COS 433: Cryptography

Announcements:

- Problem Set 5 due 12/2

This week: Zero-Knowledge Proofs

Proofs



π



Prover

Verifier

CS insight: think of a proof as a form of computation

Proofs



Claim: $\forall x \forall y \forall z (\neq 0)$,
$\forall n > 2, x^n + y^n \neq z^n$

π



Prover

Verifier

CS insight: think of a proof as a form of computation

E.g.: AW writes a proof in a paper, Bob can then read the paper and check it.

Proofs



Claim: $\forall x \forall y \forall z (\neq 0)$,
$\forall n > 2, x^n + y^n \neq z^n$



π

Prover

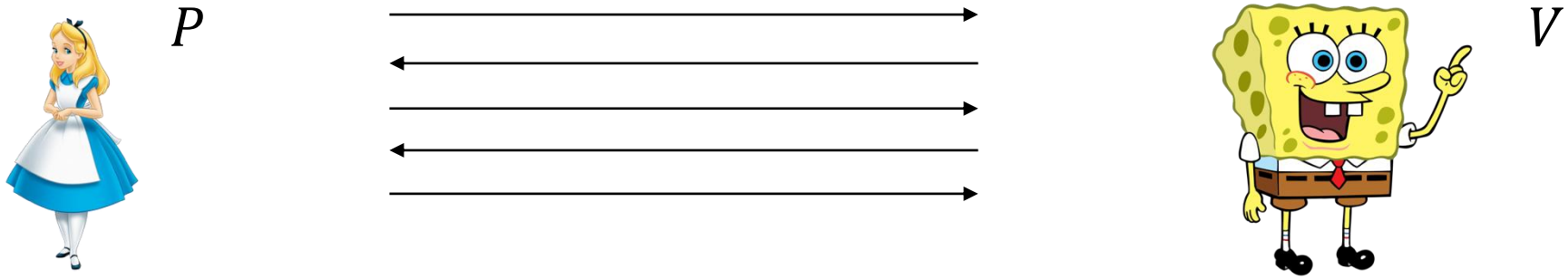
Verifier



A seminar talk:

- Does not provide an entire, formal proof
- Goal (one of many): convince the audience that a formal proof exists
- Leverages audience interaction to make the proof easier to check

Proof Systems



Definition: a proof system consists of a pair of interactive algorithms (P, V)

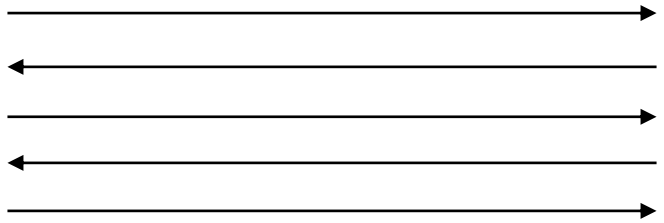
- V will always be a polynomial-time algorithm
- Polynomial with respect to what?

Proof Systems

Claim: $x \in L$



$P(x, w)$





$V(x)$

Output: $b \in \{0,1\}$

Definition: a proof system for a language $L \subset \{0,1\}^*$ consists of a pair of interactive algorithms (P, V) run on an input $x \in \{0,1\}^n$

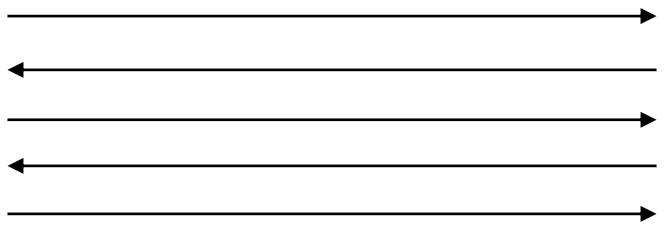
- V will always be a $\mathbf{poly}(n)$-time algorithm
- P will be either unbounded, or polynomial-time given some additional information $w \in \{0,1\}^{\mathbf{poly}(n)}$ ("efficient prover")

Proof Systems

Claim: $x \in L$



$P(x)$





$V(x)$

Output: $b \in \{0,1\}$

Definition: a proof system for a language $L \subset \{0,1\}^*$ consists of a pair of interactive algorithms (P, V) run on an input $x \in \{0,1\}^n$

- V will always be a $\mathbf{poly}(n)$-time algorithm

It must satisfy the following properties:

1. **Completeness:** if $x \in L$ and the two parties run (P, V), then V outputs 1 (accept)
2. **Soundness:** if $x \notin L$, then for any strategy P^*, V accepts with probability $\mathbf{negl}(n)$.

Note: we allow for P, V to be randomized.

Proof Systems: Examples

Claim: $N \in L$



$P(N)$ $\xrightarrow{(c,d)}$ $V(N)$



Check that $c, d > 1$ and $N = cd$

Example: composite integers

- $L = \{N : N \text{ is composite}\}$
- $P(N)$: factor $N = c \cdot d$, send (c, d)
- $V(N, c, d)$: check that $N = cd$ and $c, d > 1$

Use case: Alice could generate and release $N = pq$, and prove that it is composite at any later time by sending (p, q)

Proof Systems: Examples

Claim: $(N, y) \in L$  $P(N, y)$

x →

 $V(N, y)$

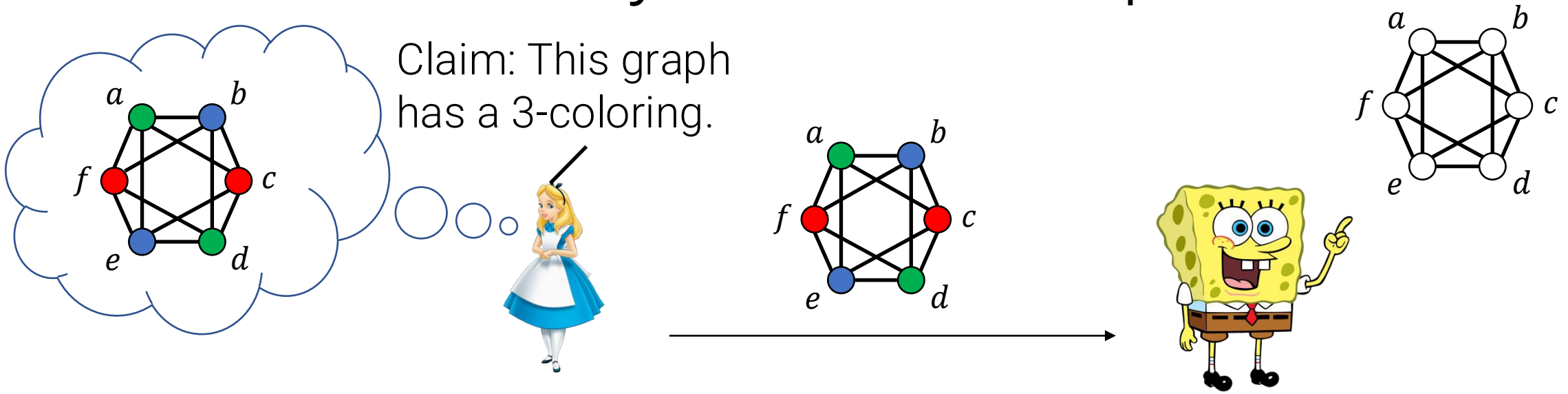
Check that $y \equiv x^2 \pmod{N}$

Example: quadratic residuosity

- $L = \{(N, y): y \text{ is a square mod } N\}$
- $P(N)$: find and send square root x
- $V(N, y, x)$: check that $y \equiv x^2 \pmod{N}$

Use case: Alice could generate and release $N = pq$, and (efficiently) prove that any square y is a perfect square (root finding is easy given factorization).

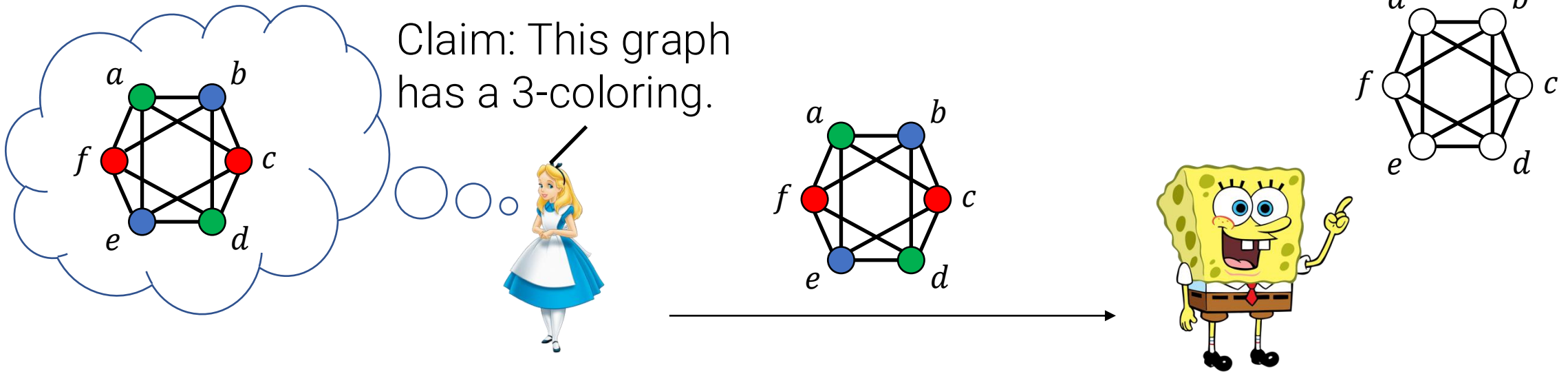
Proof Systems: Examples



Example: graph 3-coloring

- $L = \{\text{graph } G : G \text{ is 3-colorable}\}$
- $P(G)$: find and send color assignment $c : \text{Vert}(G) \to \{0, 1, 2\}$
- $V(G, c)$: check that all edges of G are bichromatic
  - Takes time $O(n^2)$ in the worst case

Proof Systems: Examples



These are all extremely simple proof systems!

- Non-interactive
- Deterministic

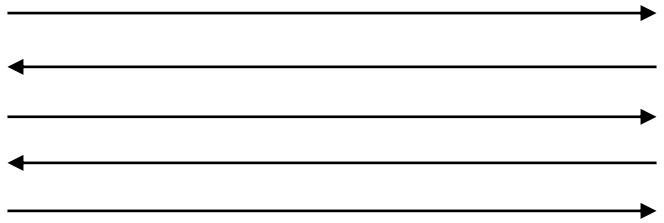
Definition: The complexity class $\mathbf{NP}$ is the set of all languages $L \subset \{0,1\}^*$ that have a deterministic, non-interactive proof system

Proof Systems: Examples

Claim: $x \in L$



$P(x, w)$





$V(x)$

Output: $b \in \{0,1\}$

Today and next time: focus on proof systems for languages $L \in \mathbf{NP}$

- For $x \in L$ and non-interactive proof system for L, call a valid proof a *witness* w
- Assume honest prover has a witness w.

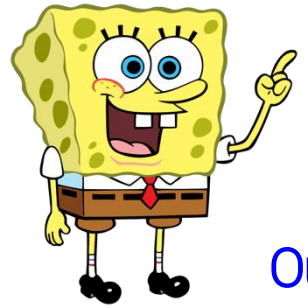
Proof Systems: Examples

Claim: $x \in L$



$P(x, w)$

w



$V(x)$

Output: $b \in \{0,1\}$

Today and next time: focus on proof systems for languages $L \in \mathbf{NP}$

- For $x \in L$ and non-interactive proof system for L, call a valid proof a *witness* w
- Assume honest prover has a witness w.

- Trivial proof system: P sends w, V checks that w is valid.

- We've done nothing…?

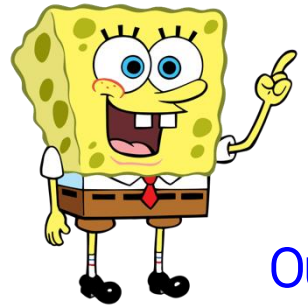
Proof Systems

Claim: $x \in L$



$P(x, w)$

w



$V(x)$

Output: $b \in \{0,1\}$

Where is the cryptography?

Idea: maybe w is supposed to be a secret!

- w could be Alice's secret key, x could be an associated public key
- w could contain sensitive data that Alice doesn't want to (or legally cannot) release

Proof Systems: Quadratic Non-Residuosity

Claim: $y \notin \mathrm{QR}_N$



$P(p, q, y)$

p, q



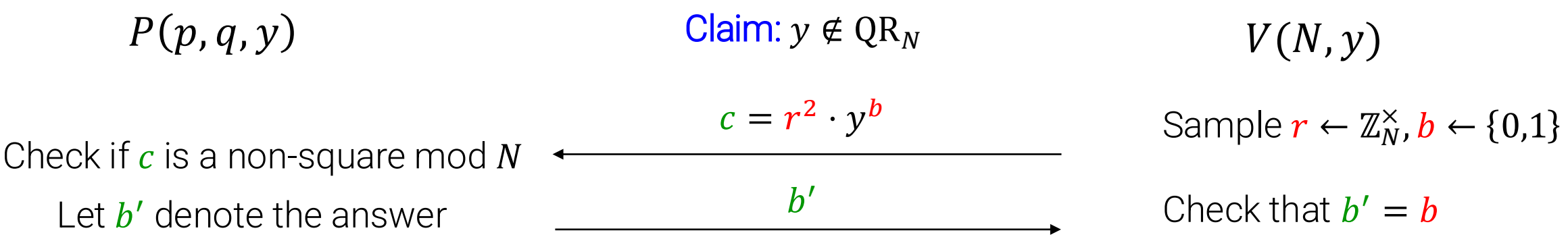
$V(N, y)$

Check $N = pq$, check if y is a square.

Can we prove y is not a square, without revealing the factorization of N?

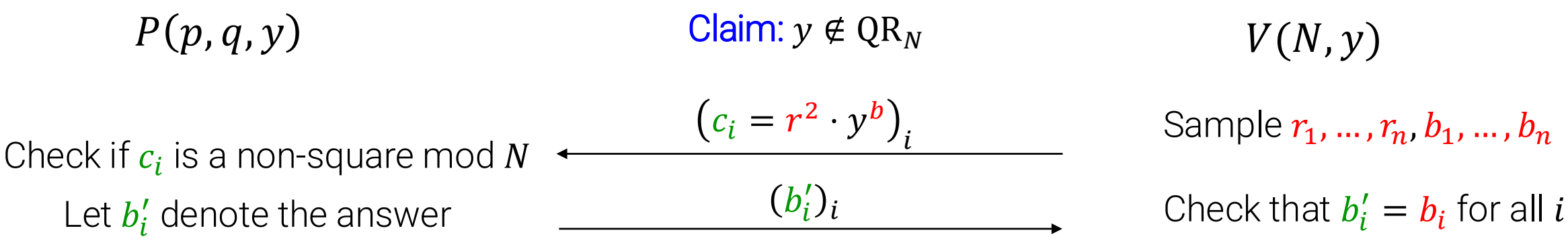
- Problem: verifier will not be able to decide if $y \in \mathrm{QR}_N$ on his own.
- Solution: leverage **interaction** and **randomness**

Proof Systems: Quadratic Non-Residuosity



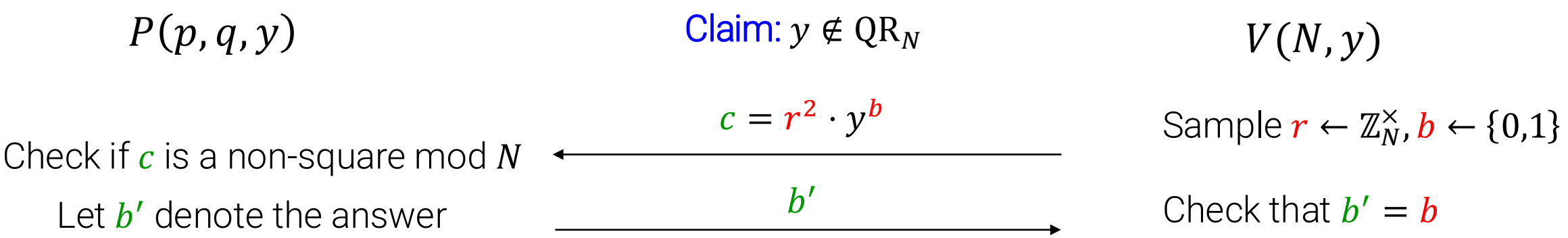
- **Completeness**: if y is not a square, then c is a non-square if and only if $b = 1$
- **Soundness**: if y is a square,
  - c is a random square mod N
  - so c hides the value of b
  - P^* can only output $b' = b$ with probability ½
  - "Soundness error ½"

Proof Systems: Quadratic Non-Residuosity



- **Soundness**: P^* can only output all $b_i' = b_i$ with probability 2^{-n}.

Proof Systems: Quadratic Non-Residuosity



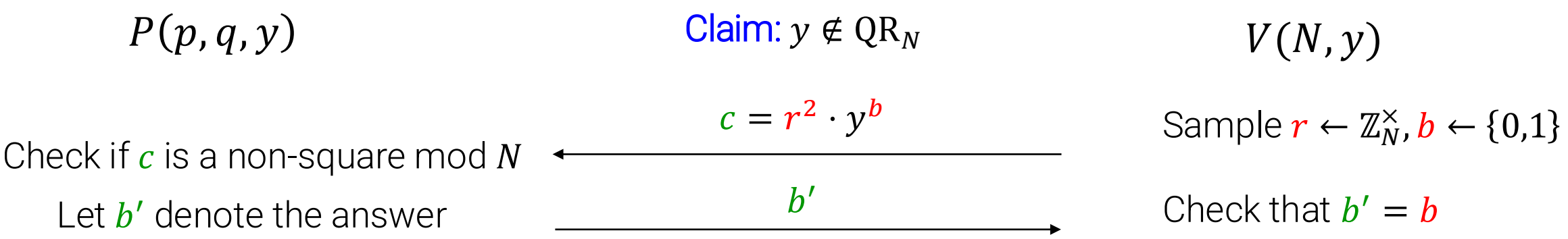
Question: can V learn the factorization of N after receiving a proof?

- Let's assume that V follows the protocol

Answer: no (assuming factoring N is hard), and we can prove this!

- Suppose V could output (p, q) additionally given (r, b, c, b')
- Algorithm $V(N, y, r, b, c, b') \rightarrow (p, q)$
- Then we can efficiently factor N: run $V(N, y, r, b, c, b)$

Proof Systems: Quadratic Non-Residuosity



Amazing property: V seems to learn **nothing** by receiving a proof!

- V does learn that y is a non-square
- But that's it!

Proof Systems: Quadratic Non-Residuosity

$P(p, q, y)$ | Claim: $y \notin \mathrm{QR}_N$ | $V(N, y)$

$c = r^2 \cdot y^b$ →

Check if c is a non-square mod N

Let b' denote the answer

Sample $r \leftarrow \mathbb{Z}_N^\times, b \leftarrow \{0,1\}$

b' →

Check that $b' = b$

We formalize this via a security property called (Honest-Verifier) Zero Knowledge

Proof Systems: Examples

Claim: $x \in L$



$P(x, w)$





$V(x)$

Output: $b \in \{0,1\}$

Definition: given a pair of interactive algorithms $P(x, w) \leftrightarrow V(x)$, we define the **view of V**, $\mathbf{view}_V(x, w)$, to consist of:

- The input x
- All random coins sampled by V, and
- All messages sent from P to V in the protocol

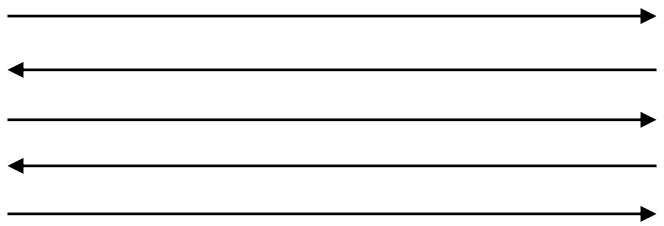
$\mathbf{view}_V$ is a distribution w.r.t. the randomness of P and V

Proof Systems: Examples

Claim: $x \in L$



$P(x, w)$





$V(x)$

Output: $b \in \{0,1\}$

Definition: a proof system (P, V) for a language L is honest-verifier zero knowledge if there exists a polynomial-time simulator

$$\mathrm{Sim}(x) \to \tau$$

such that for all $x \in L$ and all valid w,

$$\mathrm{Sim}(x) \approx_c \mathrm{view}_V(x, w)$$

Simulation paradigm: everything V learned, V could have generated on their own!

Proof Systems: Quadratic Non-Residuosity

$P(p, q, y)$ — Claim: $y \notin \mathrm{QR}_N$ — $V(N, y)$

Check if c is a non-square mod N

Let b' denote the answer

$c = r^2 \cdot y^b$ ←

b' →

Sample $r \leftarrow \mathbb{Z}_N^\times, b \leftarrow \{0,1\}$

Check that $b' = b$

Claim: this proof system is HVZK

Proof: define the simulator

$$\mathrm{Sim}(N, y) = \big((N, y), (r, b), b\big)$$

If y is a non-square, the output of $\mathrm{Sim}(N, y)$ is identical to $\mathrm{view}_V(N, y, p, q)$

Proof Systems: Quadratic Non-Residuosity

$P(p, q, y)$ | Claim: $y \notin \mathrm{QR}_N$ | $V(N, y)$

$c = r^2 \cdot y^b$ →

Check if c is a non-square mod N

Let b' denote the answer

b' →

Sample $r \leftarrow \mathbb{Z}_N^\times, b \leftarrow \{0,1\}$

Check that $b' = b$

Problem: what if we don't trust V to behave according to the protocol specification?

Malicious-Verifier Zero Knowledge

Claim: $x \in L$



$P(x, w)$





$V(x)$

Output: $b \in \{0,1\}$

Definition: a proof system (P, V) for a language L is malicious verifier zero knowledge if for all polynomial-time V^*, there exists a polynomial-time simulator

$$\mathrm{Sim}^*(x) \to \tau$$

such that for all $x \in L$ and all valid w,

$$\mathrm{Sim}^*(x) \approx_c \mathrm{view}_{V^*}(x, w)$$

Simulation paradigm: everything V^* learned, V^* could have generated on their own!

Proof Systems: Quadratic Non-Residuosity

$P(p, q, y)$ | Claim: $y \notin \mathrm{QR}_N$ | $V(N, y)$

$c = r^2 \cdot y^b$ →

Check if c is a non-square mod N

Let b' denote the answer

Sample $r \leftarrow \mathbb{Z}_N^\times, b \leftarrow \{0,1\}$

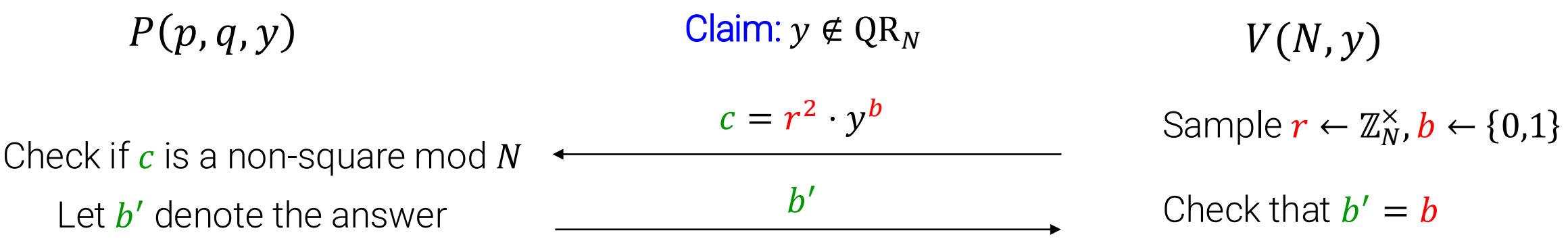
← b'

Check that $b' = b$

Is this protocol malicious-verifier ZK?

No!

Proof Systems: Quadratic Non-Residuosity



Is this protocol malicious-verifier ZK?

- Intuition: malicious V^* could send c to be any number mod N, where V^* does not know if c is a square modulo N
- Honest P will tell V^* this information, so V^* has learned something

Proof Systems: Quadratic Non-Residuosity

$P(p, q, y)$

Claim: $y \notin \mathrm{QR}_N$

$V(N, y)$

$c = r^2 \cdot y^b$

Check if c is a non-square mod N

Let b' denote the answer

b'

Sample $r \leftarrow \mathbb{Z}_N^\times, b \leftarrow \{0,1\}$

Check that $b' = b$

Magic: there's a way to change this protocol, to make it fully ZK!

A little too complicated; we will switch to a different example.