Abstract Models of Computation and Complexity Lower Bounds

Ueli Maurer

ETH Zurich, www.crypto.ethz.ch

Number Theory and Computational Cryptography
Sept. 29 - Oct. 2, 2010, Warsaw.

Discrete logarithm (DL) problem

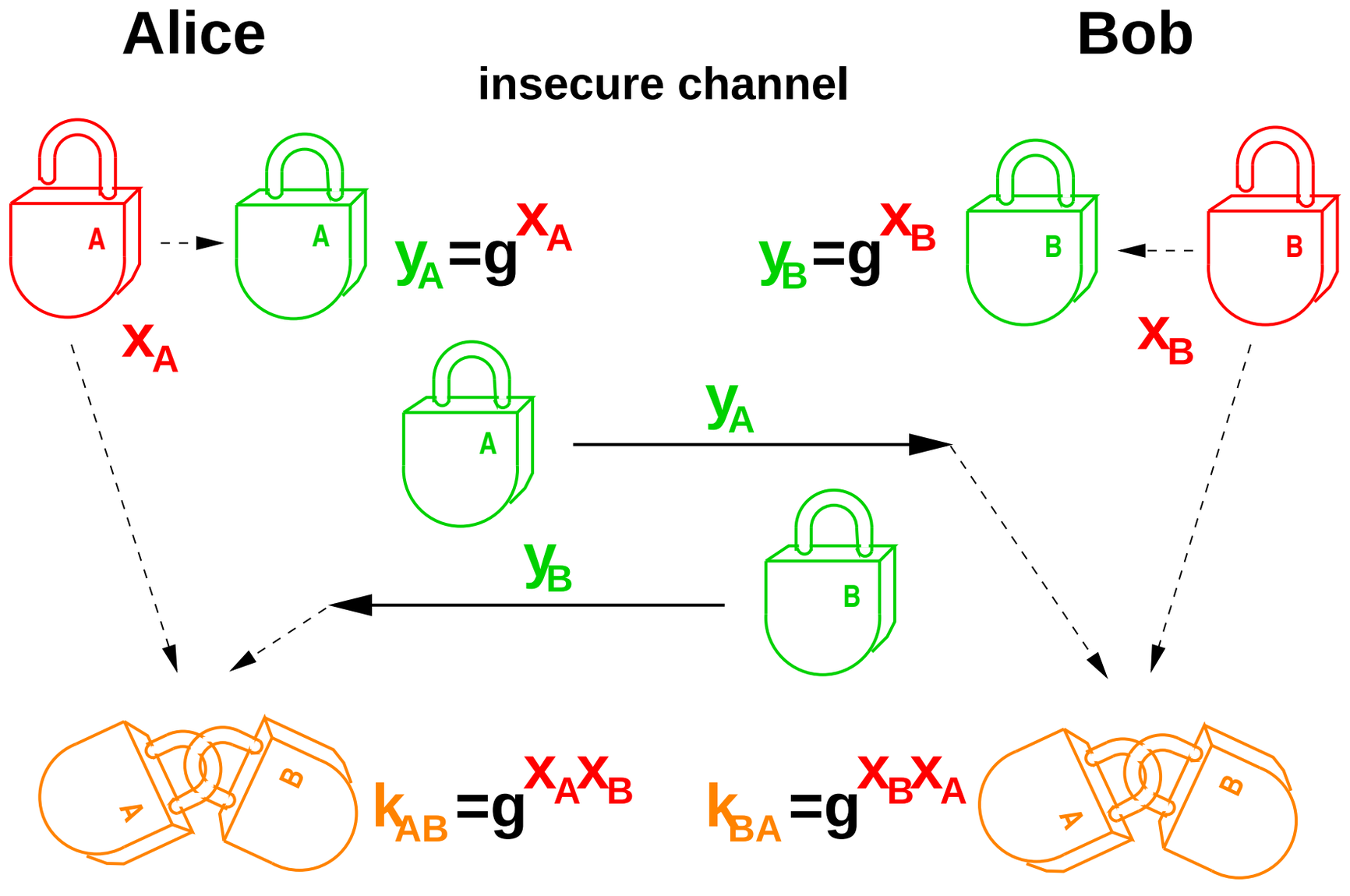
Cyclic group G of order n with generator g:

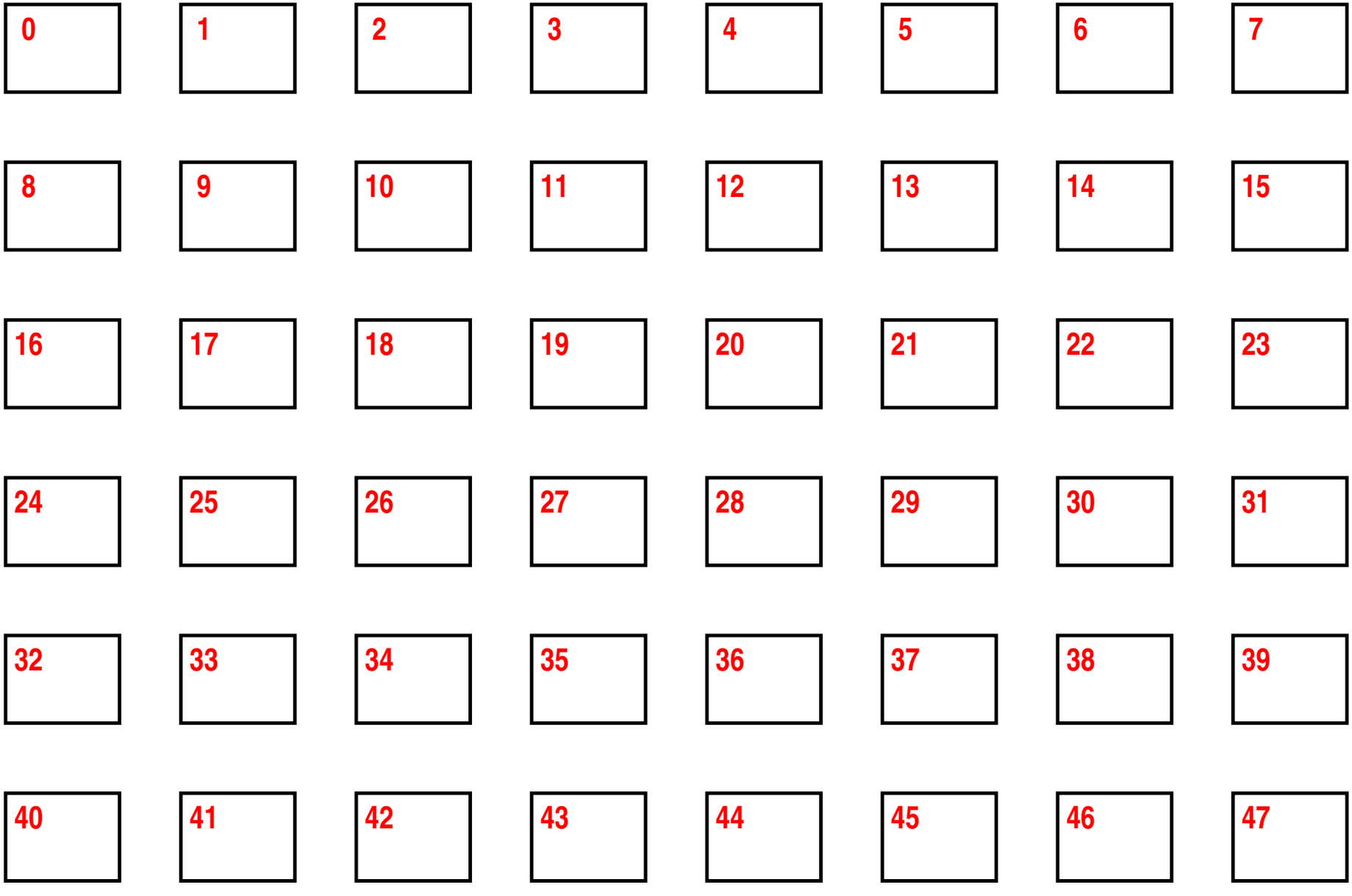
$$\mathbf{G} = \langle \mathbf{g} \rangle = \{\mathbf{g}^i : 0 \leq i < n\}$$

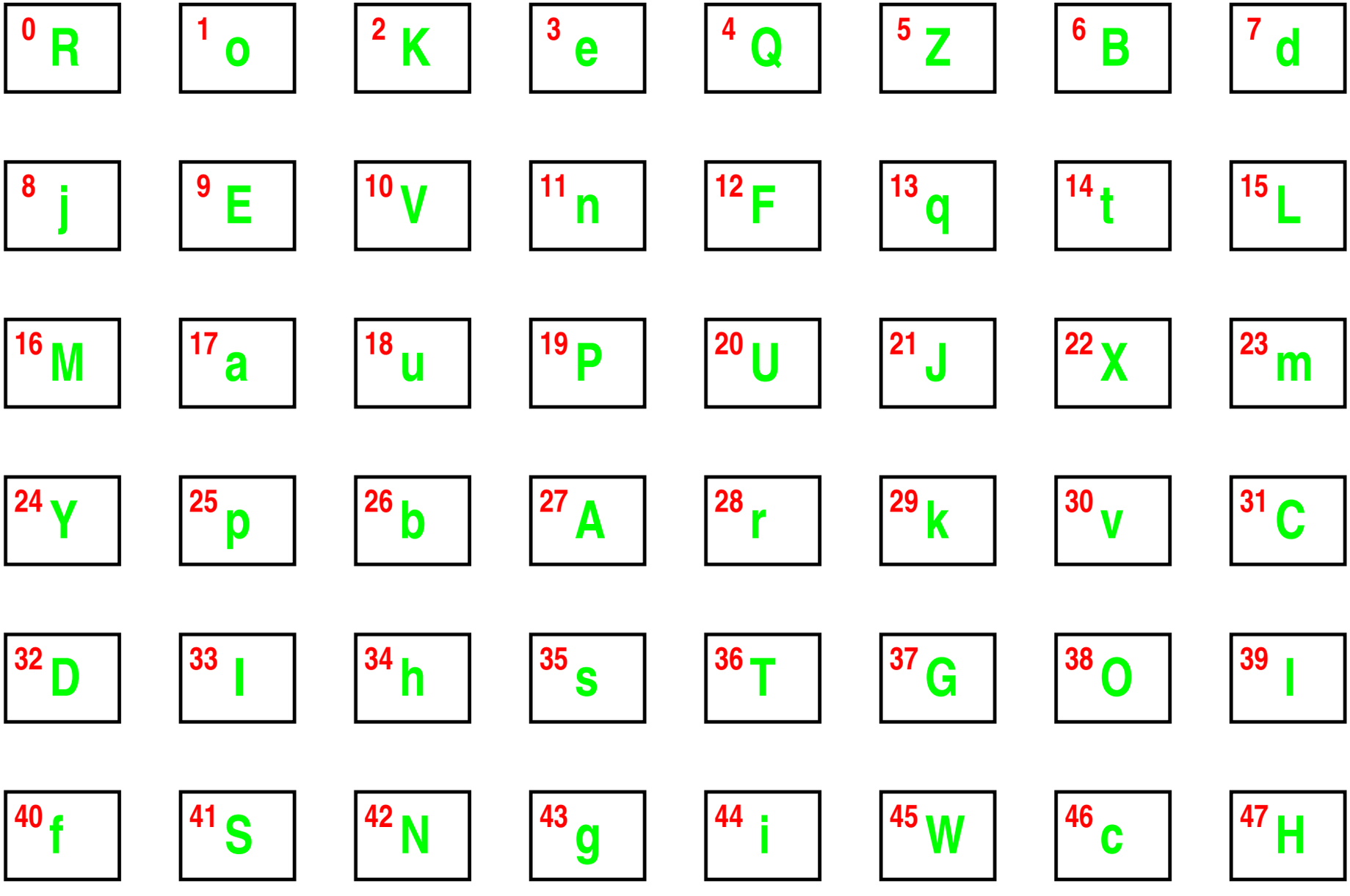
DL problem: Given a $\in$ G, find x such that a $=$ $\mathbf{g}^{\mathbf{x}}$.

$\mathbf{x} \longrightarrow \mathbf{g}^{\mathbf{x}}$ is for many groups believed to be a OWF.

Diffie-Hellman protocol: mechanical analog




0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47


0 R
1 o
2 K
3 e
4 Q
5 Z
6 B
7 d
8 j
9 E
10 V
11 n
12 F
13 q
14 t
15 L
16 M
17 a
18 u
19 P
20 U
21 J
22 X
23 m
24 Y
25 p
26 b
27 A
28 r
29 k
30 v
31 C
32 D
33 I
34 h
35 s
36 T
37 G
38 O
39 l
40 f
41 S
42 N
43 g
44 i
45 W
46 c
47 H

Possible operations on indices

Possible operations on indices

- **no operation: expected $n/2$ queries**

Possible operations on indices

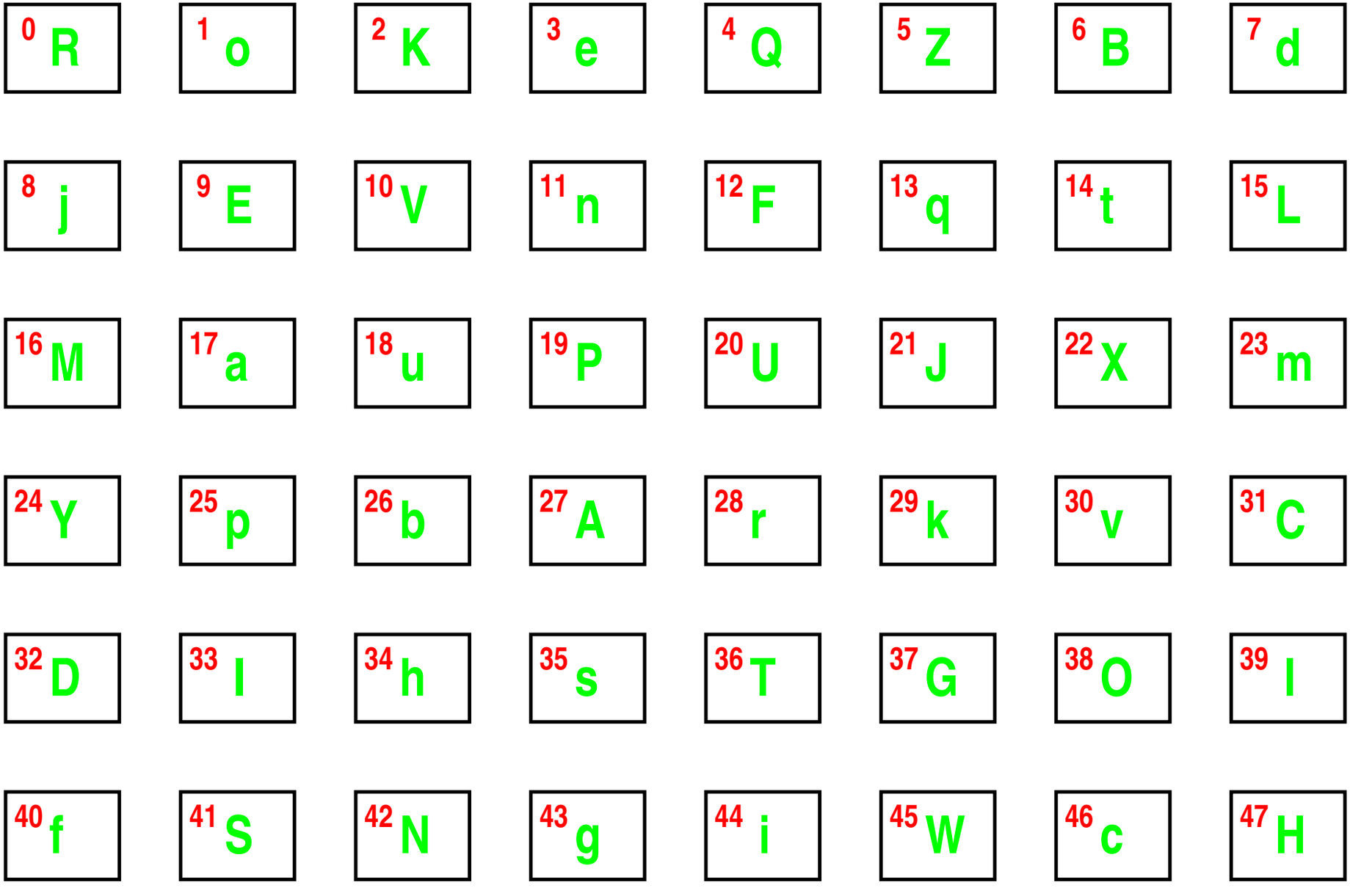
- **no operation: expected $n/2$ queries**
- **double index:**

Possible operations on indices

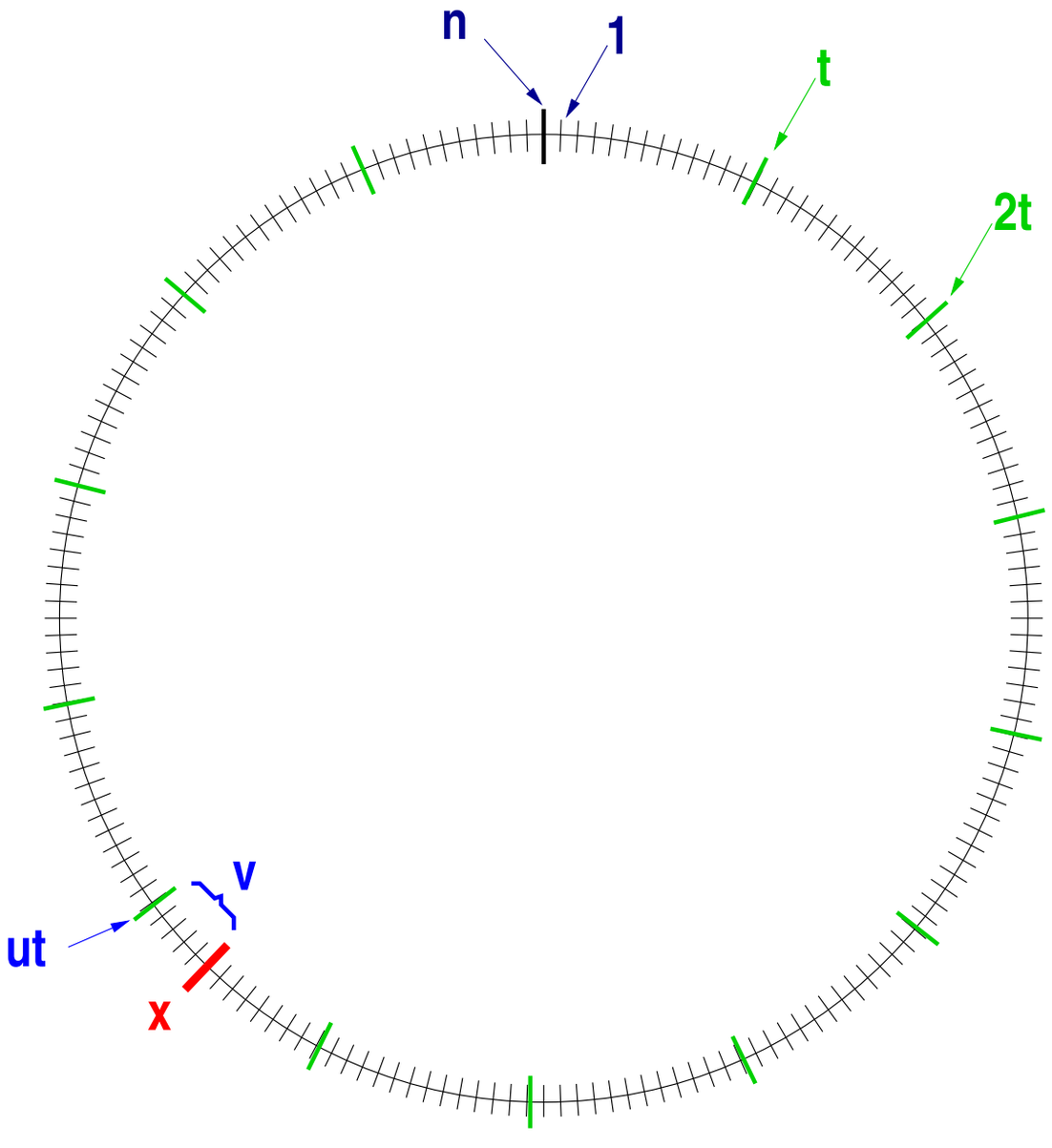
- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:**


0 R
1 o
2 K
3 e
4 Q
5 Z
6 B
7 d
8 j
9 E
10 V
11 n
12 F
13 q
14 t
15 L
16 M
17 a
18 u
19 P
20 U
21 J
22 X
23 m
24 Y
25 p
26 b
27 A
28 r
29 k
30 v
31 C
32 D
33 I
34 h
35 s
36 T
37 G
38 O
39 l
40 f
41 S
42 N
43 g
44 i
45 W
46 c
47 H

Baby-step giant-step DL algorithm



Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$

Possible operations on indices

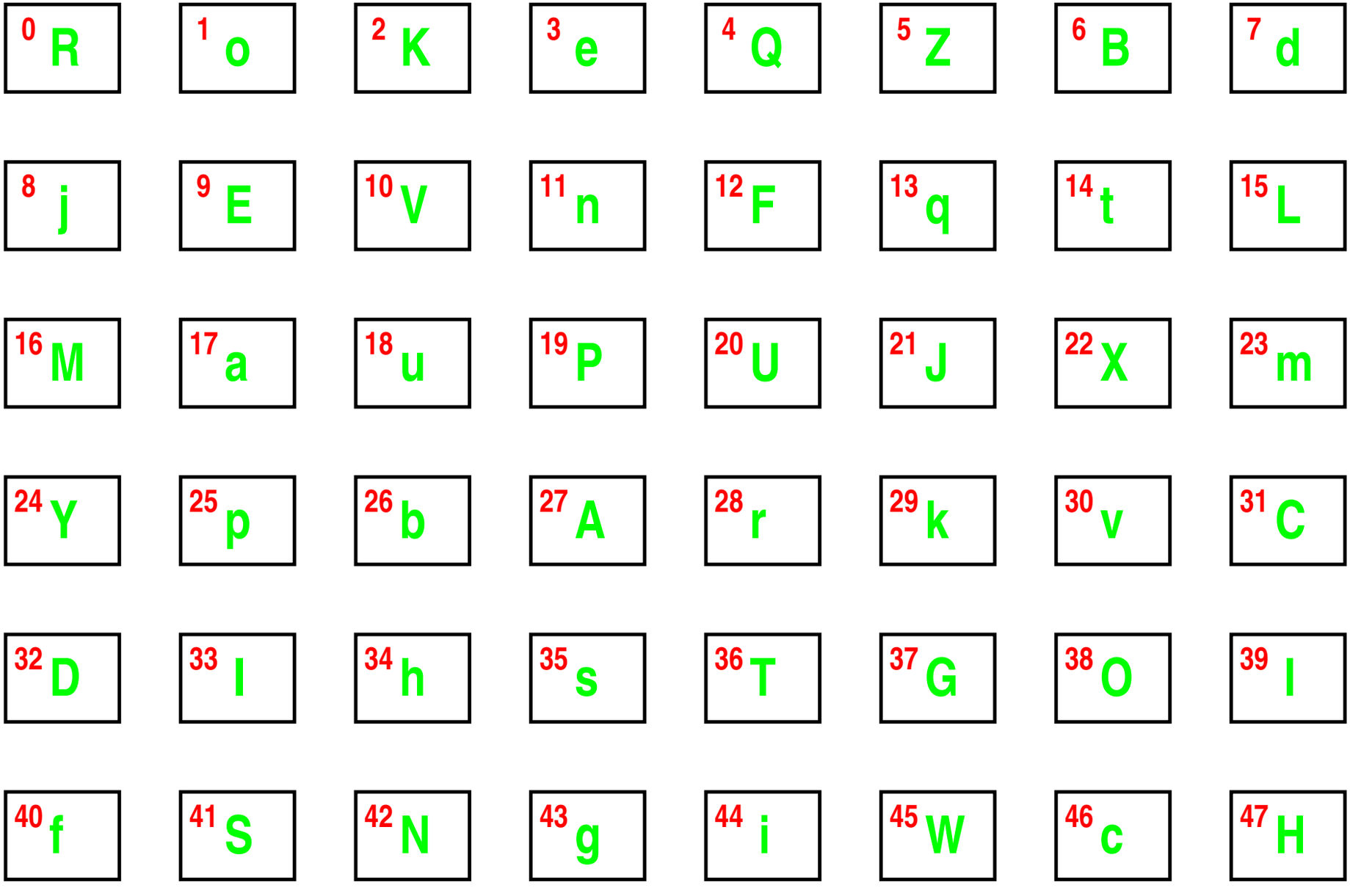
- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:**

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:** $O(\sqrt{n})$

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:** $O(\sqrt{n})$
- **add indices (mod** n**):**


0 R
1 o
2 K
3 e
4 Q
5 Z
6 B
7 d
8 j
9 E
10 V
11 n
12 F
13 q
14 t
15 L
16 M
17 a
18 u
19 P
20 U
21 J
22 X
23 m
24 Y
25 p
26 b
27 A
28 r
29 k
30 v
31 C
32 D
33 I
34 h
35 s
36 T
37 G
38 O
39 l
40 f
41 S
42 N
43 g
44 i
45 W
46 c
47 H

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:** $O(\sqrt{n})$
- **add indices (mod** n**): can be much faster!**

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:** $O(\sqrt{n})$
- **add indices (mod** n**): can be much faster!**
- **add and multiply indices:**

Possible operations on indices

- **no operation: expected** $n/2$ **queries**
- **double index:** $O(n)$
- **increment index:** $O(\sqrt{n})$
- **XOR indices:** $O(\sqrt{n})$
- **add indices (mod** n**): can be much faster!**
- **add and multiply indices:** $(\log n)^c$

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \to x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \to g^{xy}$
- **Squaring (SCDH):** $g^x \to g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \to z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \to z \stackrel{?}{=} x^2$

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \rightarrow x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \rightarrow g^{xy}$
- **Squaring (SCDH):** $g^x \rightarrow g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \rightarrow z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \rightarrow z \stackrel{?}{=} x^2$

Questions:

- **Can DL be reduced to CDH?**
- **Can CDH be reduced to DDH?**
- **Can CDH be reduced to SCDH?**
- **Can DDH be reduced to SDDH?**

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \rightarrow x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \rightarrow g^{xy}$
- **Squaring (SCDH):** $g^x \rightarrow g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \rightarrow z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \rightarrow z \stackrel{?}{=} x^2$

Questions:

- **Can DL be reduced to CDH?** **(yes)**
- **Can CDH be reduced to DDH?**
- **Can CDH be reduced to SCDH?**
- **Can DDH be reduced to SDDH?**

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \rightarrow x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \rightarrow g^{xy}$
- **Squaring (SCDH):** $g^x \rightarrow g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \rightarrow z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \rightarrow z \stackrel{?}{=} x^2$

Questions:

- **Can DL be reduced to CDH?** **(yes)**
- **Can CDH be reduced to DDH?** **no**
- **Can CDH be reduced to SCDH?**
- **Can DDH be reduced to SDDH?**

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \rightarrow x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \rightarrow g^{xy}$
- **Squaring (SCDH):** $g^x \rightarrow g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \rightarrow z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \rightarrow z \stackrel{?}{=} x^2$

Questions:

- **Can DL be reduced to CDH?** **(yes)**
- **Can CDH be reduced to DDH?** **no**
- **Can CDH be reduced to SCDH?** **yes**
- **Can DDH be reduced to SDDH?**

Computational problems for a cyclic group

- **Discrete logarithm (DL):** $g^x \rightarrow x$
- **Comput. Diffie-Hellman (CDH):** $g^x, g^y \rightarrow g^{xy}$
- **Squaring (SCDH):** $g^x \rightarrow g^{x^2}$
- **Decisional Diffie-H. (DDH):** $g^x, g^y, g^z \rightarrow z \stackrel{?}{=} xy$
- **Decisional squaring (SDDH):** $g^x, g^z \rightarrow z \stackrel{?}{=} x^2$

Questions:

- **Can DL be reduced to CDH?** **(yes)**
- **Can CDH be reduced to DDH?** **no**
- **Can CDH be reduced to SCDH?** **yes**
- **Can DDH be reduced to SDDH?** **no**

Goals

- **Propose an abstract model of computation**
- **Capture reasonable restrictions on an algorithm's power**
- **Describe generic algorithms as special case, in a new and simpler model**
- **Derive lower bounds on computational problems and reductions**

Generic algorithms

Generic algorithms

- We need a representation of elements as bitstrings

Generic algorithms

- We need a representation of elements as bitstrings
- **Generic algorithm:** cannot exploit representation, except for trivial properties

Generic algorithms

- **We need a representation of elements as bitstrings**
- **Generic algorithm: cannot exploit representation, except for trivial properties**
- **Note: GA's apply to any group.**

  Many known algorithms are generic.

Generic algorithms

- We need a representation of elements as bitstrings
- **Generic algorithm:** cannot exploit representation, except for trivial properties
- Note: GA's apply to any group.

  Many known algorithms are generic.
- Modeled as a **random mapping** to some set of bitstrings (Shoup)

  Here: simpler model

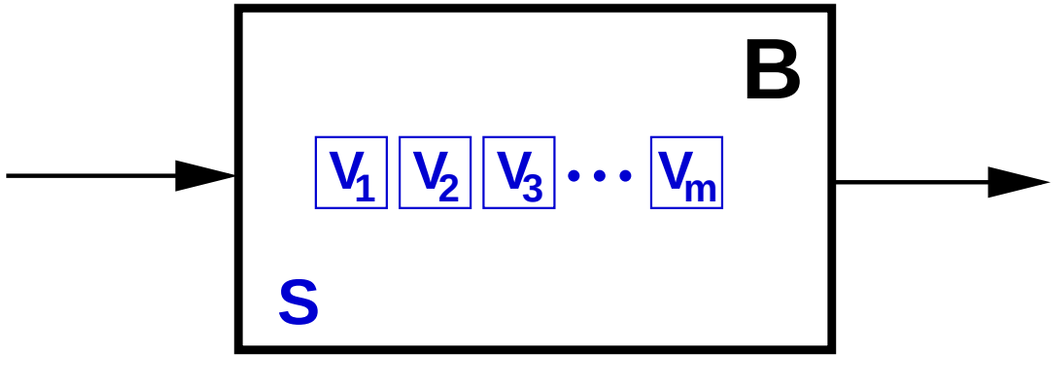
Generic algorithms

- We need a representation of elements as bitstrings
- **Generic algorithm:** cannot exploit representation, except for trivial properties
- Note: GA's apply to any group.

  Many known algorithms are generic.
- Modeled as a **random mapping** to some set of bitstrings (Shoup)

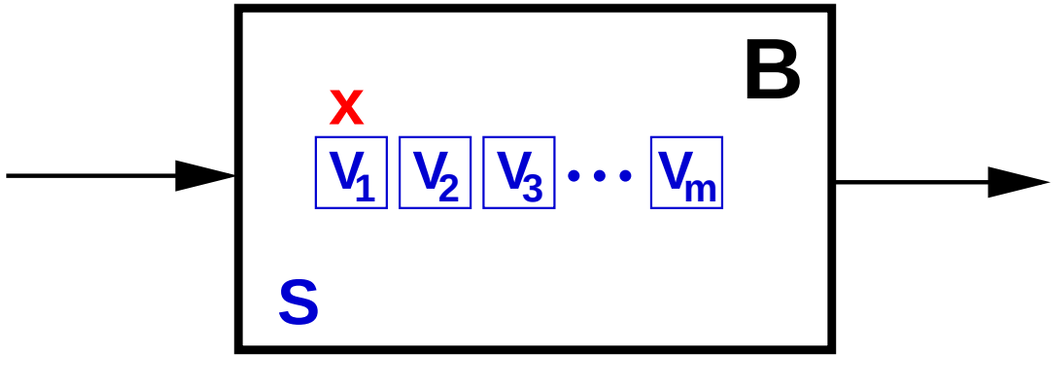
  Here: simpler model
- Goal: Prove lower bounds for DL etc.

Modeling generic algorithms



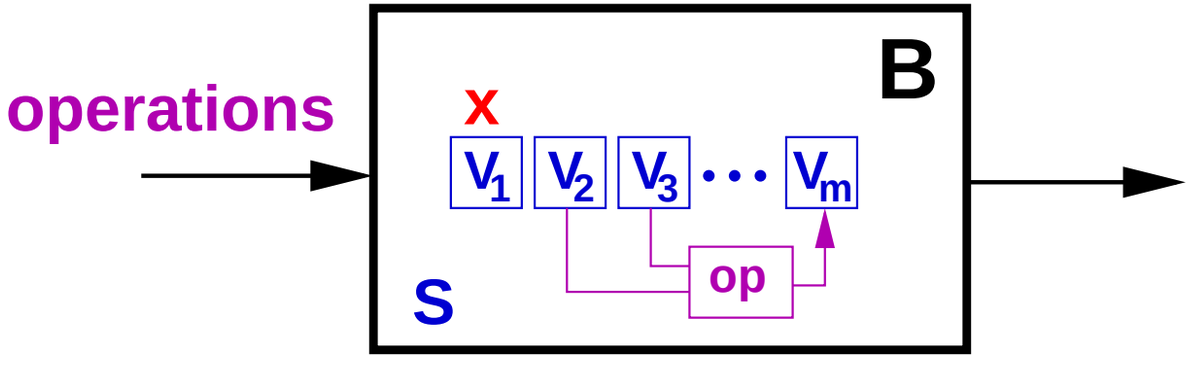
- **Blackbox B contains registers $V_1, \ldots, V_m$**

Modeling generic algorithms



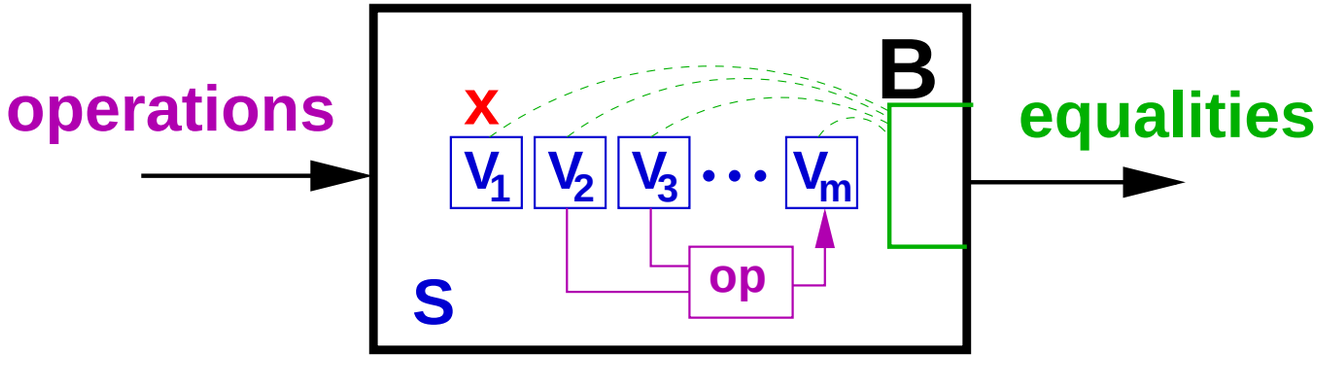
- Blackbox B contains registers $V_1, \ldots, V_m$ with $V_1 = x$

Modeling generic algorithms



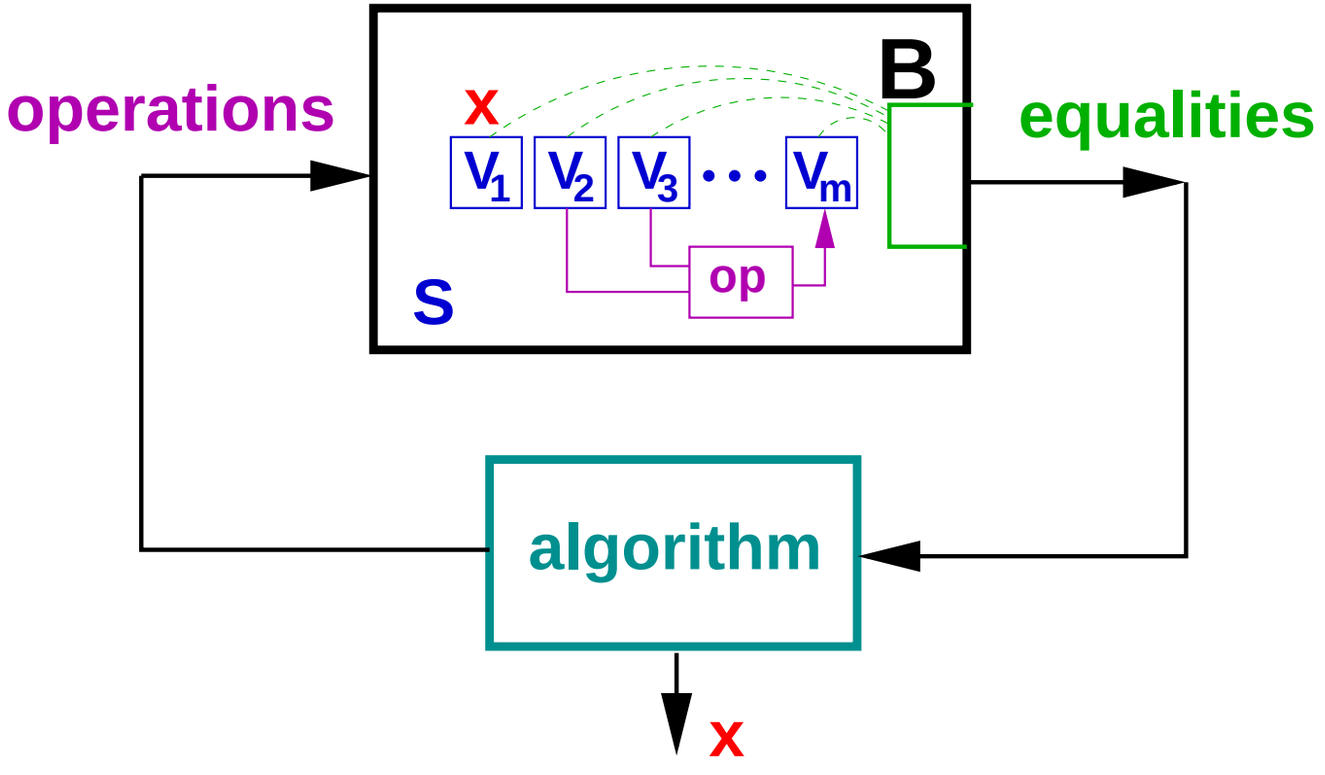
- Blackbox B contains registers $V_1, \ldots, V_m$ with $V_1 = x$
- B can perform internal operations operations.

Modeling generic algorithms



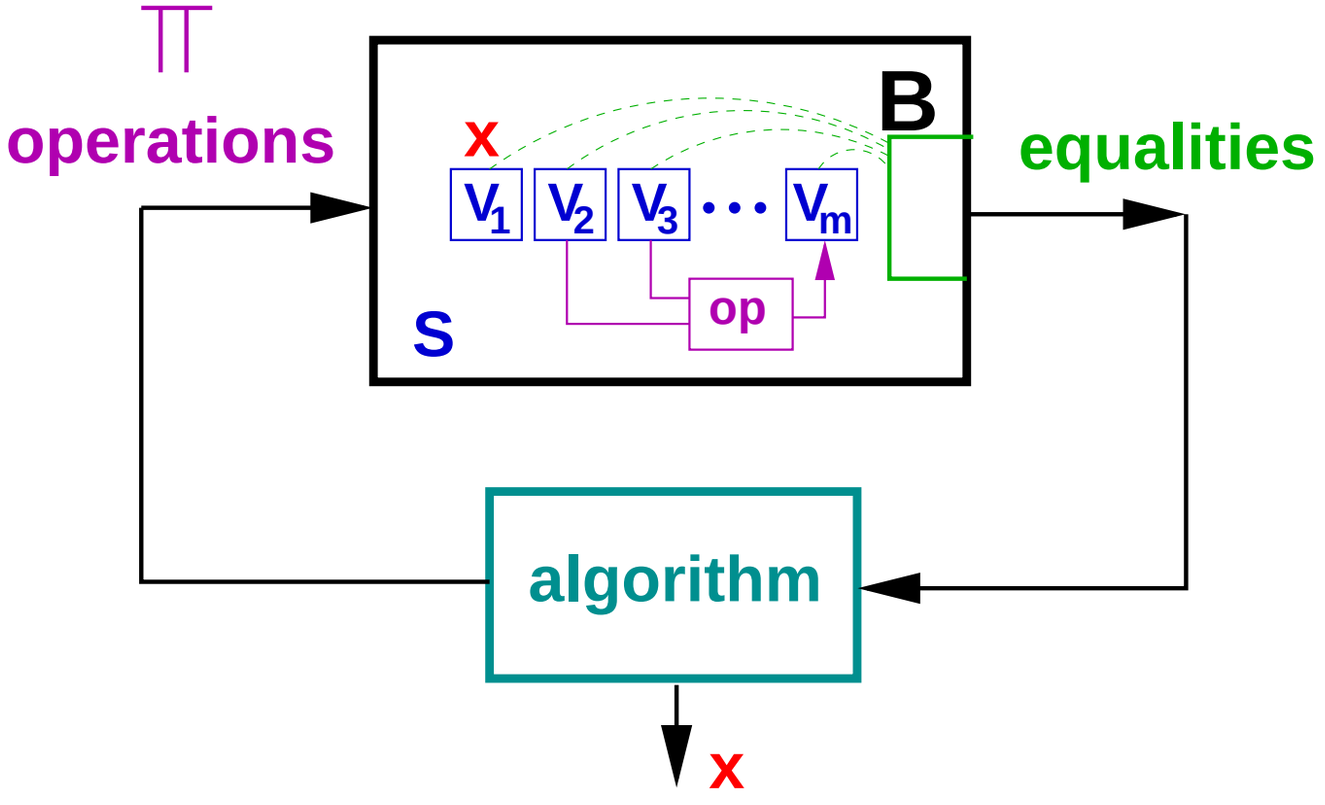
- Blackbox B contains registers $V_1, \ldots, V_m$ with $V_1 = x$
- B can perform internal operations operations.
- B yields as output at which indices equality occurs.

Modeling generic algorithms



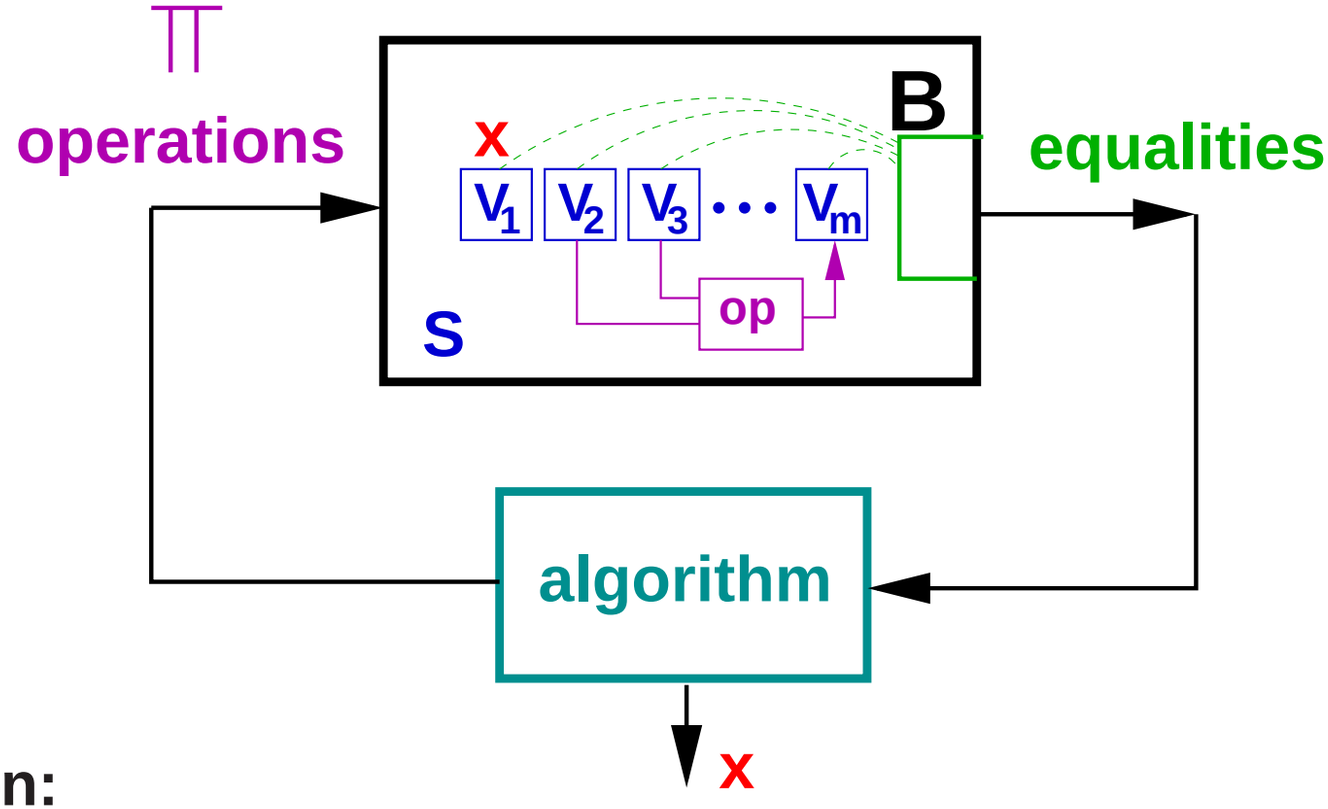
- Blackbox B contains registers $V_1, \ldots, V_m$ with $V_1 = x$
- B can perform internal operations operations.
- B yields as output at which indices equality occurs.
- Task of a generic algorithm: compute x.

Modeling generic algorithms



- Blackbox B contains registers $V_1, \ldots, V_m$ with $V_1 = x$
- B can perform internal operations operations.
- B yields as output at which indices equality occurs.
- Task of a generic algorithm: compute x.

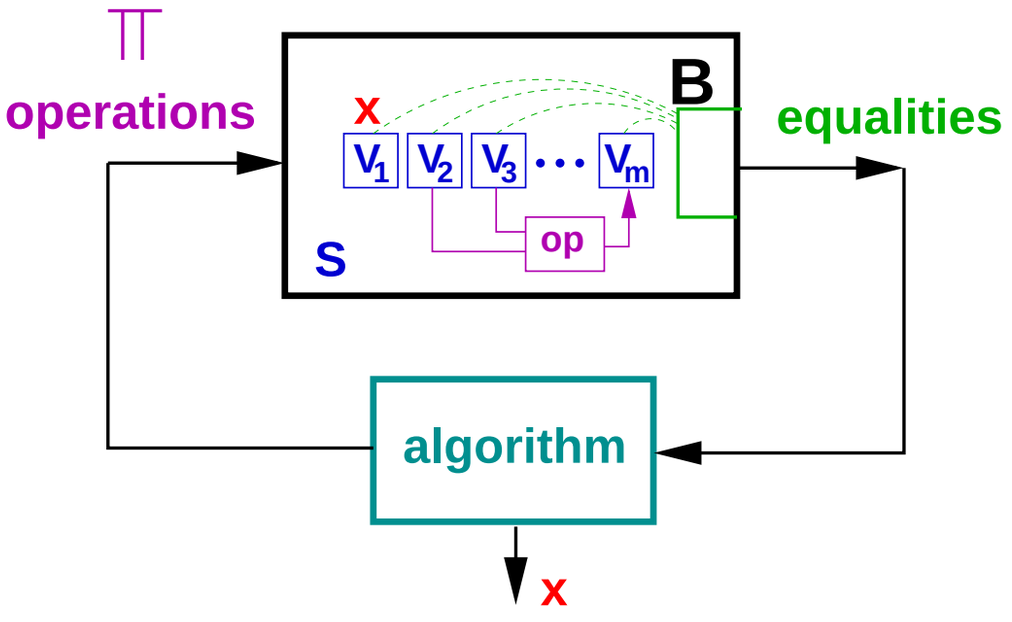
Modeling generic algorithms



Notation:

- $|\mathbf{S}| = n$
- $\mathcal{C}$ = set of constant functions
- $k = \#$ queries
- $\mathbf{p_S}$ = success probability
- $\overline{\Pi}$ = set of functions computable by Π

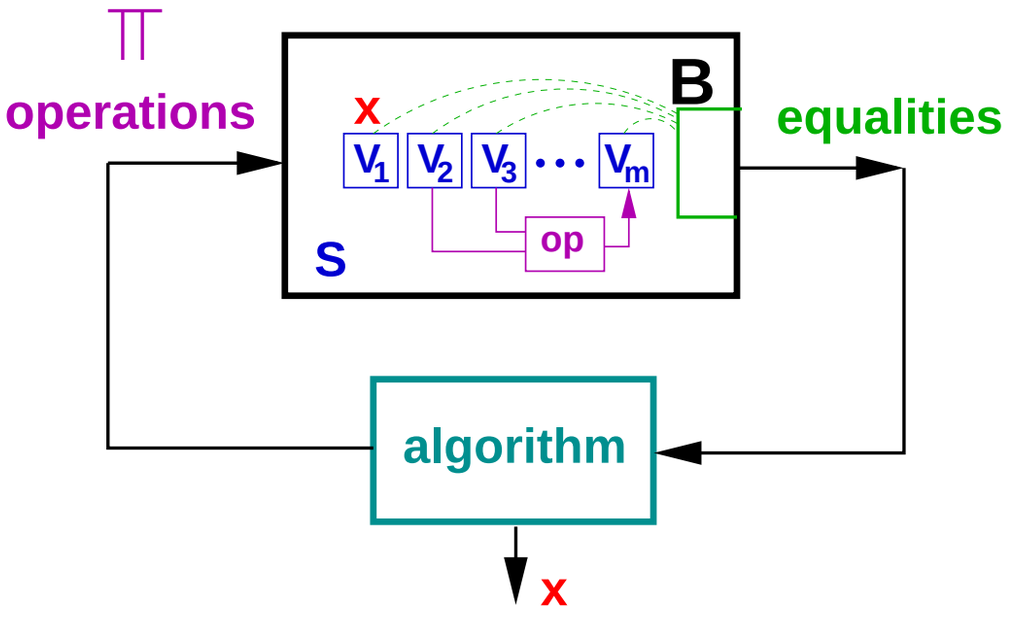
Example: no structure assumed



S is general

$\Pi = \mathcal{C}$

Example: no structure assumed

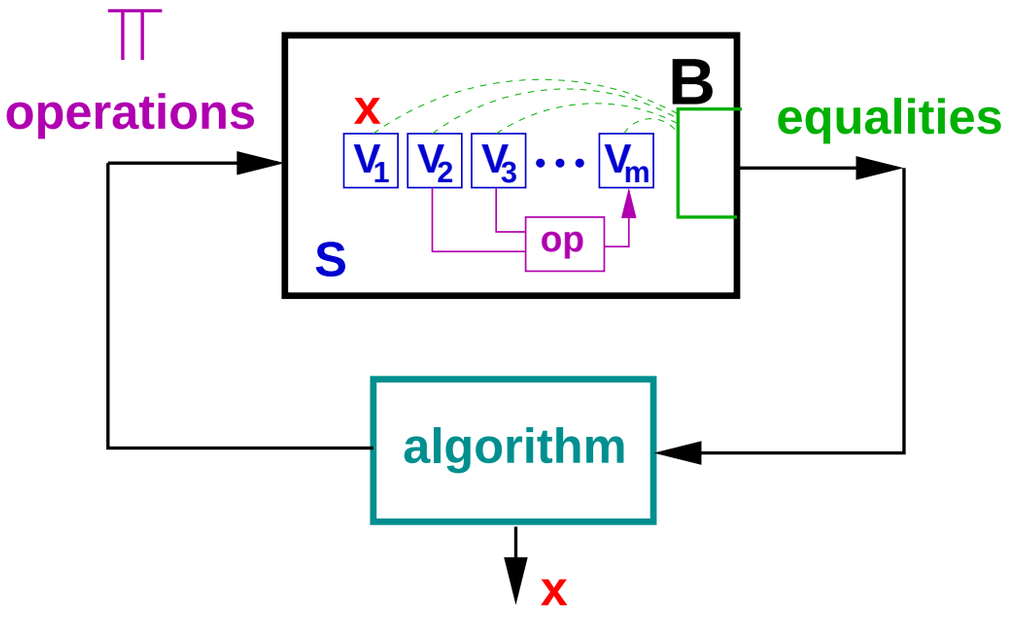


S is general

$\Pi = \mathcal{C}$

Proposition: $p_S \leq k/n \quad \Rightarrow O(n)$ **lower bound**

Example: no structure assumed



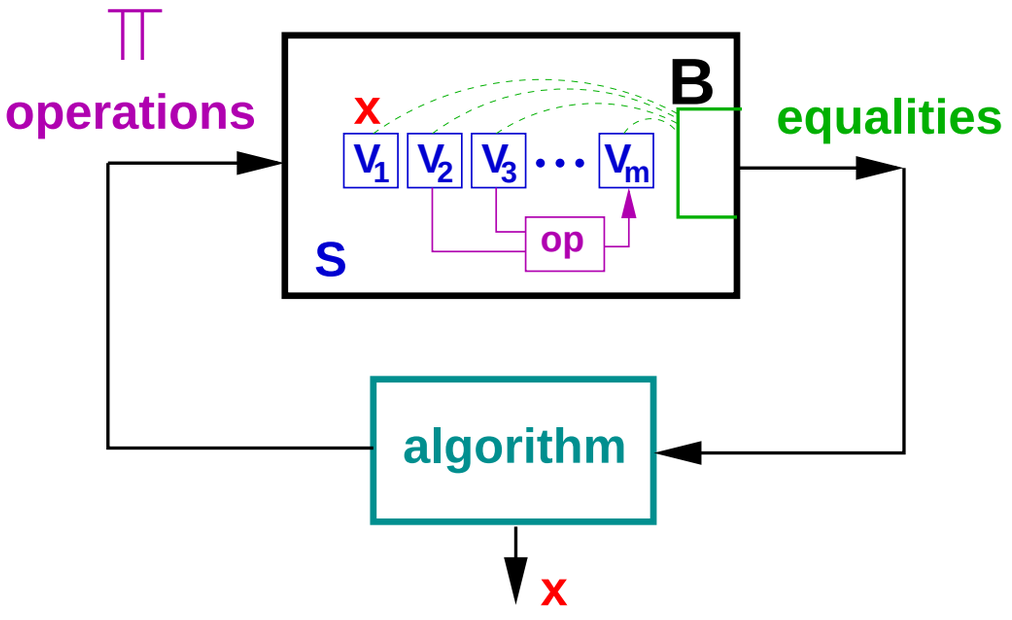
S is general

$\Pi = \mathcal{C}$

Proposition: $p_S \leq k/n \quad \Rightarrow O(n)$ **lower bound**

Proof: obvious?

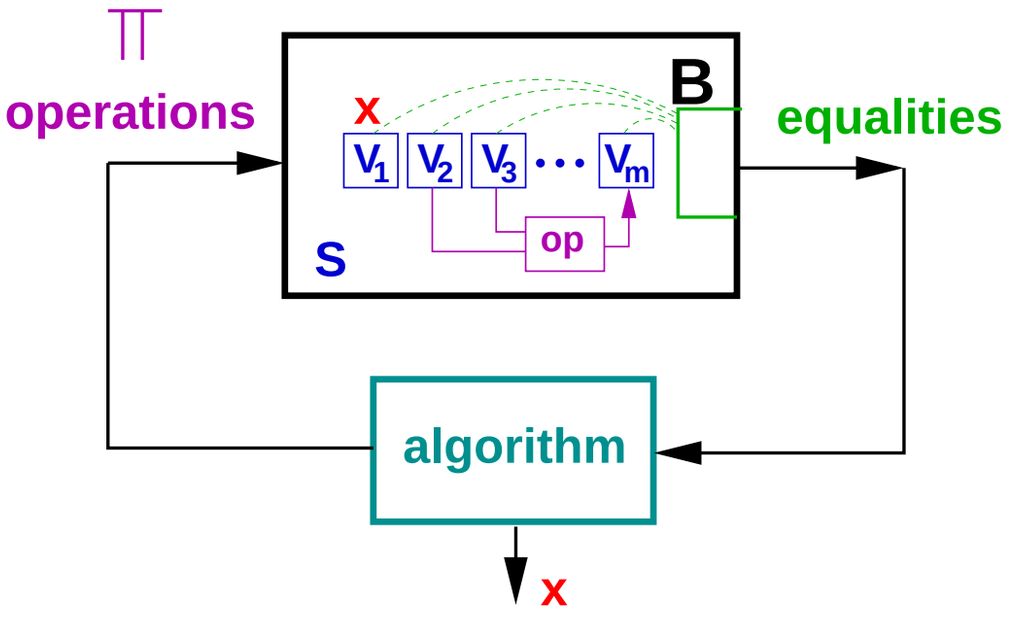
Group action



S is group

$$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$$

Group action

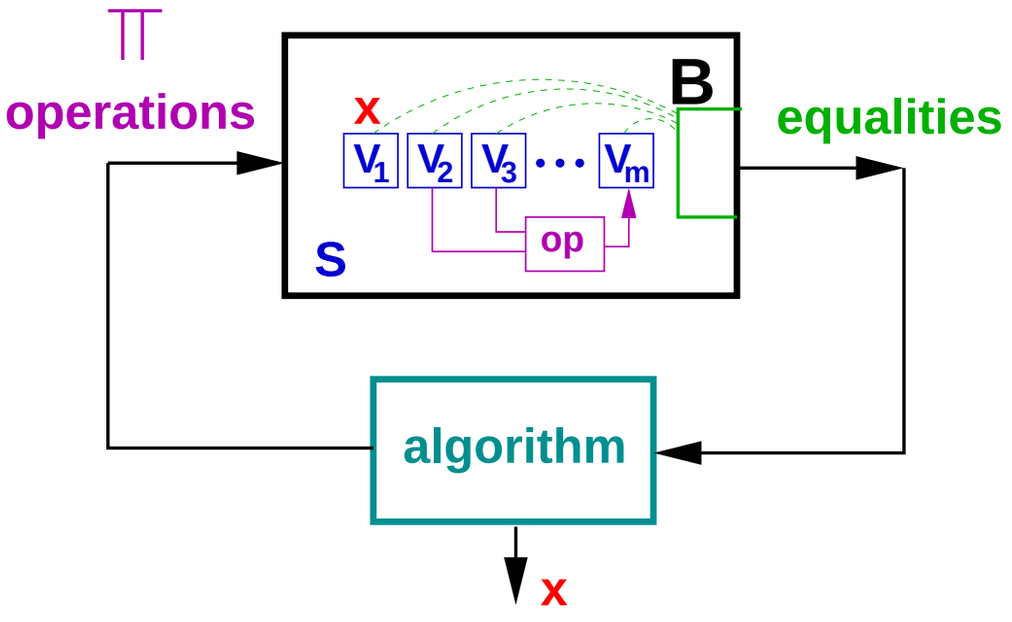


S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Group action



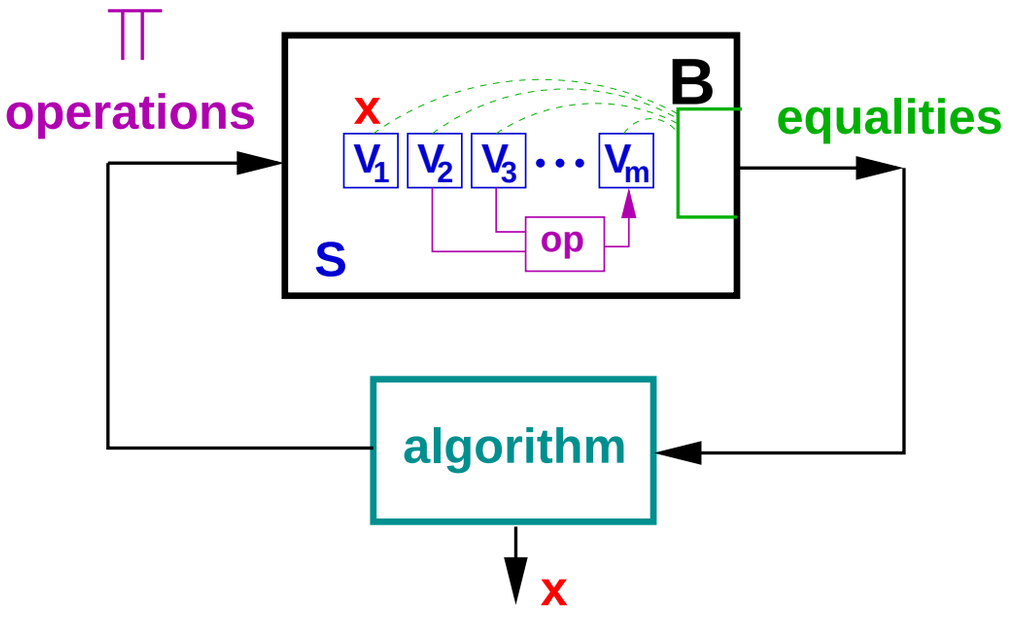
S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof: **1. Assume: First collision yields x.**

Group action



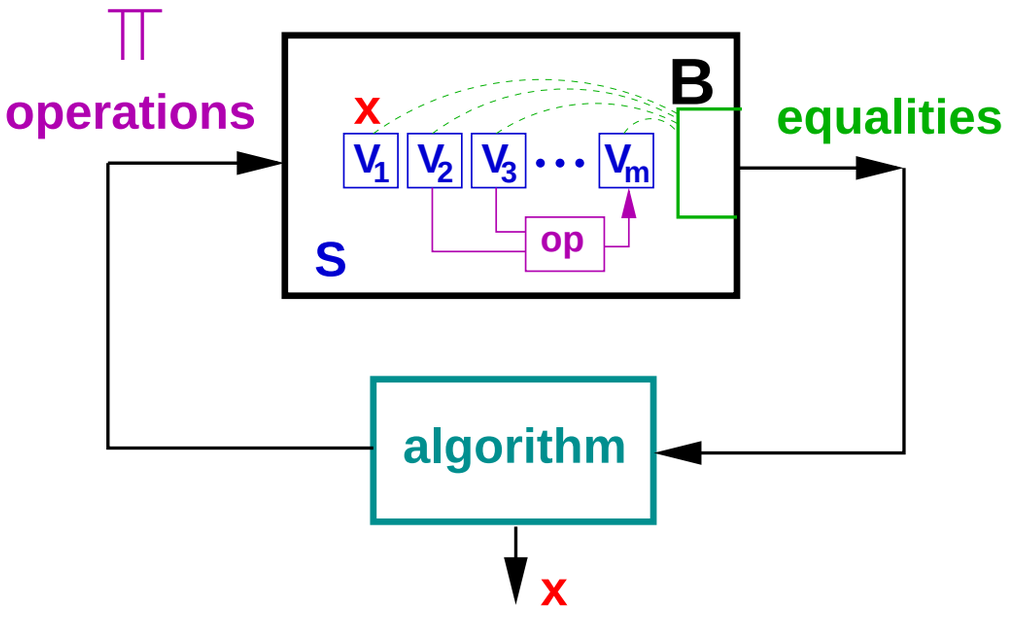
S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof:
1. **Assume: First collision yields x.**
2. **Adaptiveness of no help in provoking first collision.**

Group action



S is group

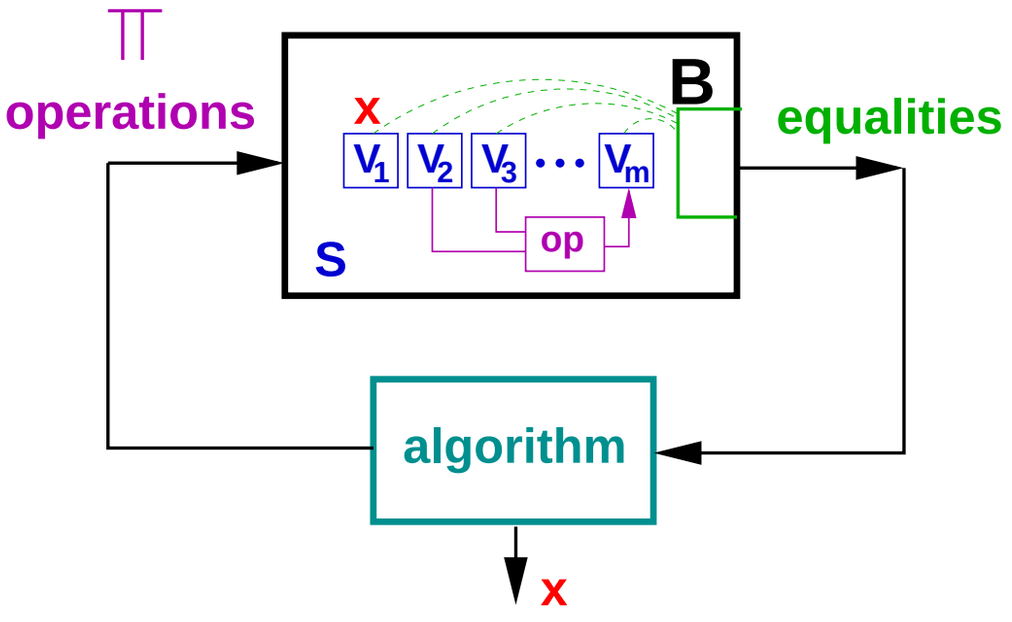
$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof:

1. **Assume: First collision yields x.**
2. **Adaptiveness of no help in provoking first collision.**
3. **Assume one can directly compute functions in $\overline{\Pi}$.**

Group action



S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

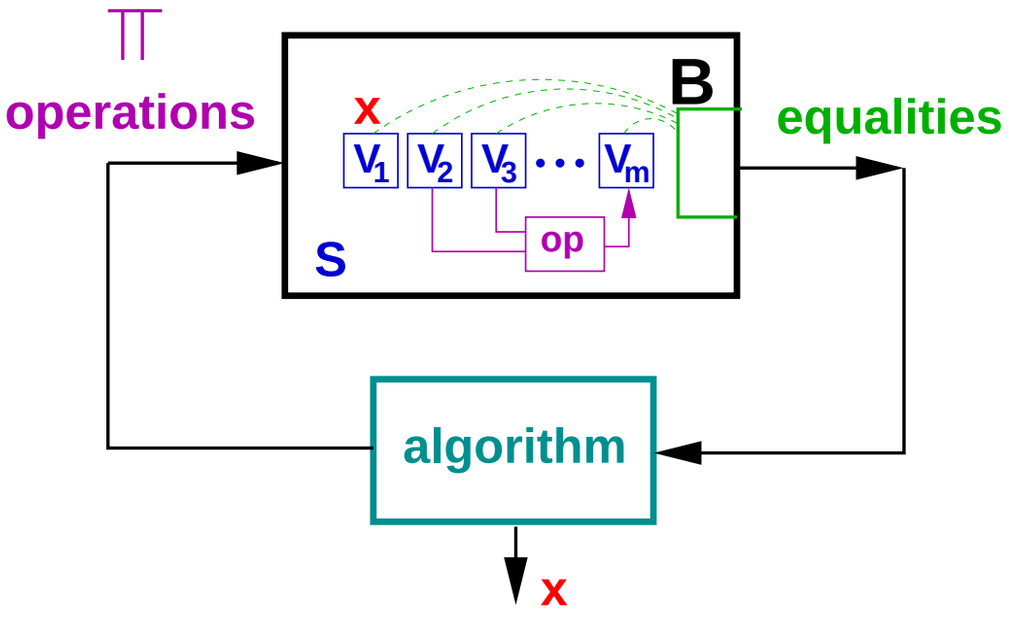
Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof:

1. **Assume: First collision yields x.**
2. **Adaptiveness of no help in provoking first collision.**
3. **Assume one can directly compute functions in $\overline{\Pi}$.**

$$\overline{\Pi} = \underbrace{\{\mathbf{b} |\ \mathbf{b} \in \mathbf{S}\}}_{\mathbf{t}} \cup \underbrace{\{\mathbf{x} \star \mathbf{a} \mid \mathbf{a} \in \mathbf{S}\}}_{\mathbf{u}}$$

Group action



S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

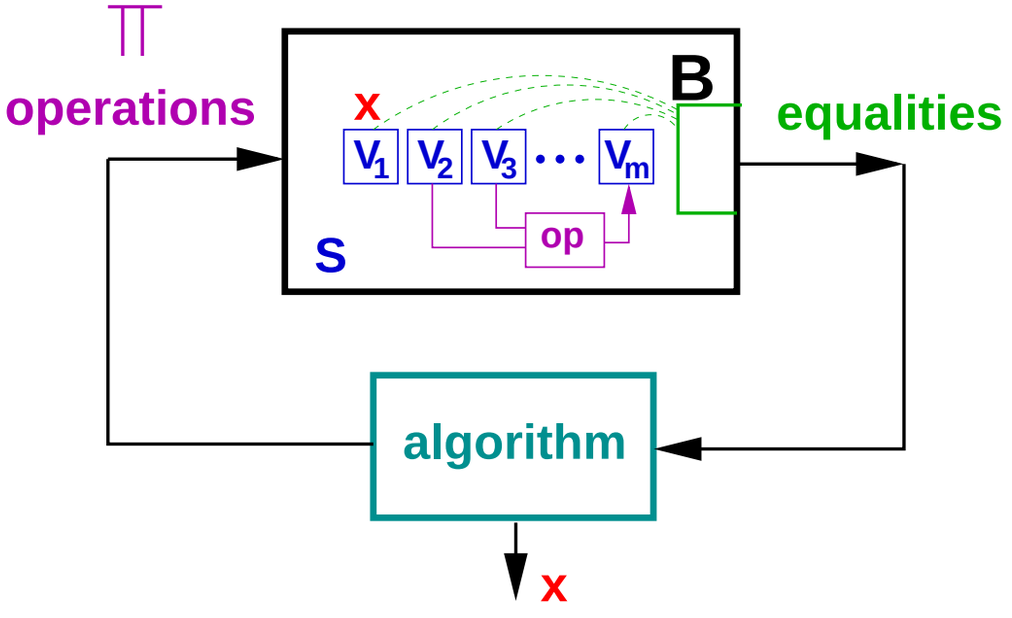
Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof:
1. **Assume: First collision yields x.**
2. **Adaptiveness of no help in provoking first collision.**
3. **Assume one can directly compute functions in $\overline{\Pi}$.**

$$\overline{\Pi} = \underbrace{\{\mathbf{b} |\ \mathbf{b} \in \mathbf{S}\}}_{\mathbf{t}} \cup \underbrace{\{\mathbf{x} \star \mathbf{a} \mid \mathbf{a} \in \mathbf{S}\}}_{\mathbf{u}}$$

Only collisions $\mathbf{x} \star \mathbf{a} = \mathbf{b} \;\Rightarrow\; \leq$ tu values x cause collision

Group action



S is group

$\Pi = \mathcal{C} \cup \{\mathbf{v} \mapsto \mathbf{v} \star \mathbf{a} | \mathbf{a} \in \mathbf{S}\}$

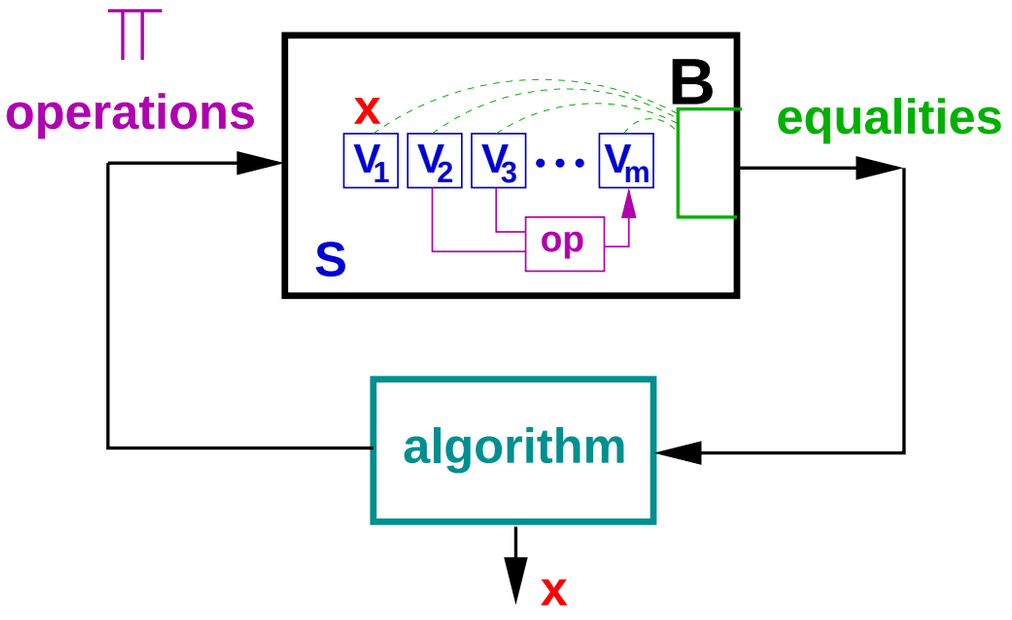
Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow \; O(\sqrt{n})$ **lower bound**

Proof:
1. **Assume: First collision yields x.**
2. **Adaptiveness of no help in provoking first collision.**
3. **Assume one can directly compute functions in $\overline{\Pi}$.**

$\overline{\Pi} = \underbrace{\{\mathbf{b} |\ \mathbf{b} \in \mathbf{S}\}}_{\mathbf{t}} \cup \underbrace{\{\mathbf{x} \star \mathbf{a} \mid \mathbf{a} \in \mathbf{S}\}}_{\mathbf{u}} \qquad \mathbf{t} = \mathbf{u} = k/2$

Only collisions $\mathbf{x} \star \mathbf{a} = \mathbf{b} \;\Rightarrow\; \leq$ tu values x cause collision

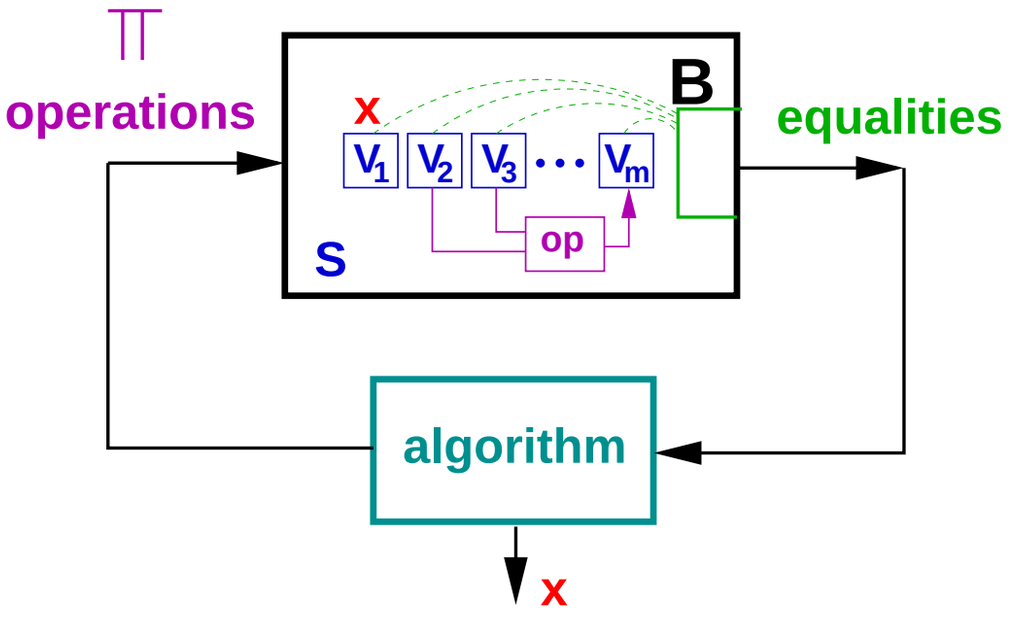
Group operation $\oplus$



$\mathbf{S} = \{0,1\}^\ell,\ n = 2^\ell$

$\Pi = \mathcal{C} \cup \{\oplus\}$

Group operation $\oplus$

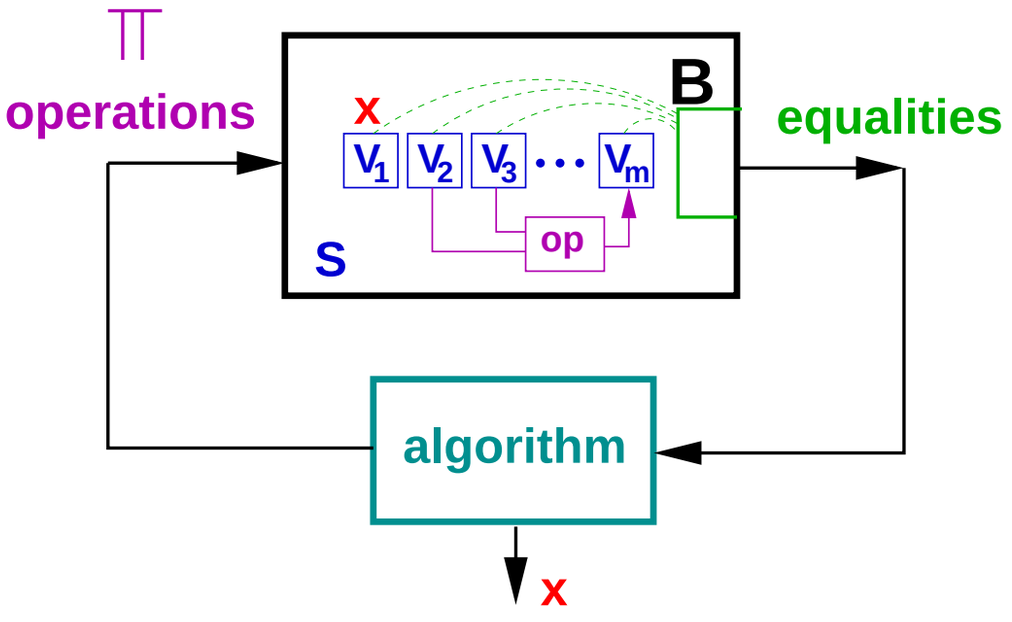


$\mathbf{S} = \{0,1\}^\ell,\ n = 2^\ell$

$\Pi = \mathcal{C} \cup \{\oplus\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow\ O(\sqrt{n})$ **lower bound**

Group operation $\oplus$



$\mathbf{S} = \{0,1\}^\ell,\ n = 2^\ell$

$\Pi = \mathcal{C} \cup \{\oplus\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{4}k^2/n \quad \Rightarrow\ O(\sqrt{n})$ **lower bound**

Proof: $\overline{\Pi} = \{\mathbf{b} |\ \mathbf{b} \in \mathbf{S}\} \cup \{\mathbf{x} \oplus \mathbf{a} \mid \mathbf{a} \in \mathbf{S}\}$

Same argument as before.

Polynomial equations modulo q

Lemma: **The fraction of solutions $(x_1, \ldots, x_k) \in \mathbf{Z}_q$ of the multivariate polynomial equation**

$$p(x_1, \ldots, x_k) \equiv_q 0$$

of degree d is at most d/q.

Polynomial equations modulo q

Lemma: **The fraction of solutions $(x_1, \ldots, x_k) \in \mathbb{Z}_q$ of the multivariate polynomial equation**

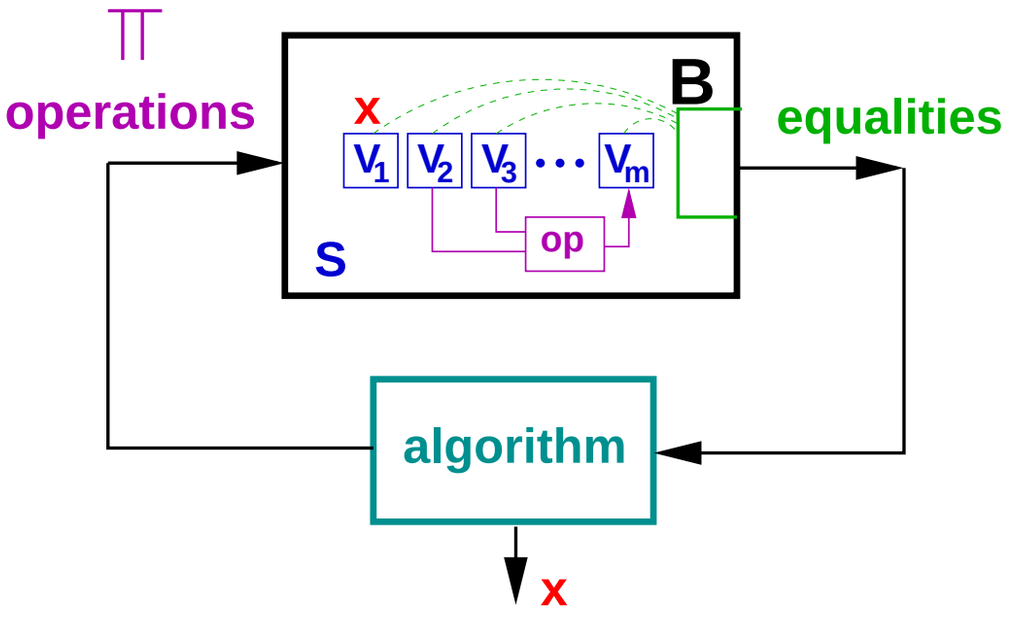
$$p(x_1, \ldots, x_k) \equiv_q 0$$

of degree d is at most d/q.

Proof sketch:

- **q prime $\Rightarrow$ $\mathbb{Z}_q$ is a field.**
- **univariate degree-d polynomial $p(x)$: # of roots $\leq d$ (unless $p(x)$ is the 0-polynomial)**
- **Generalize to multivariate polynomials**

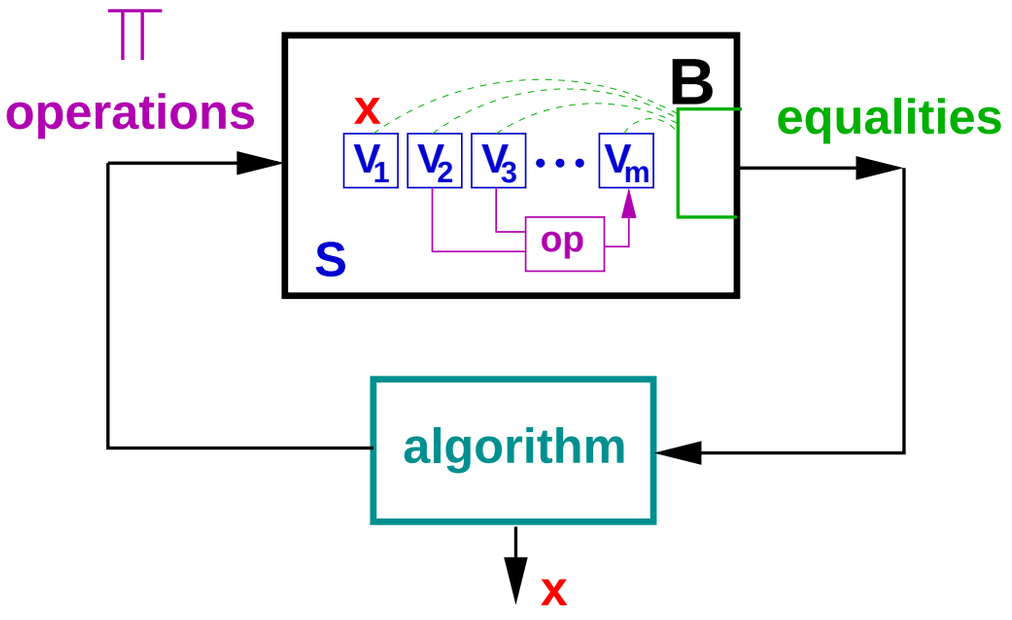
Group operation $+$



$\mathbf{S} = \mathbb{Z}_n$

$\Pi = \{\mathbf{1}, +\}$

Group operation $+$

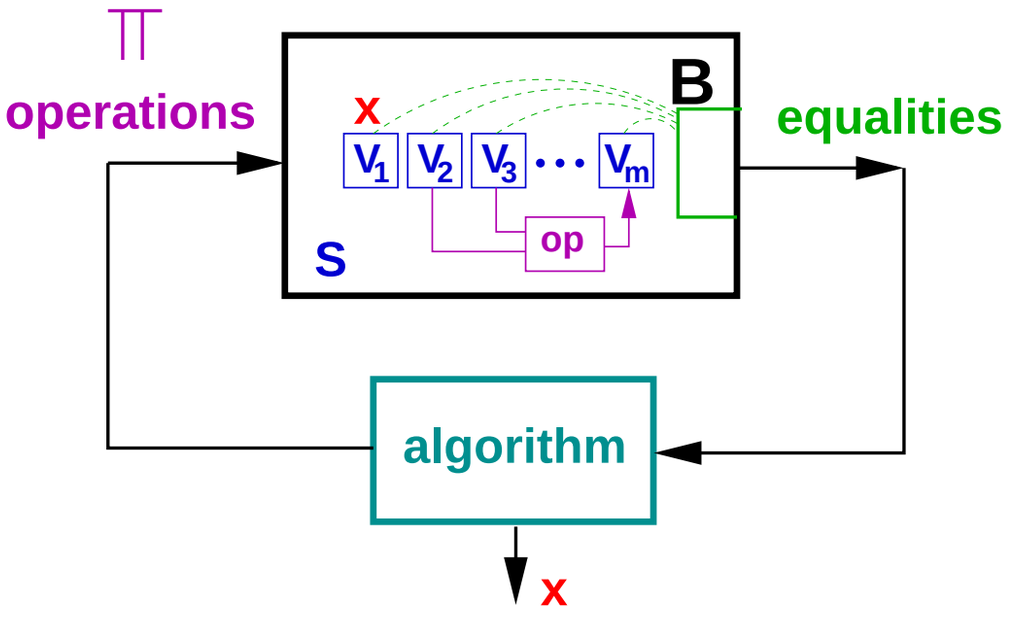


$\mathbf{S} = \mathbf{Z}_n, \quad q|n$

$\Pi = \{\mathbf{1}, +\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

Group operation $+$: Generic l. b. for DL

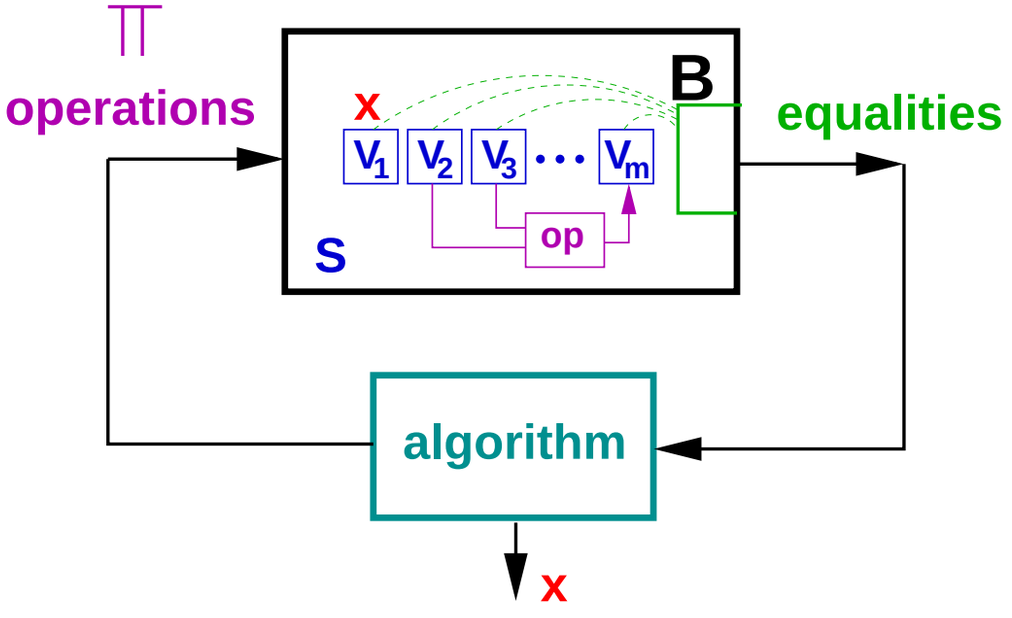


$S = Z_n, \quad q|n$

$\Pi = \{1, +\}$

Theorem: $p_S \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

Group operation $+$: Generic l. b. for DL



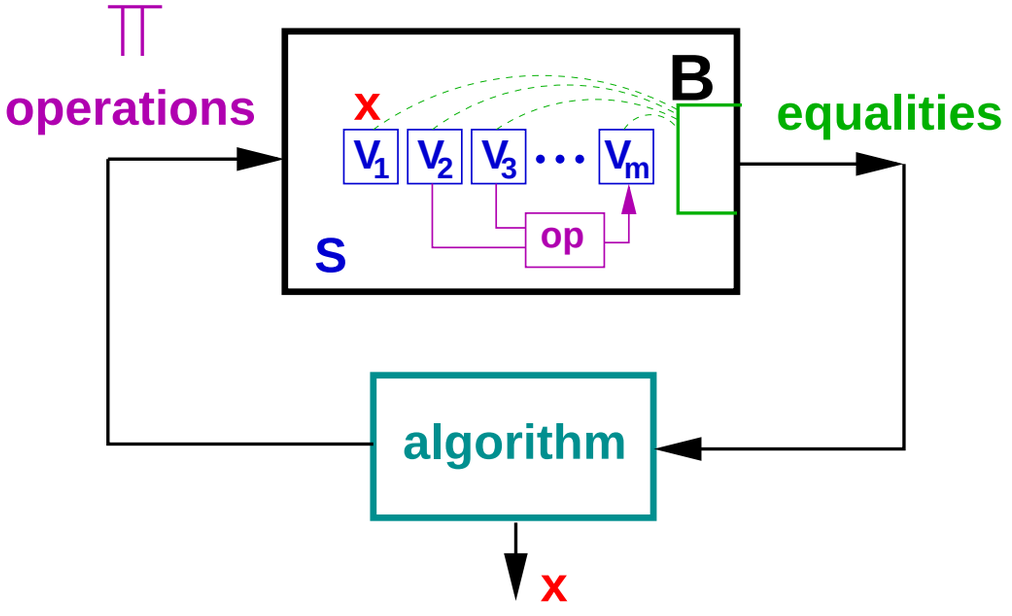
$\mathbf{S} = \mathbb{Z}_n, \quad q|n$

$\Pi = \{\mathbf{1}, +\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

Proof: **x** correct mod $n \Rightarrow$ **x** correct mod every divisor q of n

Group operation $+$: Generic l. b. for DL



$\mathbf{S} = \mathbf{Z}_n, \quad q|n$

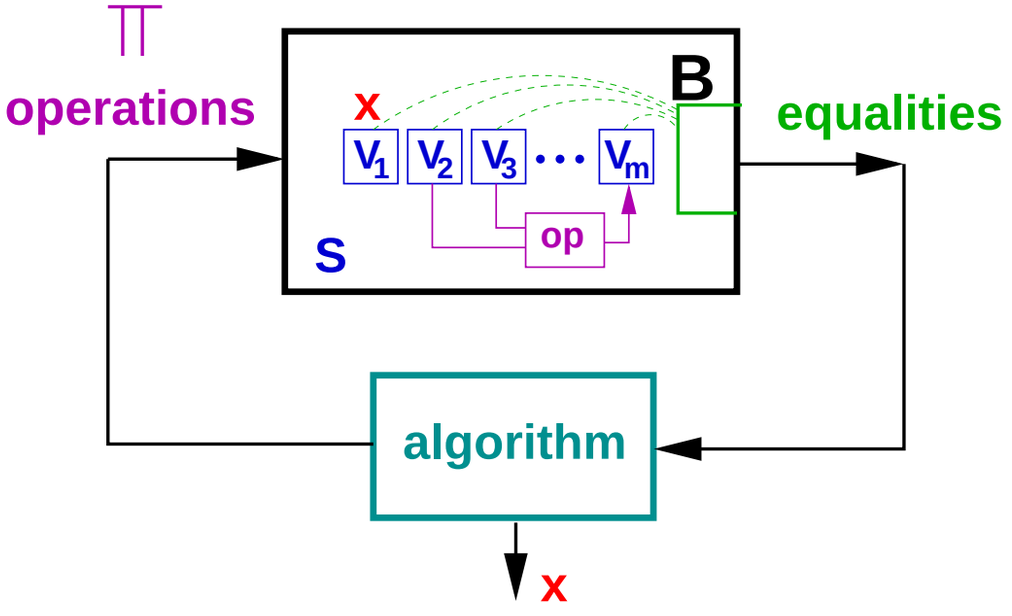
$\Pi = \{\mathbf{1}, +\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

Proof: **x correct mod** $n \Rightarrow$ **x correct mod every divisor** q **of** n

$\overline{\Pi} = \{\mathbf{a}x + \mathbf{b} \mid \mathbf{a}, \mathbf{b} \in \mathbf{S}\}$

Group operation $+$: Generic l. b. for DL



$\mathbf{S} = \mathbf{Z}_n, \quad q|n$

$\Pi = \{\mathbf{1}, +\}$

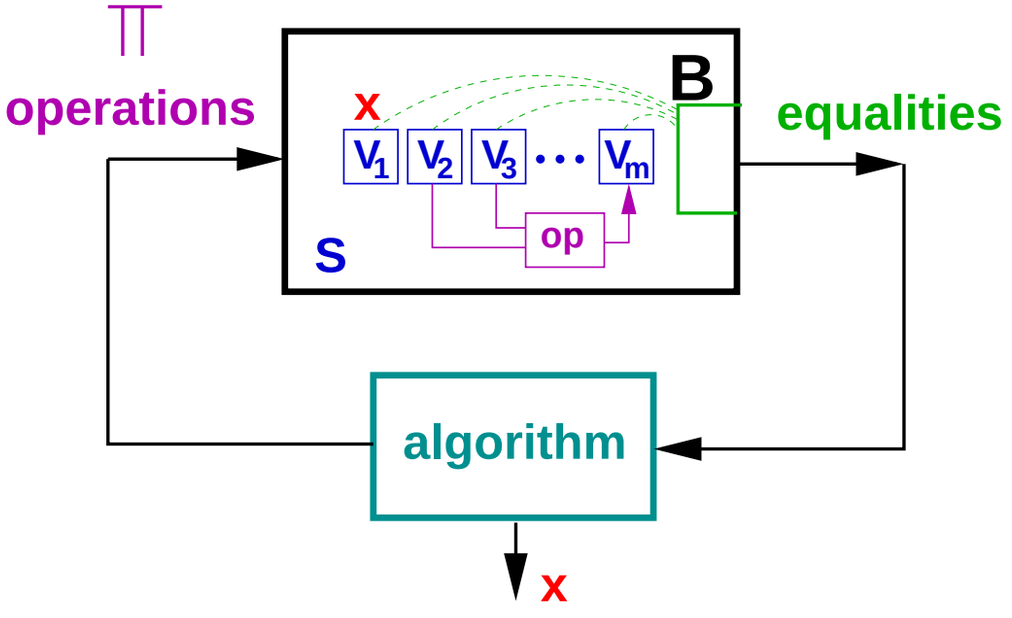
Theorem: $\mathbf{p_S} \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

Proof: **x** correct mod $n \Rightarrow$ **x** correct mod every divisor q of n

$\overline{\Pi} = \{\mathbf{a}x + \mathbf{b} \mid \mathbf{a}, \mathbf{b} \in \mathbf{S}\}$

Collision $\mathbf{a}_i x + \mathbf{b}_i = \mathbf{a}_j x + \mathbf{b}_j$ has unique solution **x** mod q.

Group operation $+$: Generic l. b. for DL



$\mathbf{S} = \mathbb{Z}_n, \quad q|n$

$\Pi = \{\mathbf{1}, +\}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}k^2/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound (exact)**

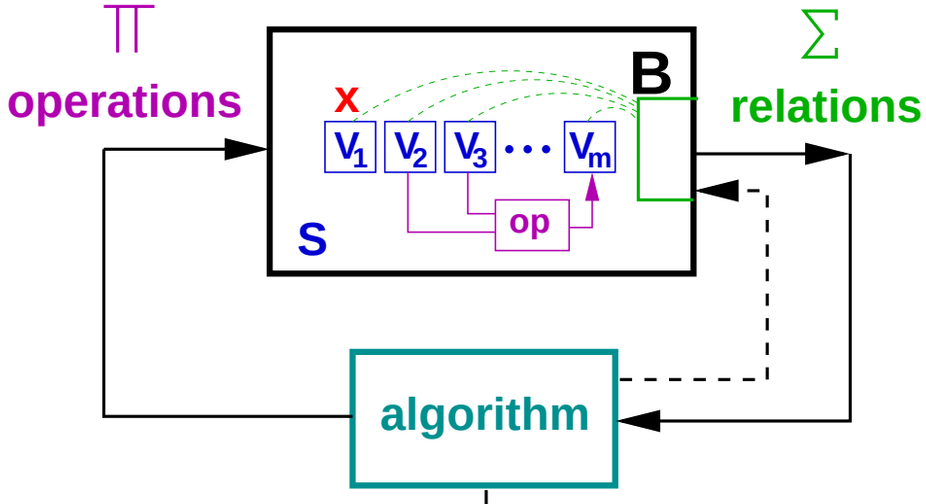
Proof: **x** correct mod $n \Rightarrow$ **x** correct mod every divisor q of n

$\overline{\Pi} = \{\mathbf{a}x + \mathbf{b} \mid \mathbf{a}, \mathbf{b} \in \mathbf{S}\}$

Collision $\mathbf{a}_i x + \mathbf{b}_i = \mathbf{a}_j x + \mathbf{b}_j$ has unique solution **x** mod q.

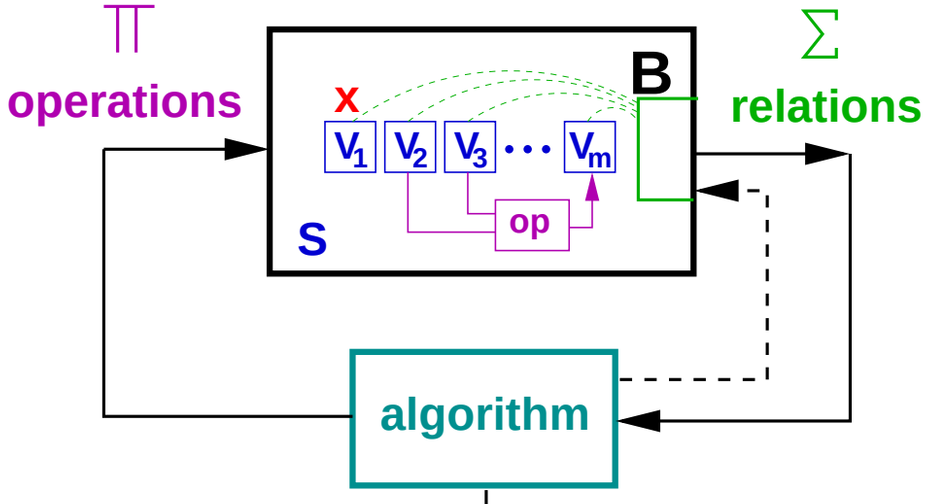
# of solutions **x** bounded by # of pairs (i, j): $\binom{k}{2}$

Three types of problems



Initial state: $V_1, \ldots, V_d$

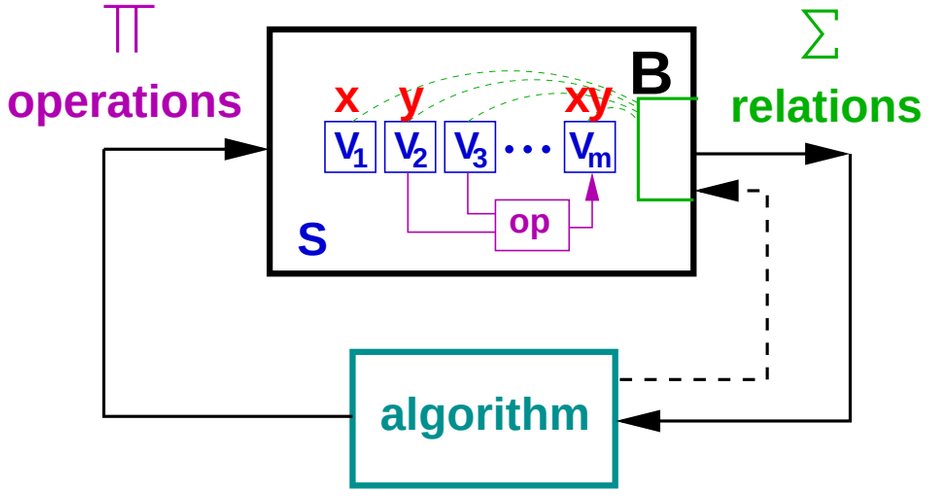
Three types of problems



Initial state: $V_1, \ldots, V_d$

- **Extraction:** Extract the initial value **x** of V_1. Example: DL

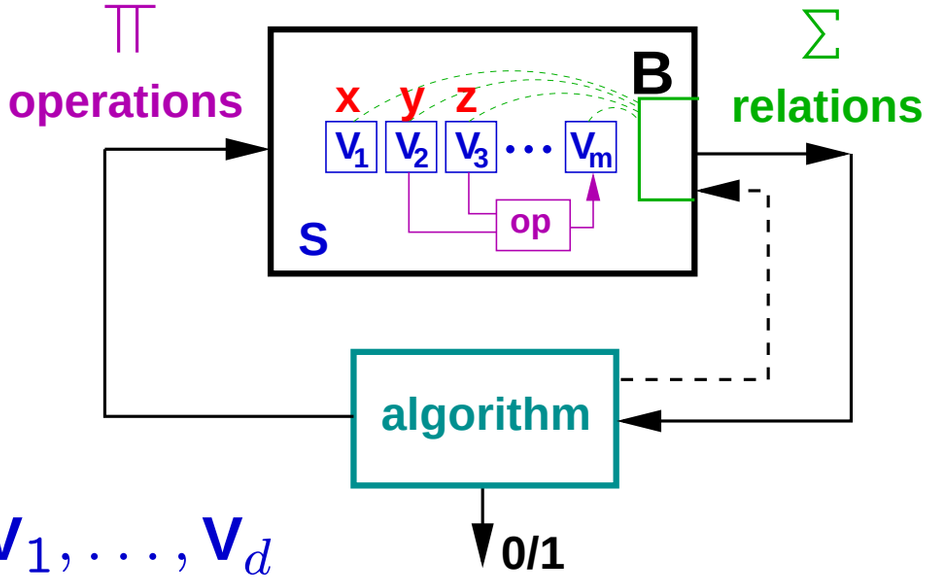
Three types of problems



Initial state: $V_1, \ldots, V_d$

- **Extraction:** Extract the initial value **x** of V_1. Example: DL
- **Computation:** Compute a function $f : S^d \rightarrow S$ of the initial state within $\mathbb{B}$. Example CDH: $x, y \rightarrow xy$

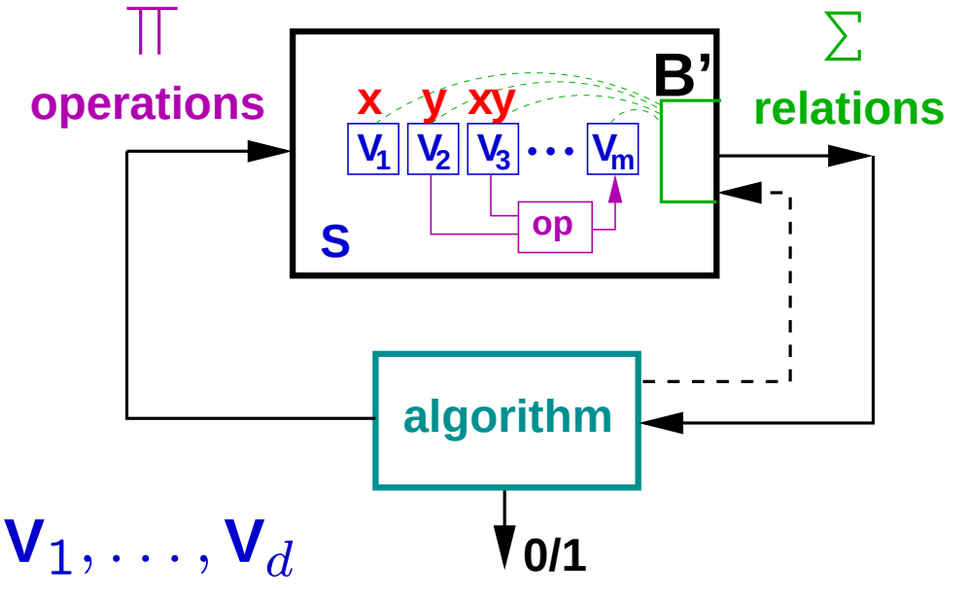
Three types of problems



Initial state: $\mathbf{V}_1, \ldots, \mathbf{V}_d$

- **Extraction:** Extract the initial value **x** of $\mathbf{V}_1$. Example: DL
- **Computation:** Compute a function $f : \mathbf{S}^d \rightarrow \mathbf{S}$ of the initial state within B. Example CDH: $\mathbf{x}, \mathbf{y} \rightarrow \mathbf{xy}$
- **Distinction:** Distinguish two black-boxes B and B' of the same type with different distributions of the initial state. Example DDH: $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ vs. $(\mathbf{x}, \mathbf{y}, \mathbf{xy})$

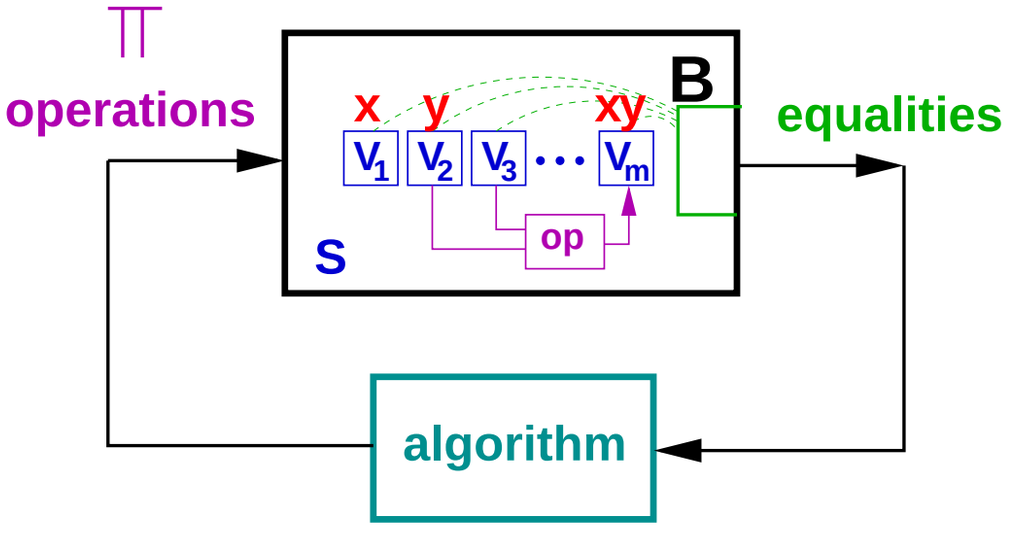
Three types of problems



Initial state: $V_1, \ldots, V_d$

- **Extraction:** Extract the initial value **x** of V_1. Example: DL
- **Computation:** Compute a function $f : S^d \to S$ of the initial state within B. Example CDH: $x, y \to xy$
- **Distinction:** Distinguish two black-boxes B and B' of the same type with different distributions of the initial state. Example DDH: (x, y, z) vs. (x, y, xy)

Generic lower bound for CDH

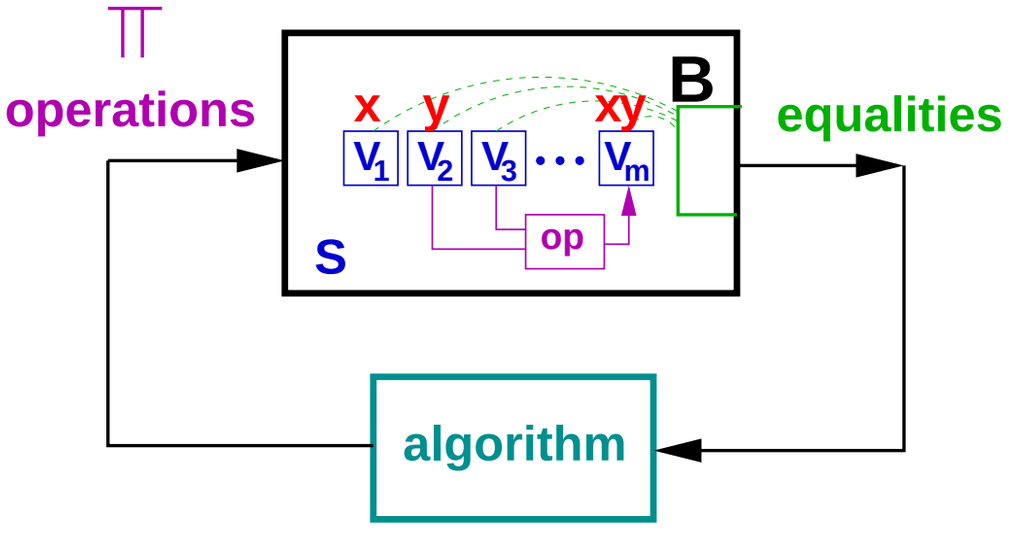


S $= Z_n$**, q larg. p.f. of n**

$\Pi = \{1, +\}$

Goal: **x**, **y** $\rightarrow$ **xy**

Generic lower bound for CDH



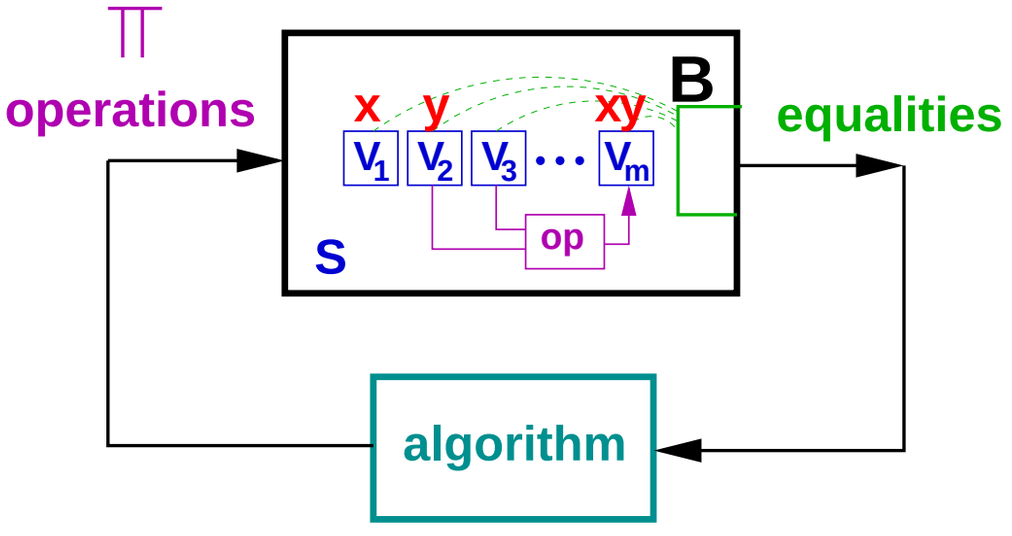
S $= \mathbb{Z}_n$, q **larg. p.f. of** n

$\sqcap = \{1, +\}$

Goal: **x**, **y** $\rightarrow$ **xy**

Theorem: $\mathbf{p_S} \leq \frac{1}{2}(k^2 + 3k)/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound**

Generic lower bound for CDH



$S = Z_n$, q larg. p.f. of n

$\Pi = \{1, +\}$

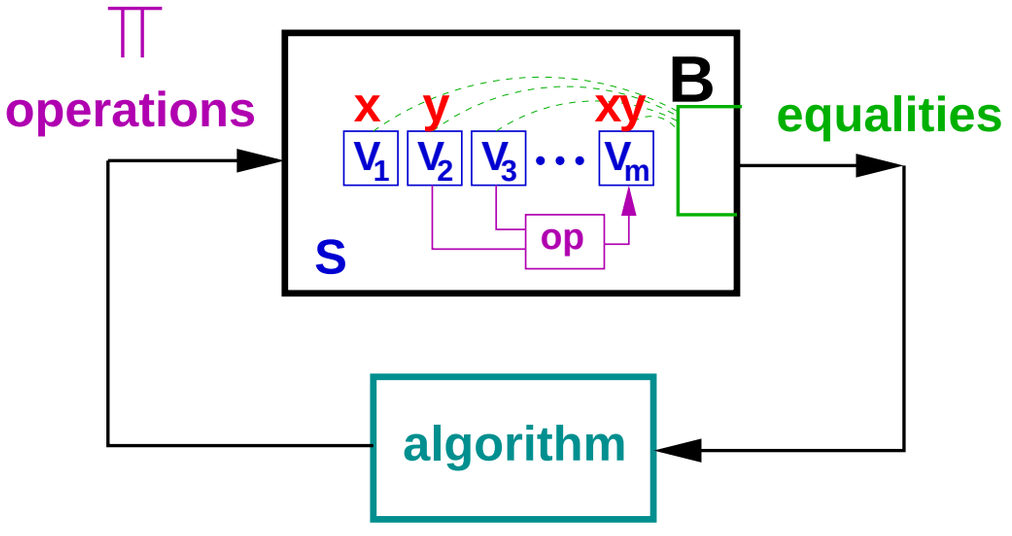
Goal: $x, y \rightarrow xy$

Theorem: $p_S \leq \frac{1}{2}(k^2 + 3k)/q \quad \Rightarrow \; O(\sqrt{q})$ lower bound

Proof:

$\overline{\Pi} = \{ax + by + c \mid a, b, c \in S\}$

Generic lower bound for CDH



$\mathbf{S} = \mathbb{Z}_n$, q **larg. p.f. of** n

$\Pi = \{\mathbf{1}, +\}$

Goal: $\mathbf{x}, \mathbf{y} \rightarrow \mathbf{xy}$

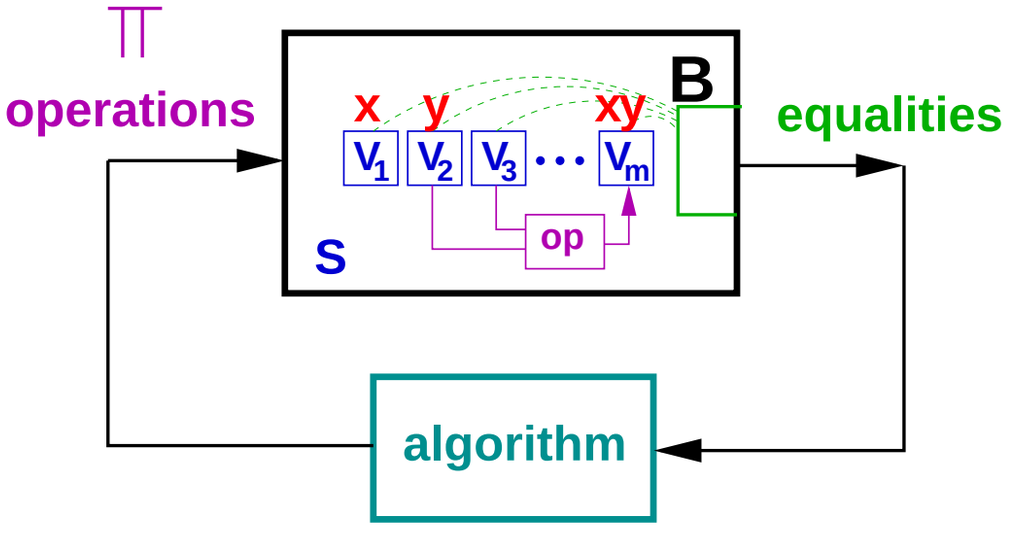
Theorem: $\mathbf{p_S} \leq \frac{1}{2}(k^2 + 3k)/q \quad \Rightarrow O(\sqrt{q})$ **lower bound**

Proof:

$\overline{\Pi} = \{\mathbf{a}\mathbf{x} + \mathbf{b}\mathbf{y} + \mathbf{c} \mid \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbf{S}\}$

Collisions: $\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i = \mathbf{a}_j\mathbf{x} + \mathbf{b}_j\mathbf{y} + \mathbf{c}_j$

Generic lower bound for CDH



$\mathbf{S} = \mathbb{Z}_n$, q **larg. p.f. of** n

$\Pi = \{\mathbf{1}, +\}$

Goal: $\mathbf{x}, \mathbf{y} \rightarrow \mathbf{xy}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}(k^2 + 3k)/q \quad \Rightarrow \; O(\sqrt{q})$ **lower bound**

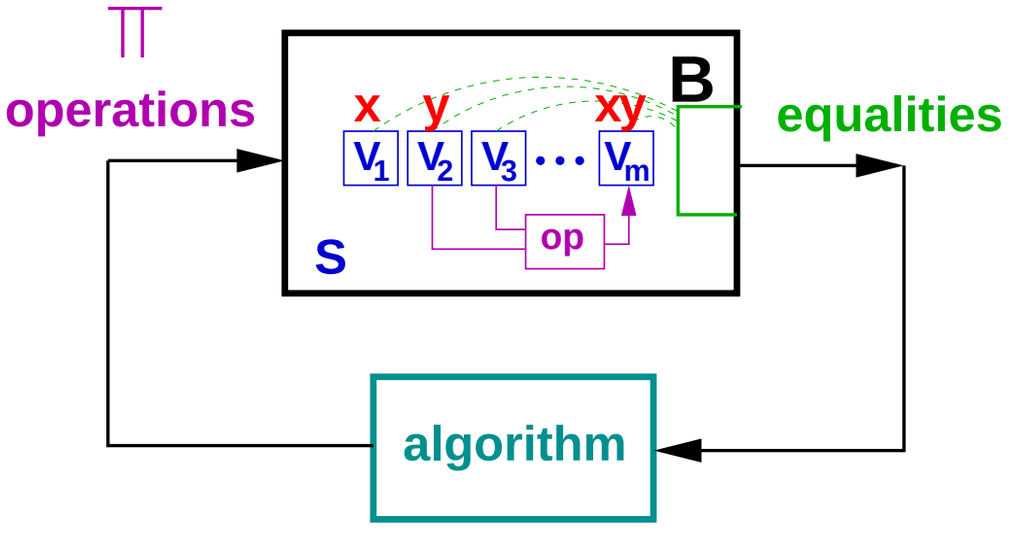
Proof:

$\overline{\Pi} = \{\mathbf{a}\mathbf{x} + \mathbf{b}\mathbf{y} + \mathbf{c} \mid \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbf{S}\}$

Collisions: $\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i = \mathbf{a}_j\mathbf{x} + \mathbf{b}_j\mathbf{y} + \mathbf{c}_j$

$\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i = \mathbf{xy}$

Generic lower bound for CDH



$\mathbf{S} = \mathbb{Z}_n$, q **larg. p.f. of** n

$\Pi = \{\mathbf{1}, +\}$

Goal: $\mathbf{x}, \mathbf{y} \rightarrow \mathbf{xy}$

Theorem: $\mathbf{p_S} \leq \frac{1}{2}(k^2 + 3k)/q \quad \Rightarrow\ O(\sqrt{q})$ **lower bound**

Proof:

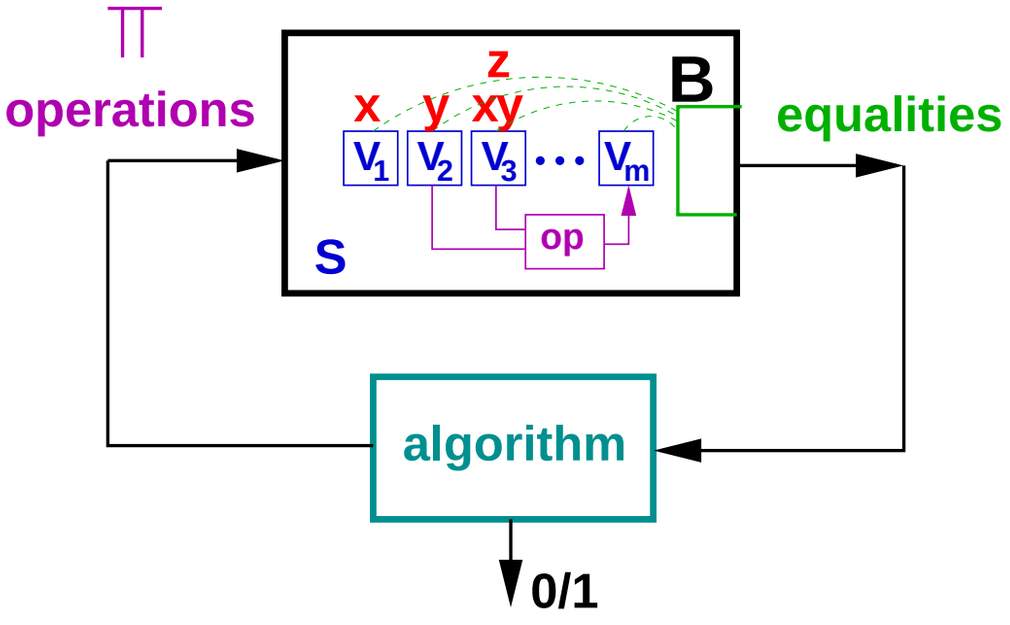
$\overline{\Pi} = \{\mathbf{a}\mathbf{x} + \mathbf{b}\mathbf{y} + \mathbf{c} \mid \mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbf{S}\}$

Collisions: $\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i = \mathbf{a}_j\mathbf{x} + \mathbf{b}_j\mathbf{y} + \mathbf{c}_j$

$\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i = \mathbf{xy}$

Fraction of pairs $(\mathbf{x}, \mathbf{y})$ **giving a collision** $\leq \left(\binom{k}{2} + 2k\right)/q$

Generic lower bound for DDH

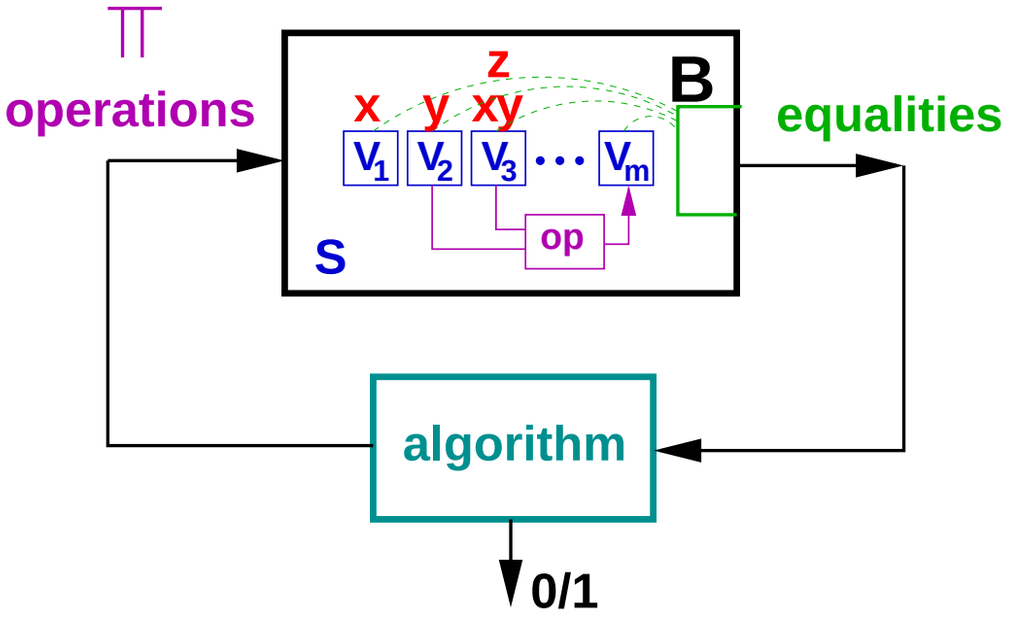


$S = Z_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

Goal: (x, y, z) **vs.** (x, y, xy)

Generic lower bound for DDH



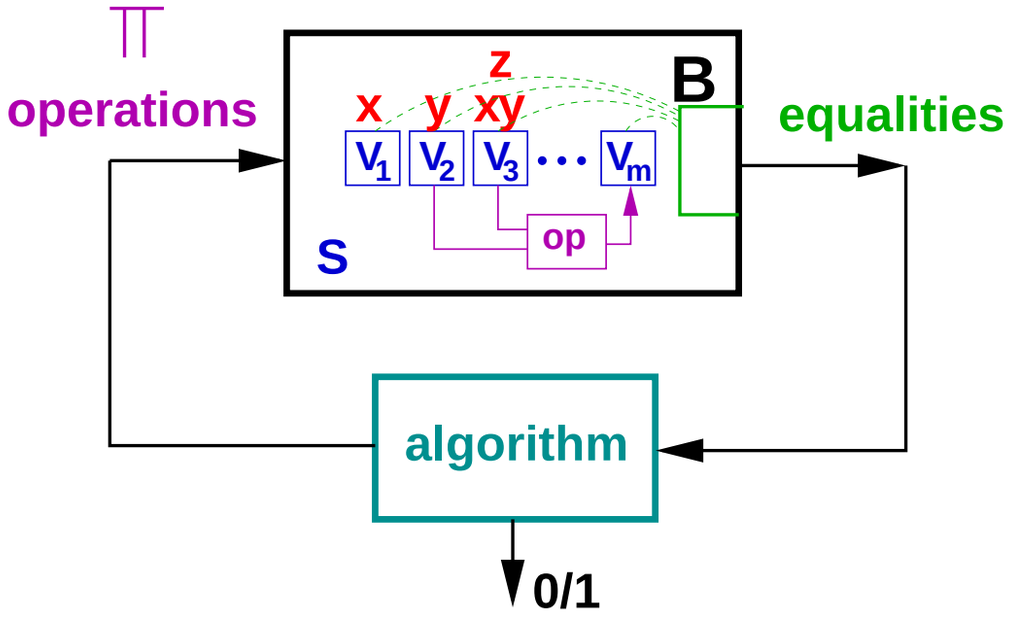
$S = Z_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

Goal: (x, y, z) **vs.** (x, y, xy)

Theorem: $p_S \leq k^2/p_{min} \quad \Rightarrow \quad O(\sqrt{p_{min}})$ **lower bound**

Generic lower bound for DDH



$S = Z_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

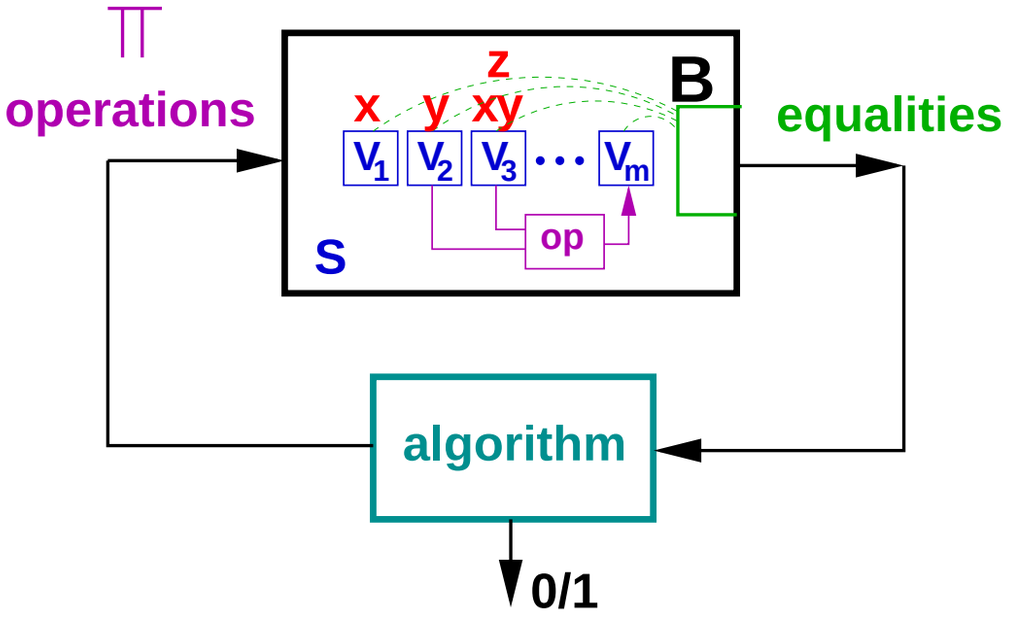
Goal: (x, y, z) **vs.** (x, y, xy)

Theorem: $p_S \leq k^2/p_{min} \quad \Rightarrow \; O(\sqrt{p_{min}})$ **lower bound**

Proof:

$\overline{\Pi} = \{ax + by + cxy + d \mid a, b, c, d \in S\}$

Generic lower bound for DDH



$S = Z_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

Goal: (x, y, z) **vs.** (x, y, xy)

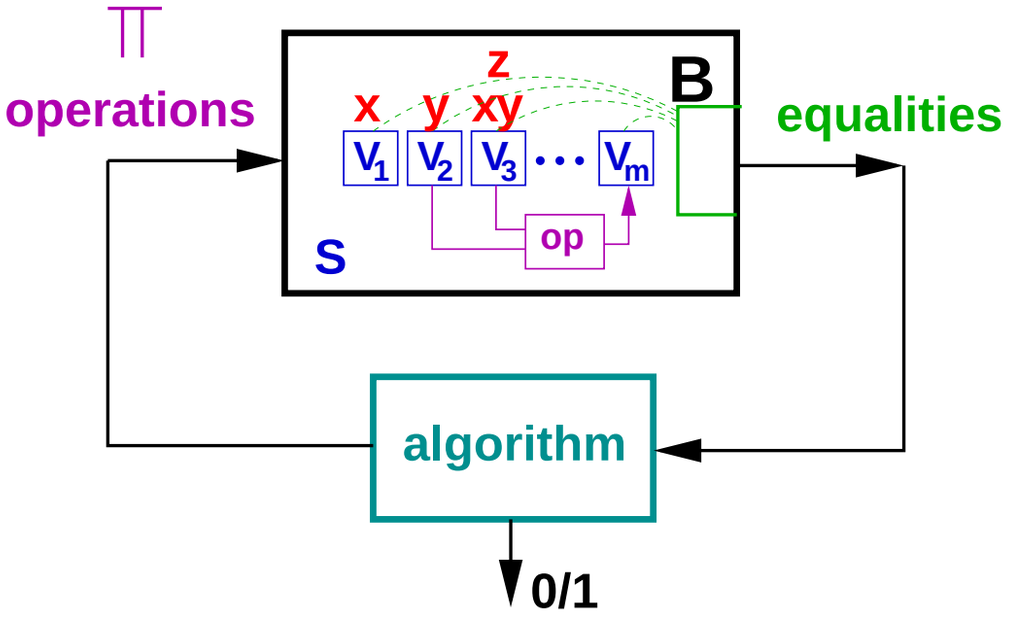
Theorem: $p_S \leq k^2/p_{min} \Rightarrow O(\sqrt{p_{min}})$ **lower bound**

Proof:

$\overline{\Pi} = \{ax + by + cxy + d \mid a, b, c, d \in S\}$

Collisions: $a_i x + b_i y + c_i xy + d_i = a_j x + b_j y + c_j xy + d_j$

Generic lower bound for DDH



$\mathbf{S} = \mathbb{Z}_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

Goal: $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ **vs.** $(\mathbf{x}, \mathbf{y}, \mathbf{xy})$

Theorem: $\mathbf{p_S} \leq k^2/p_{min} \quad \Rightarrow \; O(\sqrt{p_{min}})$ **lower bound**

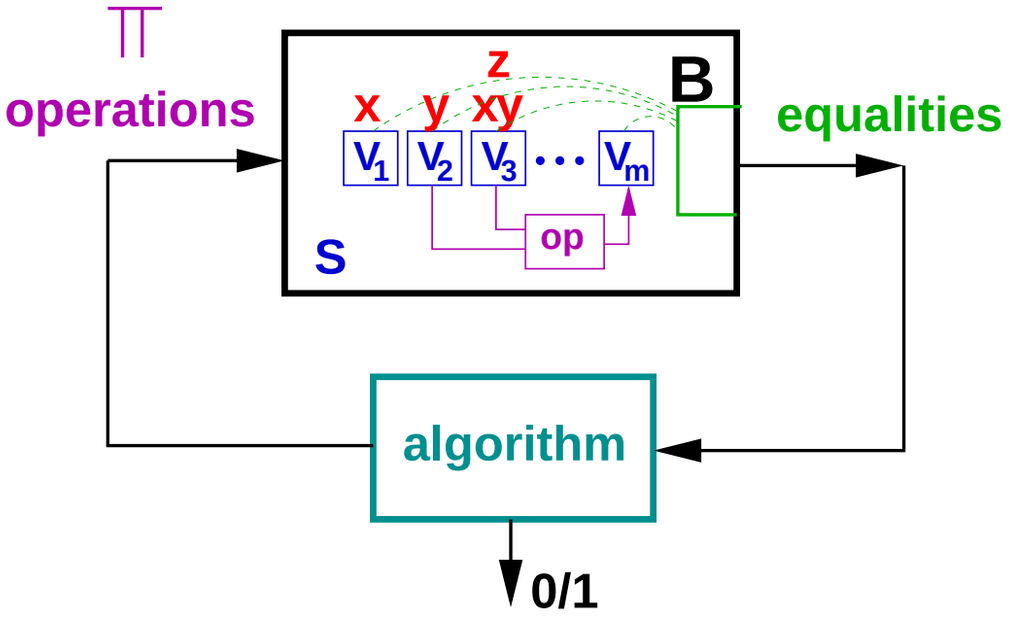
Proof:

$\overline{\Pi} = \{\mathbf{a}x + \mathbf{b}y + \mathbf{c}xy + \mathbf{d} \mid \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d} \in \mathbf{S}\}$

Collisions: $\mathbf{a}_i x + \mathbf{b}_i y + \mathbf{c}_i xy + \mathbf{d}_i = \mathbf{a}_j x + \mathbf{b}_j y + \mathbf{c}_j xy + \mathbf{d}_j$

$(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i, \mathbf{d}_i) \neq (\mathbf{a}_j, \mathbf{b}_j, \mathbf{c}_j, \mathbf{d}_j) \quad \Rightarrow \quad \neq$ **mod some** q **(where** $q | n$**)**

Generic lower bound for DDH



$\mathbf{S} = \mathbb{Z}_n, \quad p_{min} | n$

$\Pi = \{1, +\}$

Goal: $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ **vs.** $(\mathbf{x}, \mathbf{y}, \mathbf{xy})$

Theorem: $\mathbf{p_S} \leq k^2/p_{min} \quad \Rightarrow \; O(\sqrt{p_{min}})$ **lower bound**

Proof:

$\overline{\Pi} = \{\mathbf{a}\mathbf{x} + \mathbf{b}\mathbf{y} + \mathbf{c}\mathbf{x}\mathbf{y} + \mathbf{d} \mid \mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d} \in \mathbf{S}\}$

Collisions: $\mathbf{a}_i\mathbf{x} + \mathbf{b}_i\mathbf{y} + \mathbf{c}_i\mathbf{x}\mathbf{y} + \mathbf{d}_i = \mathbf{a}_j\mathbf{x} + \mathbf{b}_j\mathbf{y} + \mathbf{c}_j\mathbf{x}\mathbf{y} + \mathbf{d}_j$

$(\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i, \mathbf{d}_i) \neq (\mathbf{a}_j, \mathbf{b}_j, \mathbf{c}_j, \mathbf{d}_j) \quad \Rightarrow \quad \neq$ **mod some** q **(where** $q | n$**)**

Fraction of pairs $(\mathbf{x}, \mathbf{y})$ **giving a collision** $\leq 2\binom{k}{2}/p_{min}$

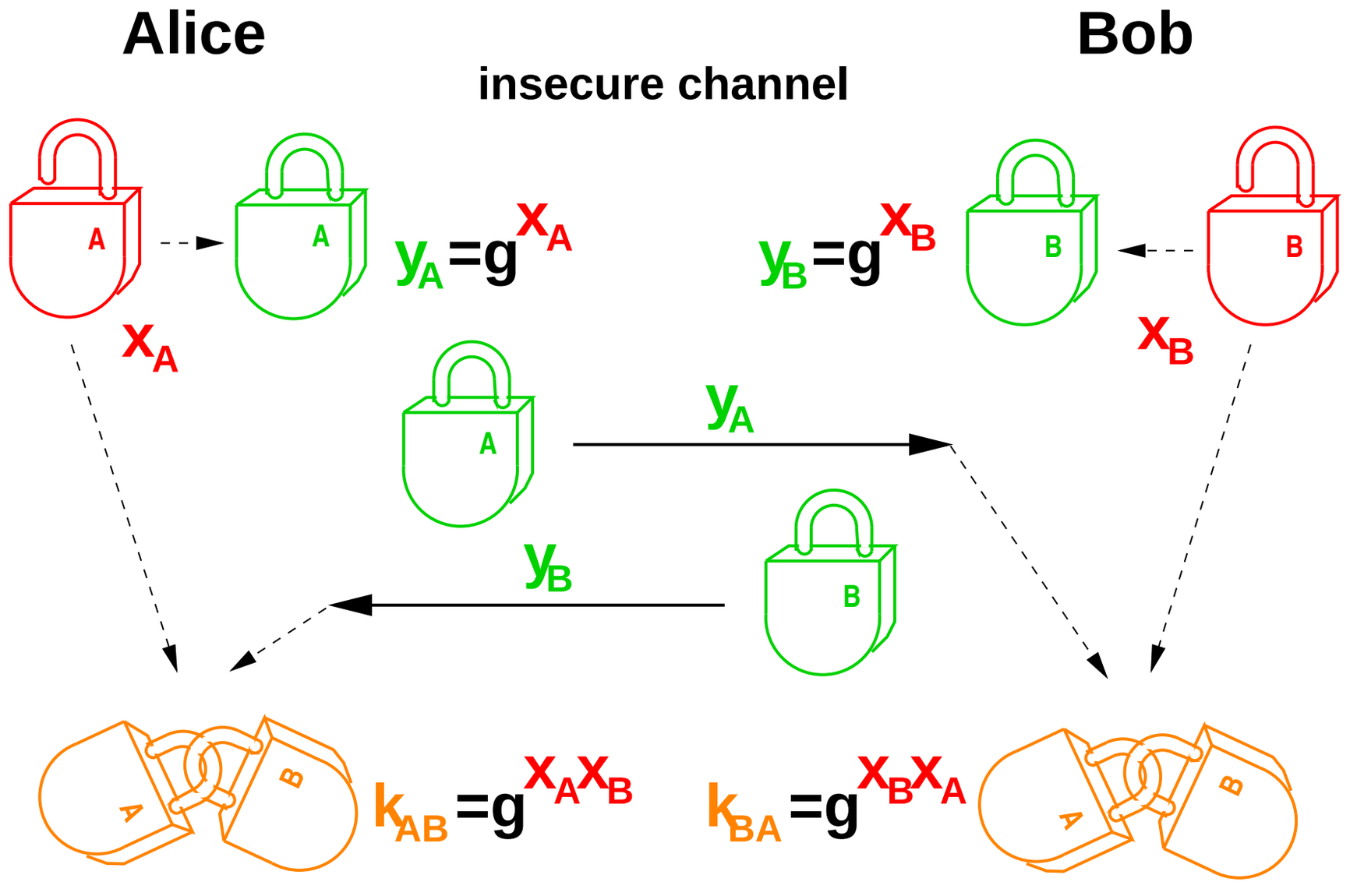
The Equivalence of Breaking Diffie-Hellman and Computing Discrete Logarithms

Ueli Maurer

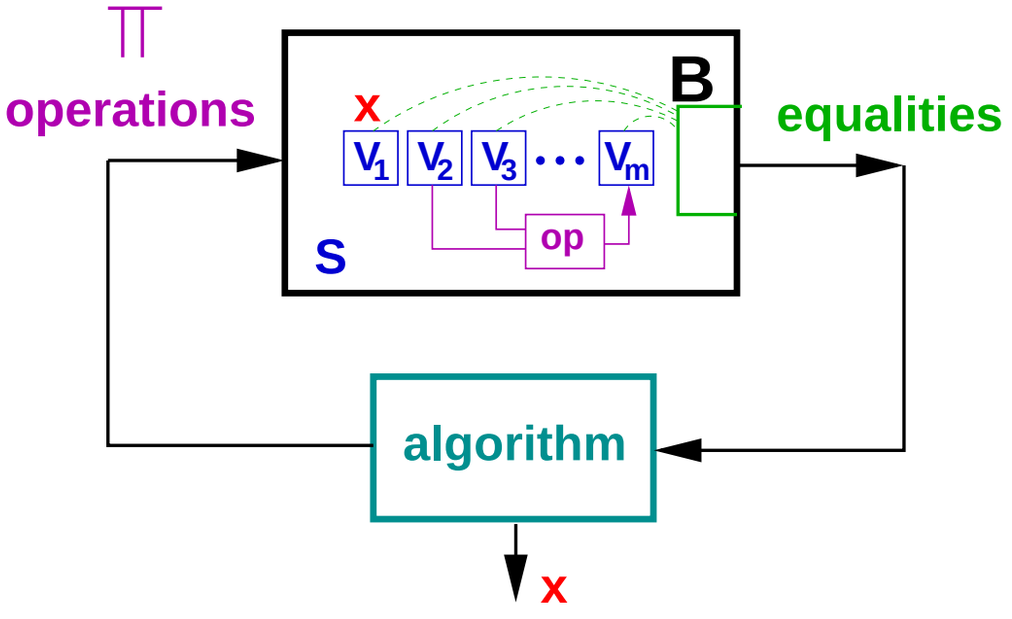
ETH Zurich, www.crypto.ethz.ch

Number Theory and Computational Cryptography
Sept. 29 - Oct. 2, 2010, Warsaw.

Diffie-Hellman protocol: mechanical analog



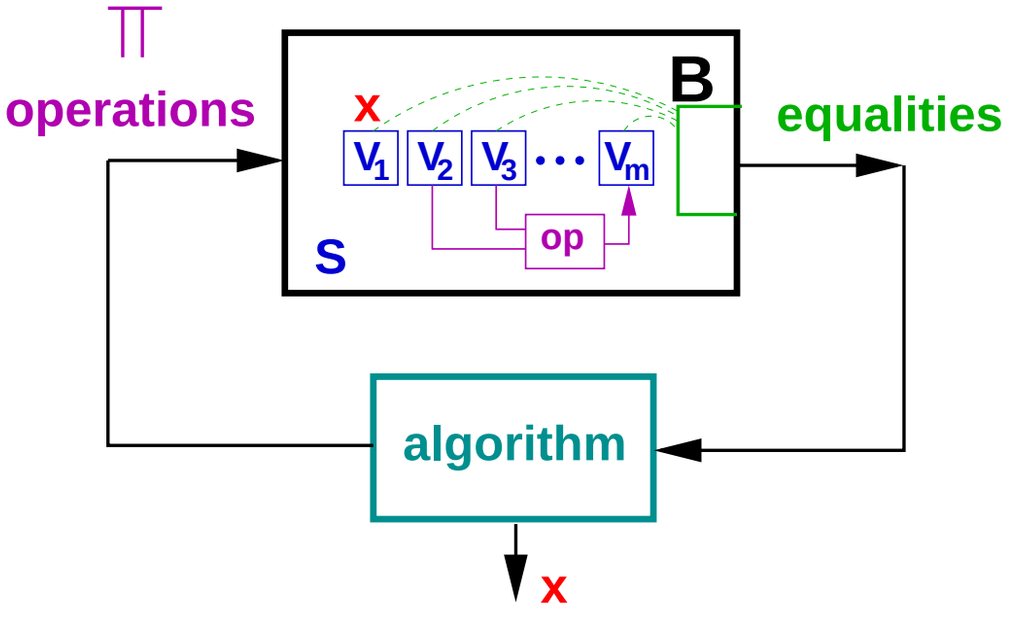
Computing DL, using CDH oracle



$\mathbf{S} = \mathbb{Z}_n$

$\Pi = \{\mathbf{1}, +, \cdot\}$

Computing DL, using CDH oracle

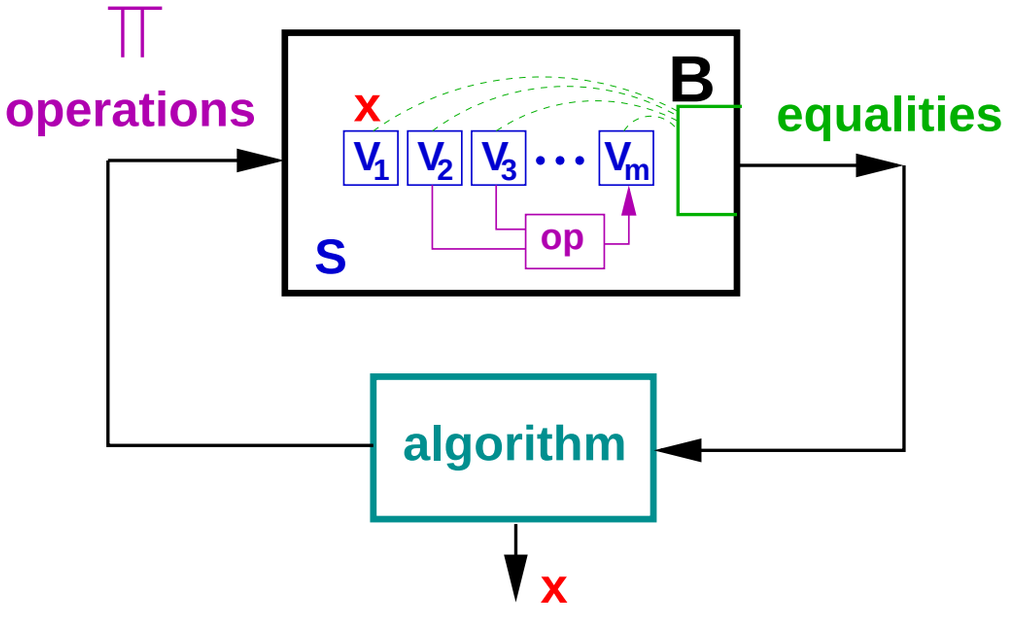


$S = \mathbb{Z}_n$

$\Pi = \{1, +, \cdot\}$

Theorem: For (almost) every n, if $\Pi = \{1, +, \cdot\}$ there exists a polynomial-time extraction algorithm.

Computing DL, using CDH oracle



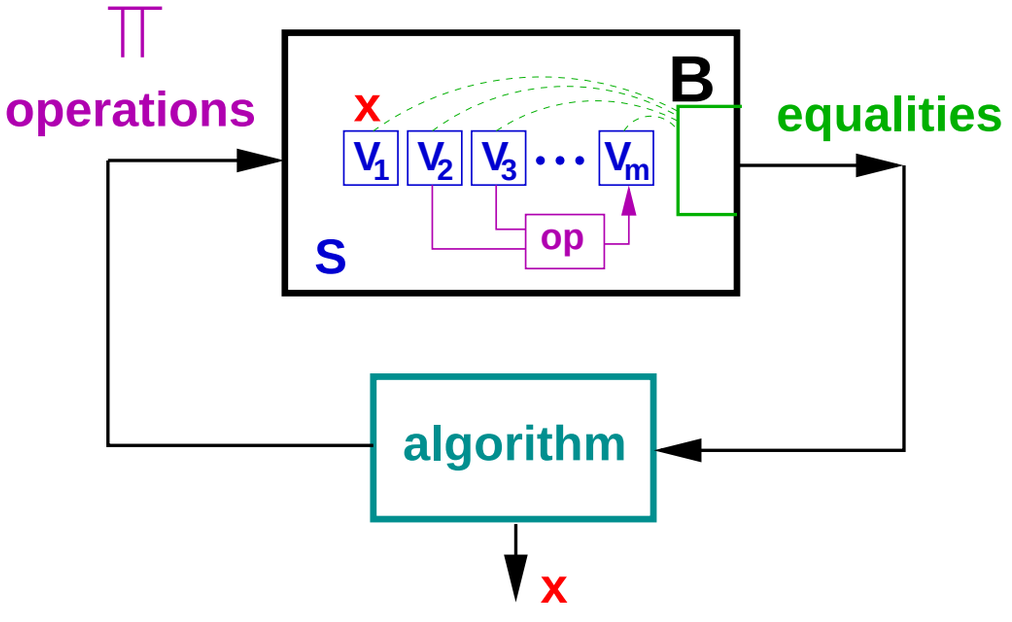
$\mathbf{S} = \mathbb{Z}_n$

$\Pi = \{\mathbf{1}, +, \cdot\}$

Theorem: For **(almost)** every n, if $\Pi = \{\mathbf{1}, +, \cdot\}$ there exists a polynomial-time extraction algorithm.

- non-uniform with respect to n,
- under a plausible **number-theoretic conjecture**.

Computing DL, using CDH oracle



$\mathbf{S} = \mathbb{Z}_n$

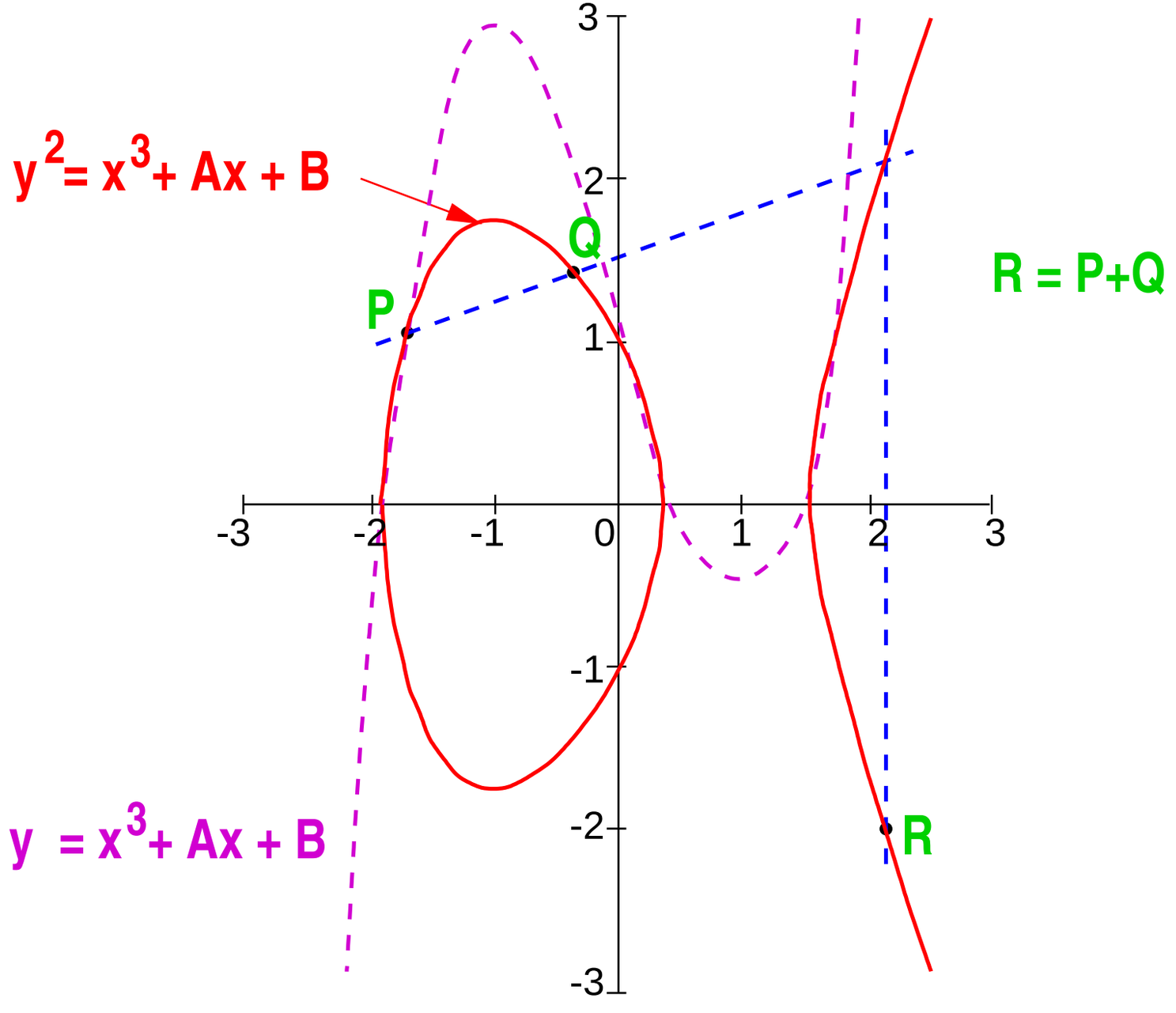
$\Pi = \{\mathbf{1}, +, \cdot\}$

Theorem: For **(almost)** every n, if $\Pi = \{\mathbf{1}, +, \cdot\}$ there exists a polynomial-time extraction algorithm.

- non-uniform with respect to n,
- under a plausible **number-theoretic conjecture**.

CDH–DL equivalence: General DH-groups for general order n can be designed together with the reduction algorithm.

Elliptic curve over the field R



Elliptic curves

An elliptic curve with parameters A and B over a given field $\mathbf{F}$ (with $\mathrm{char}(\mathbf{F}) \neq 2, 3$), where $4A^3 + 27B^2 \neq 0$, is the set of points:

$$E_{A,B} = \{(x, y) : \ y^2 = x^3 + Ax + B\} \cup \{\infty\}.$$

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ with $x_1 \neq x_2$. Then $R = P + Q = (x_3, y_3)$, where:

$$\lambda = \begin{cases} (y_2 - y_1)/(x_2 - x_1) & \text{if } x_1 \neq x_2 \\ (3x_1^2 + A)/(2y_1) & \text{if } x_1 = x_2,\ y_1 = y_2. \end{cases}$$

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \\ y_3 &= \lambda(x_1 - x_3) - y_1 \end{aligned}$$

Facts about elliptic curves over finite fields

- (Hasse, 1934). The order of the elliptic curve $E_{A,B}$ over $GF(q)$ lies in the interval

$$[q + 1 - 2\sqrt{q},\ q + 1 + 2\sqrt{q}].$$

- (Waterhouse, 1969). If q is prime, every order in this interval is assumed for some parameters A and B.

- (Rück, 1987). This is true even when we require the curve to be cyclic.

- (Lenstra, 1987). In the interval $[q + 1 - \sqrt{q},\ q + 1 + \sqrt{q}]$, the orders of elliptic curves are close to evenly distributed.

- For most elliptic curves (except e.g. super-singular curves), no DL-algorithm faster than the generic algorithms is known.

DH-DL equivalence [M94]

Let q be a divisor of n. We consider computing x modulo q.

Equality mod q: $x_i \equiv x_j \pmod{q}$ iff $(n/q)x_i = (n/q)x_j$

Inverses mod q: $x_i^{q-2} \equiv x_i^{-1} \pmod{q}$

Legendre symbol mod q: $x_i^{(q-1)/2}$

DH-DL equivalence [M94]

Let q be a divisor of n. We consider computing x modulo q.

Equality mod q: $x_i \equiv x_j \pmod{q}$ iff $(n/q)x_i = (n/q)x_j$

Inverses mod q: $x_i^{q-2} \equiv x_i^{-1} \pmod{q}$

Legendre symbol mod q: $x_i^{(q-1)/2}$

A first idea: x is determined by $(x + a/q)$ for sufficiently many a, but it appears infeasible to compute x from these values.

DH-DL equivalence [M94]

Let q be a divisor of n. We consider computing x modulo q.

Equality mod q: $x_i \equiv x_j \pmod{q}$ iff $(n/q)x_i = (n/q)x_j$

Inverses mod q: $x_i^{q-2} \equiv x_i^{-1} \pmod{q}$

Legendre symbol mod q: $x_i^{(q-1)/2}$

A first idea: x is determined by $(x + a/q)$ for sufficiently many a, but it appears infeasible to compute x from these values.

Square roots mod q: There exist algebraic square root algorithms. For instance, if $q \equiv 3 \pmod{4}$, then $x_i^{(q+1)/4}$ is a square root of x_i (provided x_i is a square mod q).

Auxiliary group technique:

1. Embed x in (the coordinate of) an element a_x of a finite **auxiliary group** H defined algebraically over $GF(q)$.

   In other words, $a_x = (x_i, x_j, \ldots, x, \ldots, x_k)$ for some $i, j, k, \ldots$.

2. Use the black-box's equality info. to compute a_x explicitly.

3. Extract x from a_x.

Auxiliary group technique:

1. Embed x in (the coordinate of) an element a_x of a finite **auxiliary group** H defined algebraically over $GF(q)$.

   In other words, $a_x = (x_i, x_j, \ldots, x, \ldots, x_k)$ for some $i, j, k, \ldots$.

2. Use the black-box's equality info. to compute a_x explicitly.

3. Extract x from a_x.

Implementation of Step 2:

- Compute the discrete logarithm z of a_x in H with respect to a generator h of H (if H is cyclic), using a generic algorithm (Pohlig-Hellman): $h^z = a_x$ in H

- Compute $a_x = h^z$ explicitly.

Auxiliary group technique:

1. Embed x in (the coordinate of) an element a_x of a finite **auxiliary group** H defined algebraically over $GF(q)$.

   In other words, $a_x = (x_i, x_j, \ldots, x, \ldots, x_k)$ for some $i, j, k, \ldots$.

2. Use the black-box's equality info. to compute a_x explicitly.

3. Extract x from a_x.

Implementation of Step 2:

- Compute the discrete logarithm z of a_x in H with respect to a generator h of H (if H is cyclic), using a generic algorithm (Pohlig-Hellman): $h^z = a_x$ in H

- Compute $a_x = h^z$ explicitly.

Note: This is efficient only if the order of H is smooth.

Example: cyclic elliptic curve

Theorem (Rück, 1987)**:** For every $d \in [q - 2\sqrt{q} + 1, q + 2\sqrt{q} + 1]$, there exists a cyclic elliptic curve over $GF(q)$ of order d.

Example: cyclic elliptic curve

Theorem (Rück, 1987)**:** For every $d \in [q-2\sqrt{q}+1, q+2\sqrt{q}+1]$, there exists a cyclic elliptic curve over $GF(q)$ of order d.

1. Choose a smooth $d \in [q-2\sqrt{q}+1, q+2\sqrt{q}+1]$.
   (Such a d exists under a plausible number-theoretic conjecture.)
2. Find the parameters $A, B \in GF(q)$ of a cyclic elliptic curve $E_{A,B}(q)$ with order d, together with a generator P.
3. Test $x, x+1, \ldots$ for quadratic residuosity mod q until $x+e$ is a quadratic residue $((x+e/q) = 1)$.
4. Compute y such that $y^2 \equiv x^3 + Ax + B \pmod{q}$.
5. Using the Pohlig-Hellman generic algorithm, compute the DL z of the point $(x+e, y)$ to the base P in $E_{A,B}(q)$.
   This is efficient because $|E_{A,B}(q)| = d$ is smooth.
6. Compute $(x', y) := z \cdot P$ and $x := x' - e$.

A number-theoretic conjecture

Let $p_1(n)$ denote the largest prime factor of n, and let

$$\nu(n,t) := \min_{n-t \le d \le n+t} p_1(d).$$

Let $\psi(n,m)$ be the number of integers in $[1,n]$ that contain no prime factor greater than m.

Theorem (Canfield, Erdös, Pomerance, 1983)**:** For any fixed u,

$$\psi(n, n^{1/u}) = \frac{n}{u^{(1+o(u))u}}.$$

Example: $\lim_{n\to\infty} \frac{\psi(n,n^{1/2})}{n} = 1 - \ln 2 = 0.31.$

Heuristic reasoning:

- Assumption: This also holds if u is a moderately growing function of n.
- If for some b the fraction $\psi(n,b)/n$ of b-smooth integers $\le n$ is substantially greater than $1/t$, then it is highly probable that there exists at least one b-smooth integer in the interval $[n-t, n+t]$, i.e., $\nu(n,t) \le b$.

Let $t = n^{1/s}$ and $b = \log^k n$. Setting $b = n^{1/u}$ we obtain $u = \log n/(k \log\log n)$ and hence

$$\psi(n, \log^k n)/n \;\approx\; u^{-u} \;=\; \left(\frac{\log n}{k \log\log n}\right)^{-\frac{\log n}{k \log\log n}} \;\gg\; n^{-1/k}$$

because

$$\log(u^u) = \frac{\log n}{k \log\log n}(\log\log n - \log\log\log n - \log k) < \frac{\log n}{k}.$$

For $k > s$ there exist on the order of $n^{1/s-1/k} \gg 1$ $(\log^k n)$-smooth integers in the interval $[n - n^{1/s}, n + n^{1/s}]$. (We need the case $s = 2$.)

Conjecture: $\nu(n, n^{1/s}) = (\log n)^{(1+o(s))s}$.

Stronger form: $\nu(n, n^{1/s}) = (\log n)^{s+o(s)}$.

Weaker, but still sufficient forms: $\nu(n, n^{1/s}) = (\log n)^{s^{O(1)}}$ or, for some function f: $\nu(n, n^{1/s}) = (\log n)^{f(s)}$.

Construction of groups G with CDH-DL equivalence

For every prime factor q_i of $|G|$, we need a suitable AG H_i.

Construction of groups G with CDH-DL equivalence

For every prime factor q_i of $|G|$, we need a suitable AG H_i.

Types of equivalence results:

- Given $|G|$, one can efficiently construct the AGs.
- The designer of G knows suitable AGs, but they may be difficult to find for anyone else.
- The designer of G knows that AGs exist, without knowing them.

Construction of groups G with CDH-DL equivalence

For every prime factor q_i of $|G|$, we need a suitable AG H_i.

Types of equivalence results:

- Given $|G|$, one can efficiently construct the AGs.
- The designer of G knows suitable AGs, but they may be difficult to find for anyone else.
- The designer of G knows that AGs exist, without knowing them.

Construction of G of third type:

1. Choose a large smooth number m.
2. Find a prime q such that $m \in [q - 2\sqrt{q} + 1, q + 2\sqrt{q} + 1]$.
3. Construct a group of order q (e.g. $\mathbf{Z}_p^*$ for some p with $q|(p-1)$.)

Construction of groups G with CDH-DL equivalence

For every prime factor q_i of $|G|$, we need a suitable AG H_i.

Types of equivalence results:

- Given $|G|$, one can efficiently construct the AGs.
- The designer of G knows suitable AGs, but they may be difficult to find for anyone else.
- The designer of G knows that AGs exist, without knowing them.

Construction of G of third type:

1. Choose a large smooth number m.
2. Find a prime q such that $m \in [q - 2\sqrt{q} + 1, q + 2\sqrt{q} + 1]$.
3. Construct a group of order q (e.g. $\mathbf{Z}_p^*$ for some p with $q|(p-1)$.)

Alternative: Use (Lay, Zimmer, 1994) to construct an elliptic curve of given order m together with the field $GF(q)$ over which it is defined.

Breaking RSA Generically is Equivalent to Factoring

Divesh Aggarwal, Ueli Maurer

ETH Zurich, www.crypto.ethz.ch

Number Theory and Computational Cryptography
Sept. 29 - Oct. 2, 2010, Warsaw.