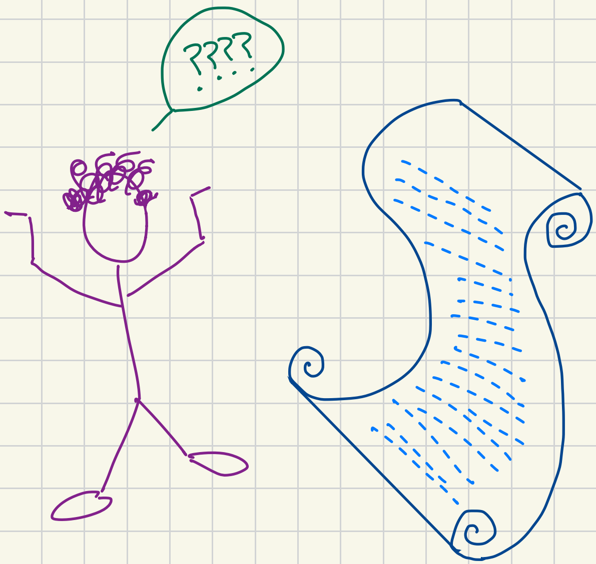
Probabilistically Checkable Proof Systems



Review:

$x \cdot y = \sum x_i y_i$ "inner product"

$x \circ y = (x_1 y_1, x_1 y_2, \ldots, x_i y_j, \ldots x_n y_n)$ "outer product"

n-bit vectors

n^2 bit vector

Fact if $\bar{a} \neq \bar{b}$ then $\Pr_{\bar{r} \in \{0,1\}^n} [\bar{a} \cdot \bar{r} \neq \bar{b} \cdot \bar{r}] \geq \frac{1}{2}$

if $A \cdot B \neq C$ then $\Pr_{\bar{r}} [A \cdot B \cdot \bar{r} \neq C \cdot \bar{r}] \geq 1/2$

$\Big\}$ also true for $\equiv \bmod 2$

(for proof of fact, see last lecture)

self-correcting: if f $\frac{1}{8}$-linear then SC-f is linear + uses $O(\log \frac{1}{\beta})$ queries to f (but may fail on any input with prob $\leq \beta$)

if f $\frac{1}{8}$-linear

define $g(x)$: Do $\log \frac{1}{\beta}$ times
- pick y randomly
- $\text{answer}_i \leftarrow f(y) + f(x-y)$

Output most common answer

then $\forall x$, $\Pr[g(x) = f(x)] \geq 1-\beta$

self-testing:

Given f

Do $O(1/\varepsilon)$ times:

Pick x, y randomly

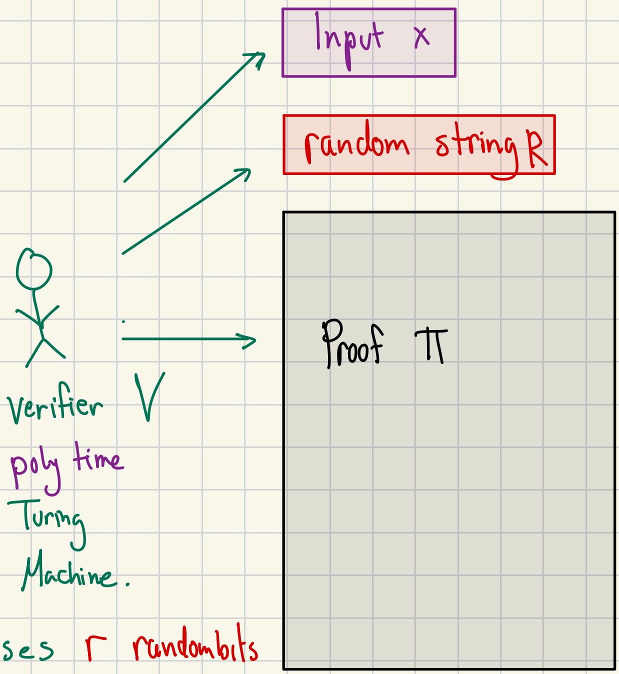
if $f(x+y) \neq f(x) + f(y)$ output "fail" & halt

Output "pass"

If f linear, test passes

If f ε-far from linear, $\Pr[\text{fail}] \geq 3/4$

Probabilistically Checkable Proofs



uses r randombits
+ q queries to proof

fixed function
"written in stone"
(or at least pen)
but
created by adversary
with unlimited
computational power.
does not change after
queries from verifier

Query to π: "what is i^{th} bit/word?"

def $L \in PCP(r, q)$ if $\exists V$ (ptime TM) s.t.

1) $\forall x \in L \quad \exists \pi$ s.t. $\Pr_{\text{V's random string}} [V, \pi \text{ accepts}] = 1$

2) $\forall x \notin L \quad \forall \pi' \quad \Pr_{\text{V's random string}} [V, \pi' \text{ accepts}] \leq 1/4$

note: $SAT \in PCP(0, n)$ ← proof is satisfying assignment. V doesn't need randomness to check.

Today: $NP \subseteq PCP(O(n^3), O(1))$

Actually: $NP \subseteq PCP(O(\log n), O(1))$

crazy? V only sees constantly many bits!

A warm up:

(recall: Fact if $\bar{a} \neq \bar{b}$ then $\Pr_{\bar{r} \in \{0,1\}^n} [\bar{a} \cdot \bar{r} \neq \bar{b} \cdot \bar{r}] \geq \frac{1}{2}$)

strange setting: get "inner product queries" in one step
(only queries cost, computations are free.)

Test if $\bar{a} = \bar{0}$ in constant queries:

Do several times:
pick $\bar{r} \in_u \{0,1\}^n$
if $\bar{a} \cdot \bar{r} \neq 0$ output "FAIL" + halt
Output "PASS"

Behavior:
if $\bar{a} = \bar{0}$ will always pass
if $\bar{a} \neq \bar{0}$, fact $\Rightarrow$ fail each time with prob $\geq 1/2$

why should we believe answers to queries are correct?
consistent with each other?
consistent with $\bar{a}$?

Making the warmup "less strange":

fix vector $\bar{a} = (a_1, a_2, \dots, a_n)$

allowed operations:

- query a_i
- specify $\bar{y}$ + query $\bar{a}\cdot\bar{y}$

} 1 step each

1st query is actually special case of $\bar{y} = e_i$

First idea for proof π:

note size of proof is 2^n → write out all answers to $\bar{a}\cdot\bar{y}$ (for each $\bar{y}$)

Test if $\bar{a} = \bar{0}$:

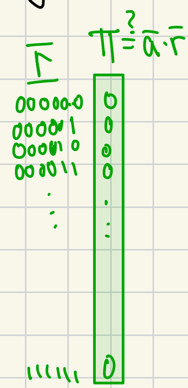
Do several times:

pick $\bar{r} \in_u \{0,1\}^n$

ask proof for value of $\bar{a}\cdot\bar{r}$

if $\bar{a}\cdot\bar{r} \neq 0$ output "FAIL" + halt

Output "PASS"



Problem: proof can cheat by writing all 0's in π (and many other ways)

How can we test that π doesn't cheat?

(will come back to this)

Idea: proof is a (extremely inefficient) way of "encoding" $\bar{a}$

Back to 3SAT:

3SAT: $F = \bigwedge C_i$ s.t. $C_i = (y_{i_1} \vee y_{i_2} \vee y_{i_3})$

where $y_{i_j} \in \{x_1 \ldots x_n, \bar{x}_1 \ldots \bar{x}_n\}$ "literals"

here $\bar{x}_i$ is "complement"

Warning:

FROM NOW ON, LOSING "BAR" FROM VECTORS TO PREVENT CONFUSION

First crack:

π = setting of sat assignment a

$a_1 = T$, $a_2 = F$, $a_3 = T, \ldots$

1 0 1 . . .

Possible protocol for V, given F, π:

pick random clause C_i

Check if a satisfies C_i ← 3 queries to π

behavior:

if a satisfies C, then

$\Pr[\text{Pass}] = 1$

if a doesn't satisfy C, then

$\Pr[\text{fail}] \geq \Pr[\text{pick } i \text{ s.t. } C_i(A) = F] \geq \frac{1}{\# \text{clauses}}$ ☹ not very good bound

e.g. $F = (x_1 \vee \bar{x}_2 \vee x_3)(x_2 \vee \bar{x}_3 \vee x_4)$

$a = (1, 0, 0, 0, \ldots)$

random clause $(x_2 \vee \bar{x}_3 \vee x_4)$ F T F ✓ passes

Arithmetization of 3SAT:

Boolean formula $F \iff$ arithmetic formula $A(F)$

over $\underbrace{\mathbb{Z}_2}_{\text{mod } 2}$

$$T \iff 1$$

$$F \iff 0$$

$$X_i \iff x_i$$

$$\overline{X_i} \iff 1 - x_i$$

$$\alpha \wedge \beta \iff \alpha \cdot \beta$$

$$\alpha \vee \beta \iff 1 - (1-\alpha)(1-\beta)$$

$$\alpha \vee \beta \vee \gamma \iff 1 - (1-\alpha)(1-\beta)(1-\gamma)$$

<u>example</u> $x_1 \vee \bar{x}_2 \vee x_3 \iff 1 - (1-x_1)(x_2)(1-x_3)$

<u>Key point</u>: F satisfied by a

iff

$$A(F)(a) = 1$$

issue 1: we like low degree polys. $A(F)$ can have very high degree

idea1: lets deal with each clause separately with some a

issue 2: we are good at 0-testing, not 1-testing!

idea2: consider complement

Vector of clause arithmetizations:

Let $\mathcal{C}(x) = (\hat{c}_1(x), \hat{c}_2(x), \dots)$

s.t. $\hat{c}_i$ = complement of arithmetization of clause C_i

$\implies$ evaluates to 0 if X satisfies C_i

$\implies \mathcal{C}(x) = (0, \dots, 0)$ if X satisfies F

Observe

(1) each $\hat{C}_i$ is deg ≤ 3 poly in x

(2) V knows coeffs of each $\hat{C}_i$

need to convince V that $\mathcal{C}(a) = (\hat{c}_1(a), \hat{c}_2(a), \dots)$

$= (0, 0, \dots 0)$

without V reading all of a

Two big ideas

1) encoding of assignment
   - can access any linear fctn of it & any bit in consistent fast way

2) Ensure that lots of simultaneous good things happen
   - encode algebraically so that "0 ≡ good"
   - test vector ≡ (00...0)

Summary:

High level idea: Special encoding of assignment

Encode satisfiability of F as a collection of polys in vars of assignment

- one for each clause
- eval to 0 if assignment satisfies clause
- low degree
- V knows coeffs - depend on structure of clause + vars of clause

Note: we are only concerned that V is poly time, here will not be sublinear but queries to proof should be constant!

Idea for proof π

- proof contains $\mathcal{C}(a)\cdot r \quad \forall r \in \{0,1\}^n$
- if $\forall i,\ \hat{C}_i(a) = 0$, $\quad \Pr_r[\mathcal{C}(a)\cdot r = 0] = 1$

  if $\exists i$ s.t. $\hat{C}_i(a) \neq 0$, $\quad \Pr_r[\mathcal{C}(a)\cdot r = 0] = 1/2$

  $$= \Pr_r[\mathcal{C}(a)\cdot r = 1]$$

mod 2 arithmetic

recall the problem from before ...

proof could write all 0's even if $\mathcal{C}(a)\cdot r \neq 0$, so need to do more

What does $\mathcal{C}(a)\cdot r$ look like?

V doesn't know

$$\sum_i r_i \cdot \hat{C}_i(a) = \Gamma + \sum_i a_i \alpha_i + \sum_{i,j} a_i \cdot a_j \beta_{ij} + \sum_{i,j,k} a_i a_j a_k \gamma_{ijk} \pmod 2$$

V does know: depends on r_i's + coeffs in $\hat{C}_i$

From here on:

$$\left.\begin{aligned} \alpha_i &\to x_i \\ \beta_{ij} &\to y_{ij} \\ \gamma_{ijk} &\to z_{ijk} \end{aligned}\right\} \text{no relation to vars of 3SAT}$$

example

$G = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_2)$

evaluates to 1 if C_1 satisfied

$A(C_1) = 1 - (1-x_1)(1-x_2) = x_1 + x_2 - x_1 x_2$

$\Rightarrow \mathcal{C}_1(A) = 1 - a_1 - a_2 + a_1 a_2$

"$1 - A(C_1)$" evaluates to 0 if C_1 satisfied

$A(C_2) = 1 - (x_1)(1-x_2) = 1 - x_1 + x_1 x_2$

$\Rightarrow \mathcal{C}_2(A) = a_1 - a_1 a_2$

$$\sum r_i \mathcal{C}_i(a) = r_1(1 - a_1 - a_2 + a_1 a_2) + r_2(a_1 - a_1 a_2)$$

$$= \underbrace{r_1 \cdot 1 + r_2 \cdot 0}_{\text{deg } 0} + \underbrace{(-r_1 + r_2) \cdot a_1 + (-r_1) \cdot a_2}_{\text{deg } 1} + \underbrace{(r_1 - r_2) a_1 a_2}_{\text{deg } 2}$$

r_1 r_2	$\sum r_i \mathcal{C}_i(a)$	sat case $a^+ = (0,1)$	unsat case $a^- = (0,0)$
0 0	0	0	0
0 1	$a_1 - a_1 a_2$	0	0
1 0	$1 - a_1 - a_2 + a_1 a_2$	$1-0-1+0=0$	$1-0-0+0=1$
1 1	$1 - a_2$	$1-1=0$	$1-0=1$

High level idea for proof: Special encoding of assignment

- proof writes out

$\boxed{\begin{array}{l} \underline{\text{all}} \quad \text{linear fctns of assignment} \\ \quad \text{deg } 2 \\ \quad \text{deg } 3 \end{array}}$

from this, V can compute whichentries it needs for previous test

- $A_a(x) = a \cdot x$

$B_a(y) = (a \circ a)^T \cdot y$

$C_a(z) = (a \circ a \circ a)^T \cdot z$

all linear fctns

$\Rightarrow$ can property test linearity & self-correct

Tester: • makes sure A_a, B_a, C_a are close to linear
• uses self-corrected version of them to <u>always</u> get linear fctn.

<u>But</u> (1) what if they come from different assignments a, a' & a''?
(2) how do we know a is satisfying?

More details:

A = all linear fctns evaluated at assignment a $\qquad A : \mathbb{F}_2^n \to \mathbb{F}_2 \qquad A(x) = \sum a_i x_i = a^T \cdot x$

B = all degree 2 fctns evaluated at a $\qquad B : \mathbb{F}_2^{n^2} \to \mathbb{F}_2 \qquad B(y) = \sum_{i,j} a_i a_j y_{ij} = (a \circ a)^T \cdot y$

C = all degree 3 fctns evaluated at a $\qquad C : \mathbb{F}_2^{n^3} \to \mathbb{F}_2 \qquad C(z) = \sum_{i,j,k} a_i a_j a_k z_{ijk} = (a \circ a \circ a)^T \cdot z$

Proof π:

complete input/output tables of $\underbrace{\tilde{A}, \tilde{B}, \tilde{C}}_{\text{hopefully } A, B, C \text{ but need to check}}$

we only care about one row:

$x = \alpha, \quad y = \beta, \quad z = \gamma \quad \leftarrow$ chosen from r_i's coeffs of C_i's

but extra info helps us check consistency

What does verifier check in proof?

(1) $\tilde{A}, \tilde{B}, \tilde{C}$ in "right form"

- all are linear fctns

can only test ε-linear <u>but</u> can self-correct to access true linear fctn

- correspond to same assignment a

i.e. $\tilde{A}(x) = a^T x \Rightarrow \tilde{B}(y) = (a \circ a)^T \cdot y \Rightarrow \tilde{C}(z) = (a \circ a \circ a)^T z$

Test consistency of self-corrections

(2) a is satisfying assignment

(Check entries in encoding which give value of $\mathcal{C}(a)$)

- all $\hat{C}_i$'s evaluate to 0 on a

Testing 1:

(1) $\tilde{A}, \tilde{B}, \tilde{C}$ in "right form"

- all are linear fctns

  can only test ε-linear but can self-correct to access true linear fctn

- correspond to same assignment a

  i.e. $\tilde{A}(x) = a^T x \Rightarrow \tilde{B}(y) = (a \circ a)^T \cdot y \Rightarrow \tilde{C}(z) = (a \circ a \circ a)^T z$

  Test consistency of self-corrections

- Use linearity test on $\tilde{A}, \tilde{B}, \tilde{C}$

  if linear pass

  if $1/8$-far, fail with prob $\geq 1 - \delta'$

  (pick δ' really small, but constant, so that all (δ')'s add up to $< \delta$)

  if pass $\Rightarrow$ $1/8$-linear $\Rightarrow$ can use with self-corrector to get linear fctns $SC\text{-}\tilde{A}$, $SC\text{-}\tilde{B}$, $SC\text{-}\tilde{C}$ from now on.

  $$SC\text{-}\tilde{A} = a^T \cdot x, \quad SC\text{-}\tilde{B} = b^T \cdot y, \quad SC\text{-}\tilde{C} = c^T \cdot z$$

- want that $SC\text{-}\tilde{A} = a^T \cdot x \Rightarrow SC\text{-}\tilde{B} = (a \circ a)^T \cdot y \Rightarrow SC\text{-}\tilde{C} = (a \circ a \circ a)^T z$

Test that $sc\text{-}\tilde{A} = a^T \cdot x \Rightarrow sc\text{-}\tilde{B} = (a \circ a)^T \cdot y \Rightarrow sc\text{-}\tilde{C} = (a \circ a \circ a)^T \cdot z$:

Outer Product Tester:

Pick random x_1, x_2, x, y

true if $b_{ij} = a_i \cdot a_j \; \forall i,j$

Test $sc\text{-}\tilde{A}(x_1) \cdot sc\text{-}\tilde{A}(x_2)$

$$= \left(\sum a_i x_{1i}\right) \cdot \left(\sum a_j x_{2j}\right)$$

holds due to ε-linear + self-corretor

$$= \sum_{i,j} a_i a_j x_{1i} x_{2j} = \sum_{i,j} b_{ij} x_{1i} x_{2j}$$

$$= sc\text{-}\tilde{B}(\underbrace{x_1 \circ x_2}_{\text{not uniformly distributed}})$$

Test $sc\text{-}\tilde{A}(x) \cdot sc\text{-}\tilde{B}(y)$

$$= \left(\sum a_i x_i\right) \cdot \left(\sum b_{jk} \cdot y_{jk}\right)$$

true if $a_i \cdot b_{jk} = c_{ijk}$ $\forall i,j,k$

$$= \sum_{ijk} (a_i \cdot b_{jk}) x_i y_{jk} = \sum a_i a_j a_k x_i y_{jk}$$

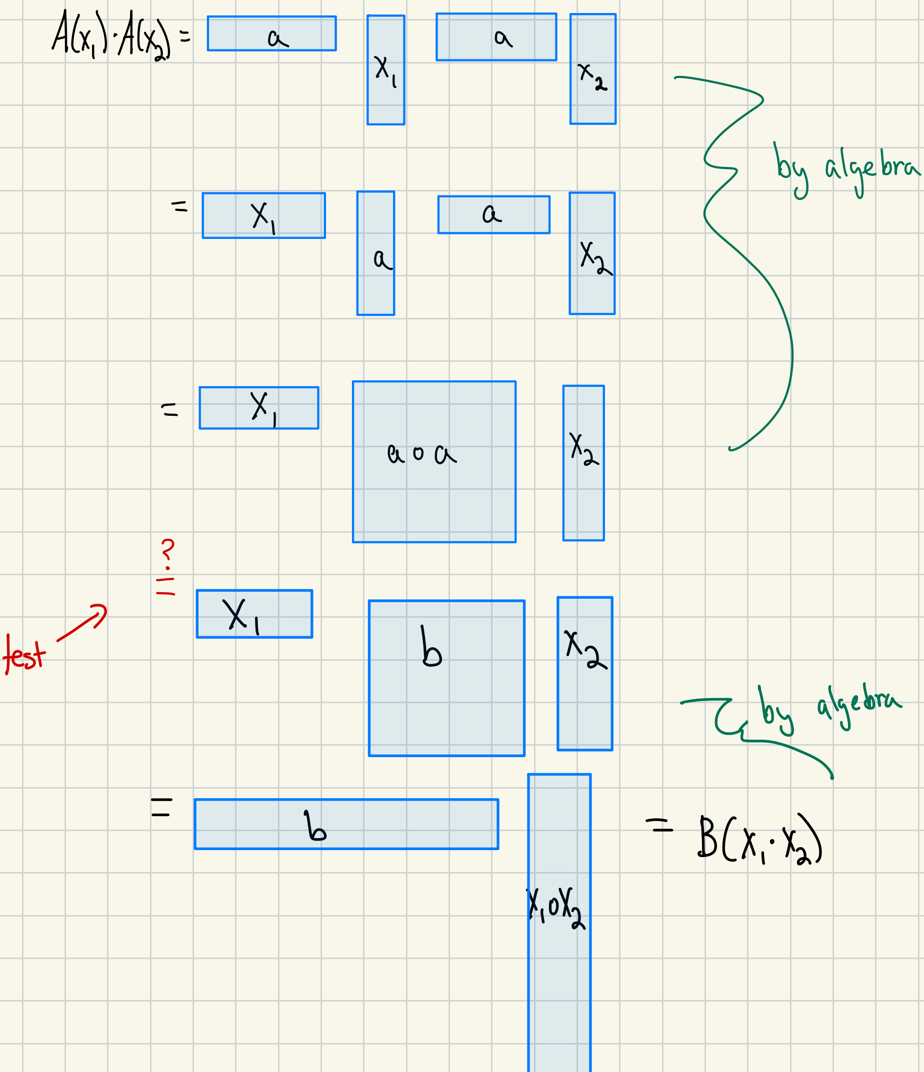
$$= sc\text{-}\tilde{C}(\underbrace{x \circ y}_{\text{not unif distributed}})$$

$\boxed{\text{we use self-corrector since not all queries uniform}}$

Analysis of outer-product tester:

if $b = a \circ a$, test passes (green equalities hold)

if $b \neq a \circ a$:



if $b \neq a \circ a$:

Fact $\Rightarrow$ $\Pr_{x_2}[(a \circ a) \cdot x_2 \neq b \cdot x_2] = 1/2$ (*)

if $(a \circ a) \cdot x_2 \neq b \cdot x_2$ then

Fact $\Rightarrow$ $\Pr_{x_1}[x_1 (a \circ a) x_2 \neq x_1 b x_2] = 1/2$ (**)

$\Rightarrow$ $\Pr[\text{fail test}] = \Pr[(*) \,\&\, (**) \text{ happen}]$

$\geq 1/4$

so test pass $\Rightarrow$ safe to assume $b = a \circ a$

Similar argument $\Rightarrow$ safe to assume $c = a \circ a \circ a$

Query Complexity for part 1

\# random bits $= O(n^3)$

\# queries to proof $= O(1)$

Testing 2:

(2) a is satisfying assignment

(check entries in encoding which give value of $\mathcal{C}(a)$)

- all $\hat{C}_i$'s evaluate to 0 on a

- call self-correctors $\Rightarrow$ recover linear fctns a, $a \circ a$, $a \circ a \circ a$
- a represents assignment, but we don't know if
- a satisfies formula $\Leftrightarrow$ $\mathcal{C}(a) = (\hat{C}_1(a), \hat{C}_2(a), \ldots)$
$= (0, 0, 0 \ldots, 0)$

Satisfiability Test:

Pick $r \in \mathbb{F}_2^n$

Compute Γ, α_i's, β_{ij}'s, γ_{ijk}'s

$\downarrow x_i$'s $\quad \downarrow y_{ij}$'s $\quad \downarrow z_{ijk}$'s

fctns of r + coeffs of deg 3 polys

query proof to get $SC\text{-}\tilde{A}(\alpha_1 \dots \alpha_n) = w_0$

$SC\text{-}\tilde{B}(\beta_{11} \cdots \beta_{nn}) = w_1$

$SC\text{-}\tilde{C}(\gamma_{111} \cdots \gamma_{nnn}) = w_2$

Verify $0 = \Gamma + w_0 + w_1 + w_2 \pmod 2$

hopefully means $\sum r_i \hat{C}_i(a) = 0$

Behavior of test:

if $\forall i \ \hat{C}_i(a) = 0$, $Pr[\text{pass}] = 1$ ✓

if $\exists i$ s.t. $\hat{C}_i(a) \neq 0$,

Fact $\Rightarrow Pr[\sum r_j \hat{C}_j(a) = 0] = 1/2$

So after k times, $Pr[\text{pass}] \leq 1/2^k$

\# random bits $= O(n)$

\# queries to proof $= O(1)$

Conclusion

PCP protocol for 3SAT in which verifier uses:

$O(n^3)$ random bits

& $O(1)$ queries