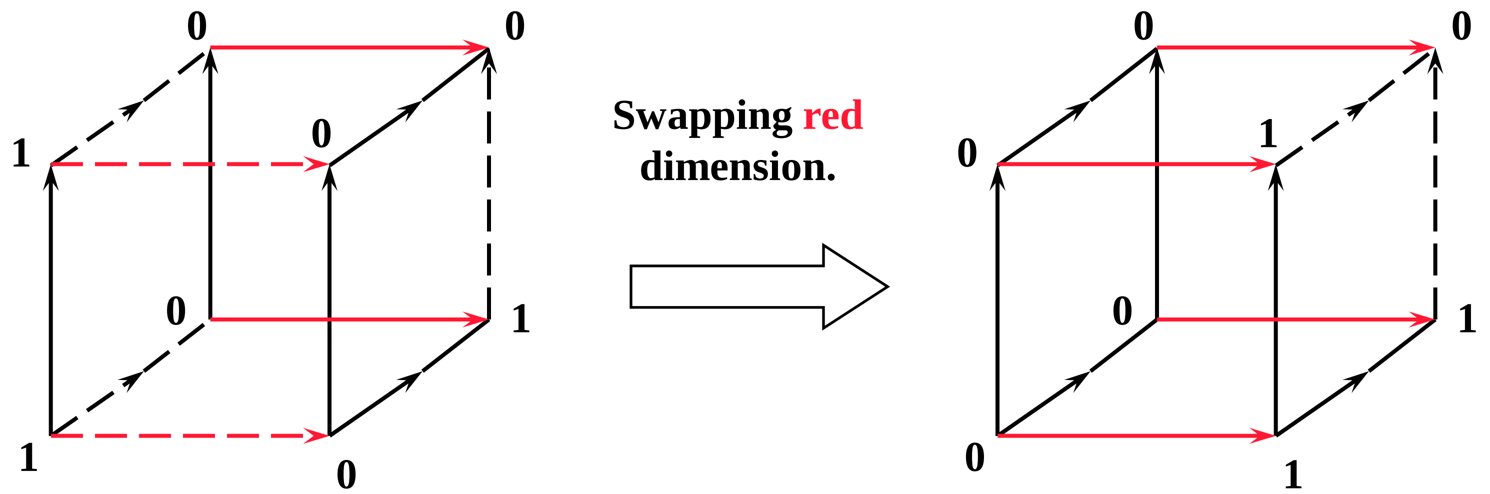
Repairing Violated Edges in One Dimension

Swap violated edges $1 \to 0$ **in red dimension to** $0 \to 1$.



Repairing Violated Edges in One Dimension

Swap violated edges $1\rightarrow 0$ **in red dimension to** $0\rightarrow 1$**.**

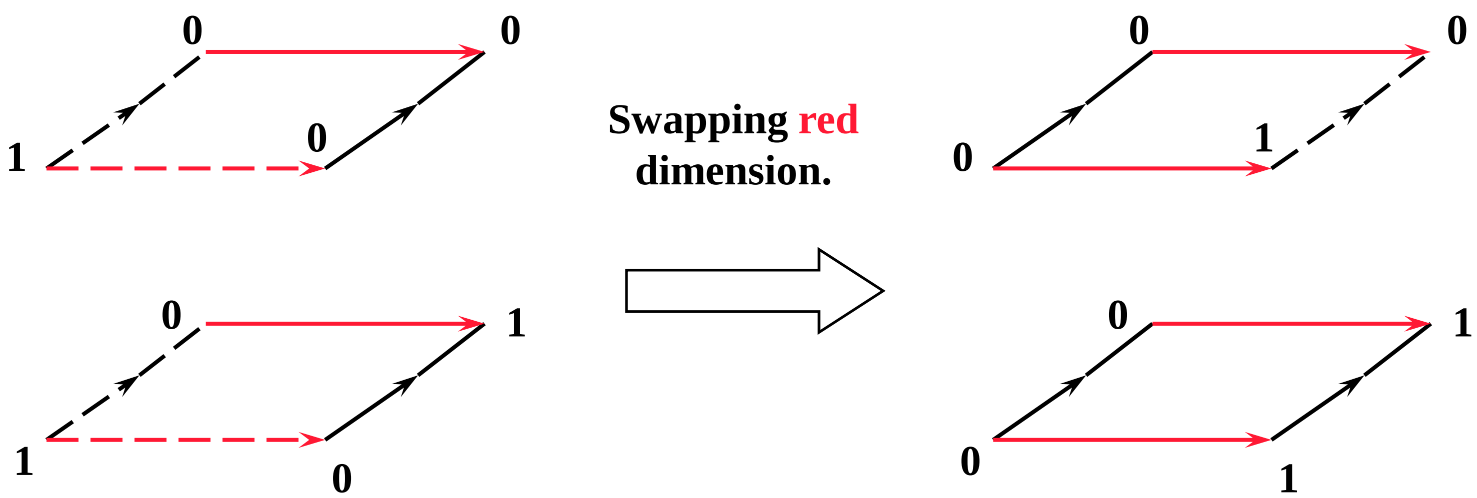


LEMMA. Swapping violated edges in dimension i

1. repairs all violated edges in dimension i;
2. does not increase the number of violated edges in dimension j, for all $j \neq i$.

Repairing Violated Edges in One Dimension

Swap violated edges *1→0* **in red dimension to** *0→1*.



LEMMA. Swapping violated edges in dimension *i*

1. repairs all violated edges in dimension *i*;
2. does not increase the number of violated edges in dimension *j*, for all $j \neq i$.

Back to the Proof Plan

- BINARY RANGE ($f : \{0,1\}^n \mapsto \{0,1\}$)

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges}$$

- RANGE REDUCTION ($f : \{0,1\}^n \mapsto R$)

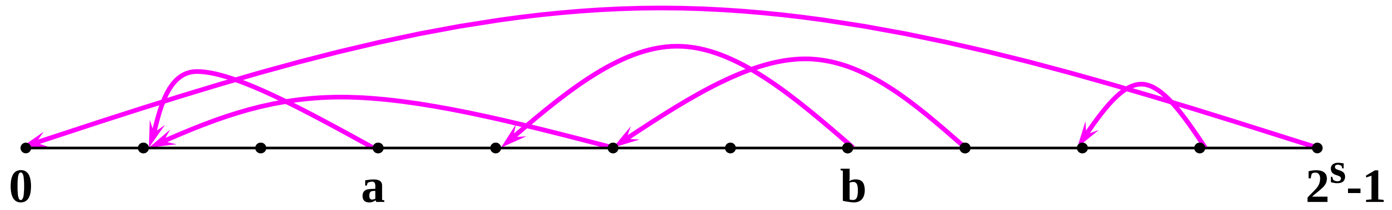
$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot \log |R|$$

W.l.g. assume $R=\{0, 1, \ldots, 2^s-1\}$.

Prove $\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot s$

by induction on s.

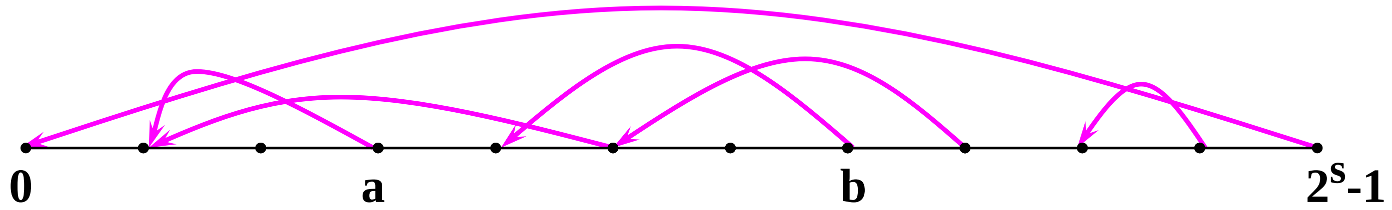
How can we make f monotone?



violated edges of *f*

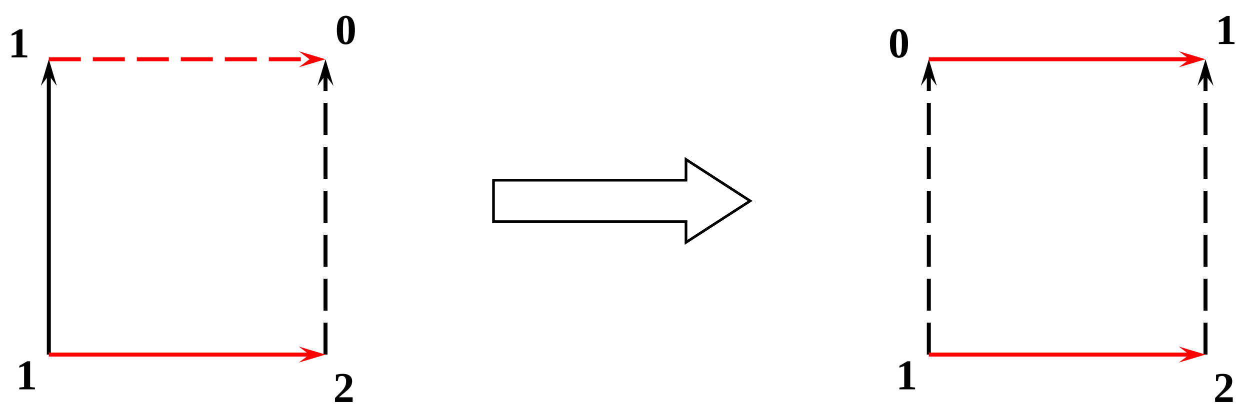
Swap violated edges in red dimension?

How can we make f monotone?

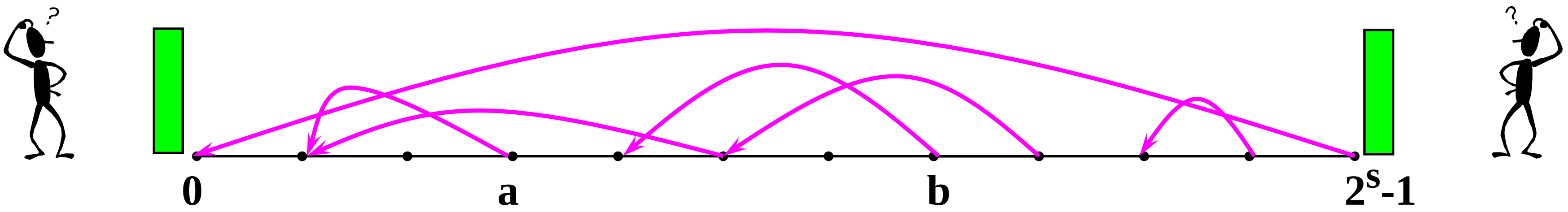


violated edges of *f*

Swap violated edges in red dimension?

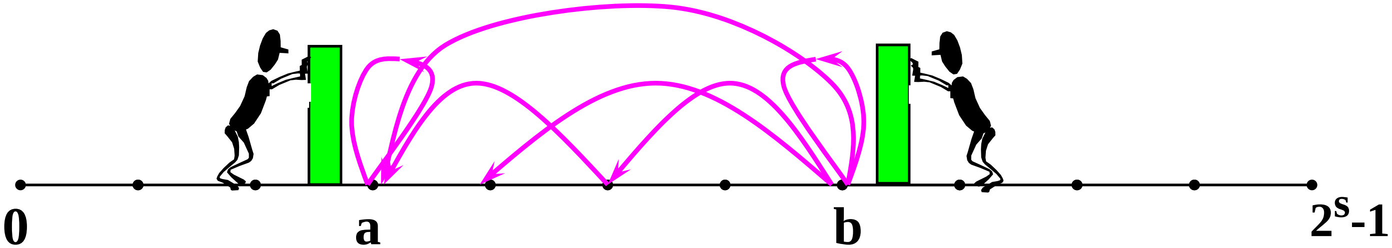


Operator SQUASH



violated edges of *f*

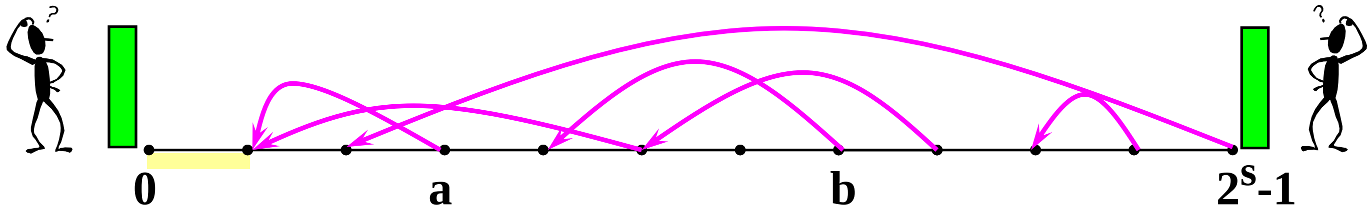
$$\text{SQUASH}[f,a,b](x)=\begin{cases} a & \text{if } f(x)\le a \\ b & \text{if } f(x)\ge b \\ f(x) & \text{otherwise} \end{cases}$$



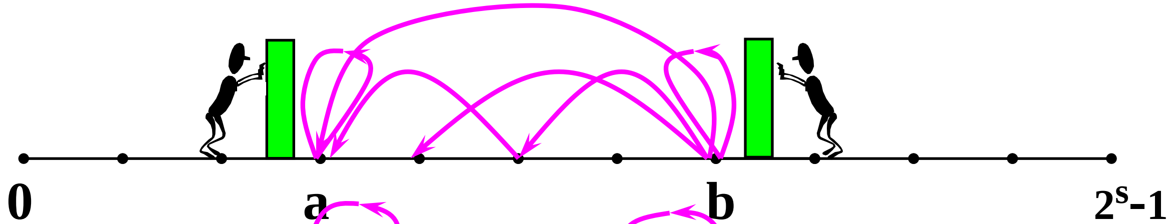
violated edges of
SQUASH[*f*,*a*,*b*]

PROPERTY: does not introduce new violated edges.

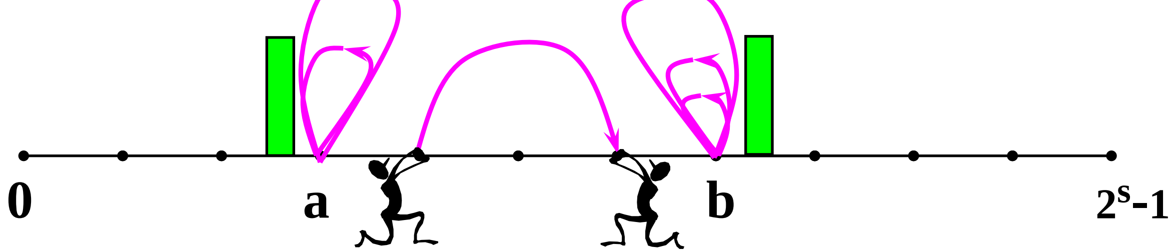
Operator CLEAR



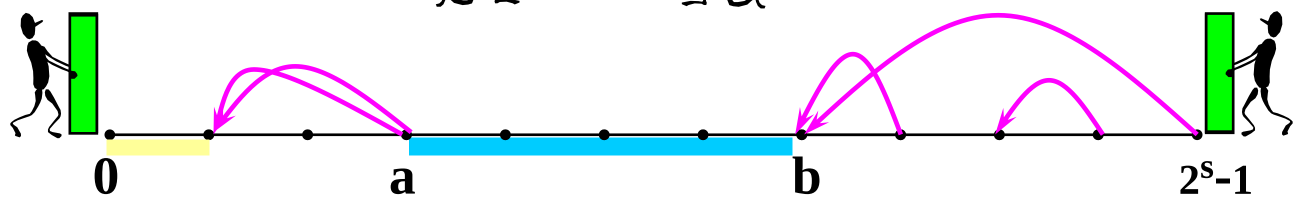
Squash:



Repair (**switch to closest monotone function with range {a,...,b}**):



Unsquash **values not altered by repair**:



PROPERTIES:

1. does not introduce new violated edges.
2. clears interval [a, b].
3. leaves clear intervals clear.

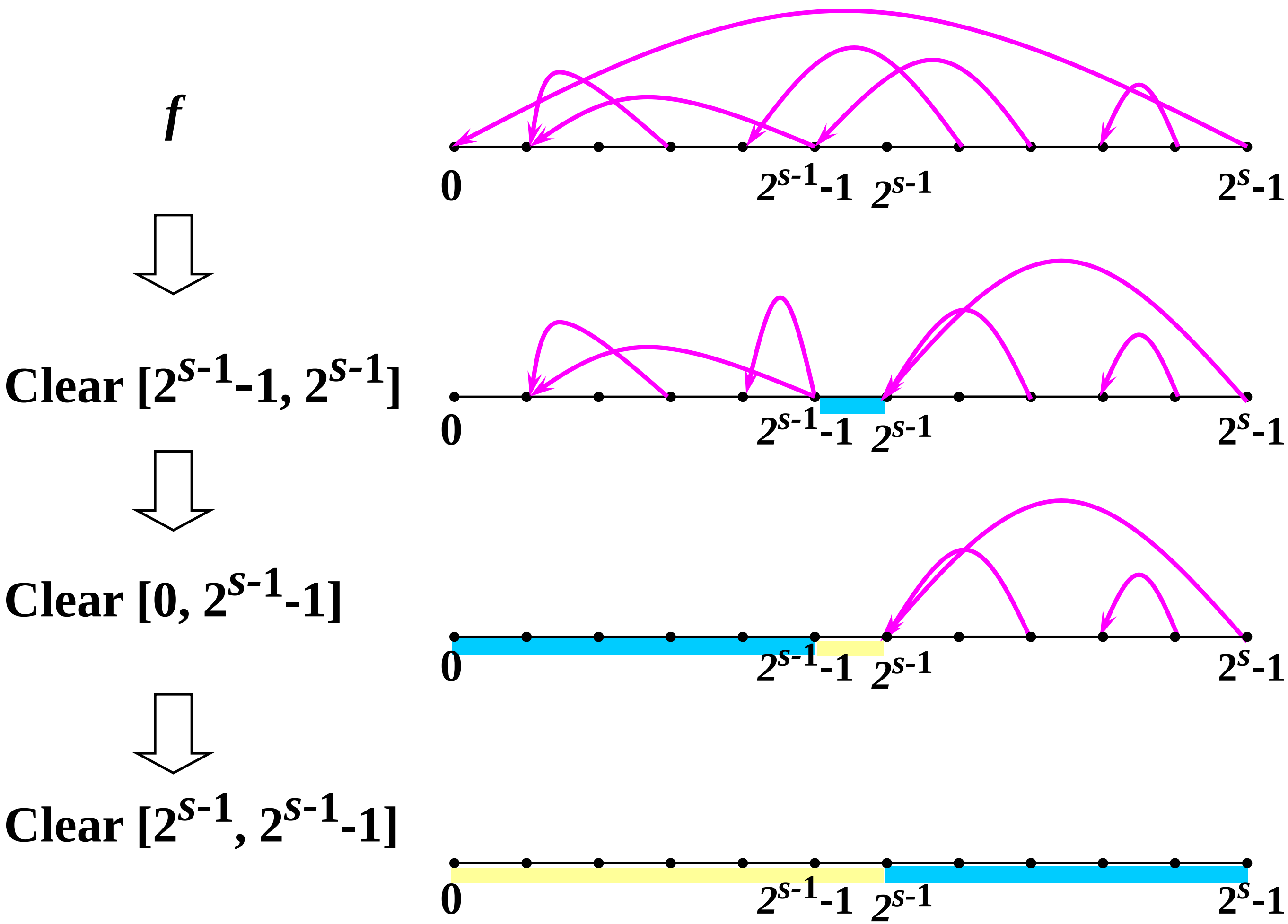
Making f monotone

f

Clear $[2^{s-1}-1, 2^{s-1}]$

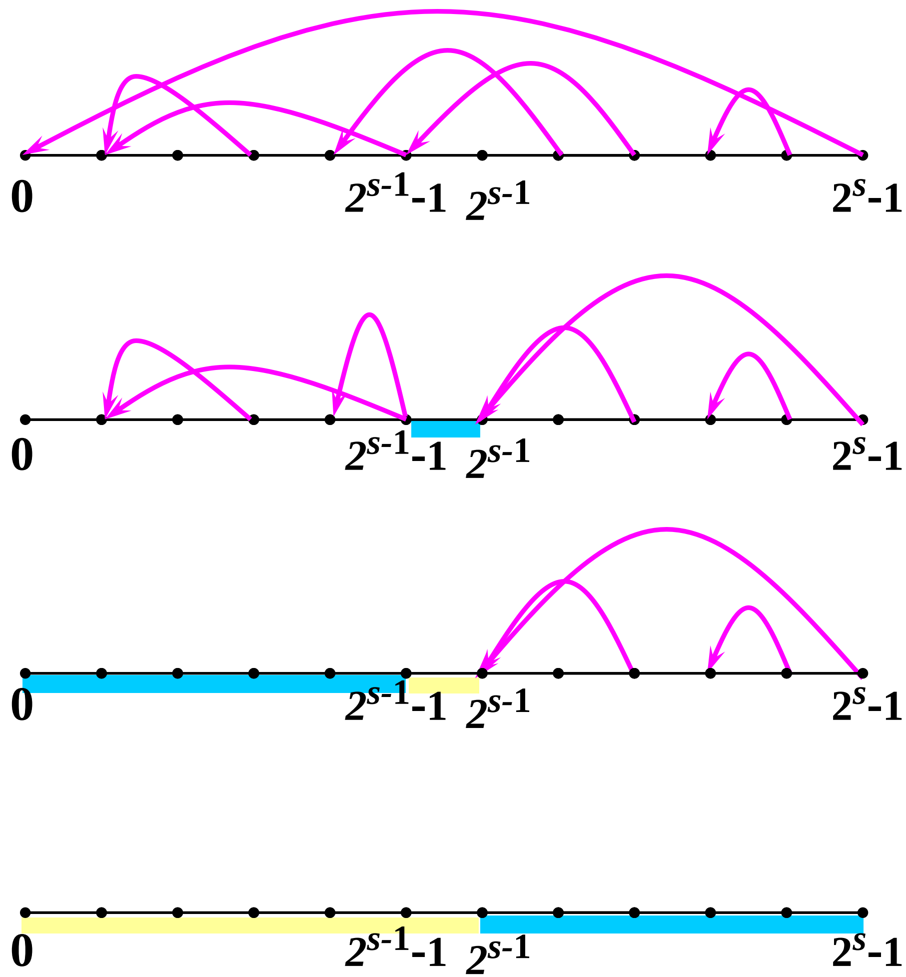
Clear $[0, 2^{s-1}-1]$

Clear $[2^{s-1}, 2^{s-1}-1]$



Making f monotone

		#violated edges
f		$2v$
Clear $[2^{s-1}-1, 2^{s-1}]$		$2(l+r) \le 2v$
Clear $[0, 2^{s-1}-1]$		$2r$
Clear $[2^{s-1}, 2^{s-1}-1]$		0



Reminder

We are proving (by induction on S)

that for functions with a range of size 2^S,

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot S.$$

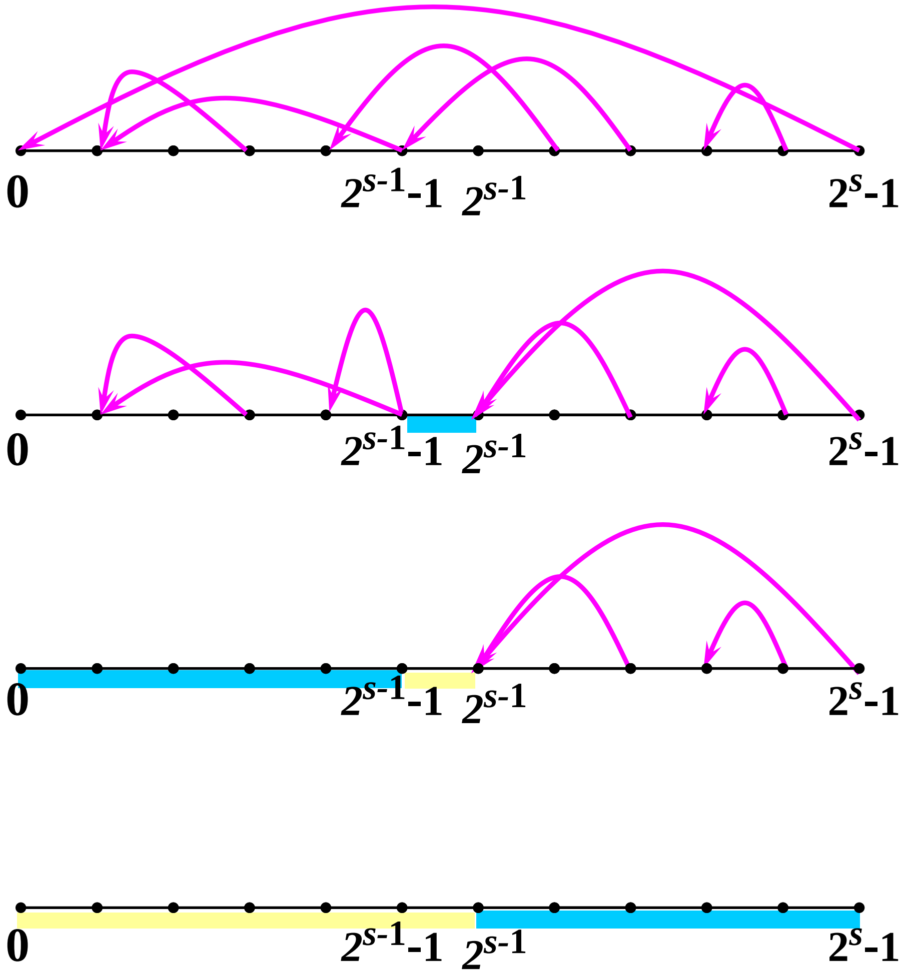
Base case [functions with a range of size 2]:

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges}.$$

Induction hypothesis [functions with a range of size 2^{S-1}]:

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot (S-1).$$

Making f monotone



	#violated edges	#points changed
f	$2v$	**0**
Clear [2^{s-1}-1, 2^{s-1}]	$2(l+r) \le 2v$	$\le 2v$
Clear [0, 2^{s-1}-1]	$2r$	$\le 2l(s-1)$
Clear [2^{s-1}, 2^{s-1}-1]	0	$\le 2r(s-1)$
	total	$\le 2v \cdot s$

We are done

- BINARY RANGE ($f : \{0,1\}^n \mapsto \{0,1\}$)

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges}$$

- RANGE REDUCTION ($f : \{0,1\}^n \mapsto R$)

$$\#\,\text{altered points} \le 2 \cdot \#\,\text{violated edges} \cdot \log |R|$$

W.l.g. assume $R=\{0, 1,\ldots, 2^s\text{-}1\}$.

Proof by induction on s.

Conclusions

- SUMMARY OF RESULTS IN THIS TALK
  - Monotonicity test for $f:\{0,1\}^n \mapsto \{0,1\}$ [switching argument].
  - Monotonicity test for $f:\{0,1\}^n \mapsto R$ [SQUASH and CLEAR argument].
- SUMMARY OF RESULTS IN THE PAPER
  - Designed good monotonicity tests for $f:\Sigma \mapsto \{0,1\}$.
  - Reduced testing monotonicity of $f:\Sigma^n \mapsto \{0,1\}$ to the case $n=1$ [sorting argument (a generalization of the switching argument)].
  - Reduced testing monotonicity of $f:\Sigma^n \mapsto R$ to the case $f:\Sigma^n \mapsto \{0,1\}$ [SQUASH and CLEAR argument].
- OPEN PROBLEMS
  - Query complexity independent of the size of the range?
  - $f:D \mapsto R$, where D is any partially ordered set.
  - Tests for other properties.