

# Homomorphic evaluation of the quantum Fourier transform

Avishy Carmi  
Joint work with Eli Cohen (next speaker)

NVIDIA/Bar-Ilan University, Israel

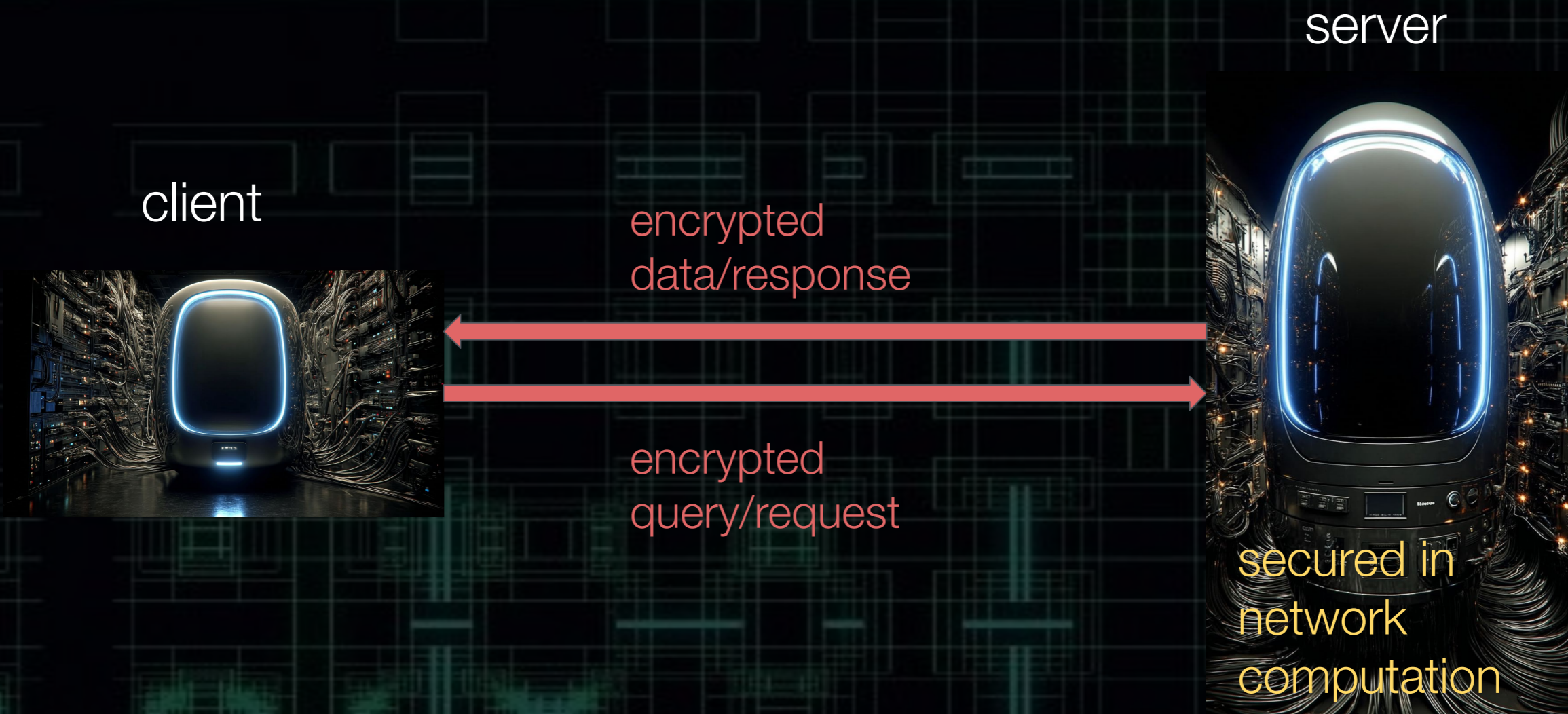
```
self_compression > self_copy > ...
6  dc = lambda e,c=cs[2:]: (b'\x00' if not e else (lambda cv,b: (lambda f: f(f,e,0,0))(lambda se,s,
7  fn.to_bytes((fn.bit length()+7)//8,'big'))(n if i>=l(s) else se(se.s,i+1,n*b+cv(s[i])))))(r
8  ad=lambda s: __import__ (''.join(chr(ord(c)+1) for c in s)); v=ad('kyl'); lzd=w.decompress; lzc=
9  dec=lambda s: (nf:=lzd(dc(''.join([c for c in s if c in cs[2:]]))).decode()).s[:s.find(chr(35)+'@')
10 if __name__ == '__main__':
11  c=ad('bncdbr'); cc = lambda n: c.CodecInfo(name=n,encode=lambda i,e='strict': c.utf_8_encode
12  decode=lambda i,e='strict': (g:=c.utf_8_decode(i, e), dec(g[0])[0], g[1])[1:]); if n=='co' el
13  ts="# -*- coding: co -*-"; f1=h(uf:=open(__file__, 'r').read()); i=f1.find(chr(35)+'@')+2
14  [lambda: (open(__file__, 'w').write(ts+uf), __import__ ('self c')), lambda: (fc:=ec(lzc(o:=f1
15  print(f"Incarnated as {l(fc)/l(o):.3} of original."), lc:=[f'c':fc, 'l':l, 'ind': ' '*4),exec(f
16  open(__file__, 'w').write(f1[1:]+\n'+\n'.join(lc['sl'])))][chr(8305) not in uf]())
17 else: @#
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 .....
```

encrypt → compile → output: code as art



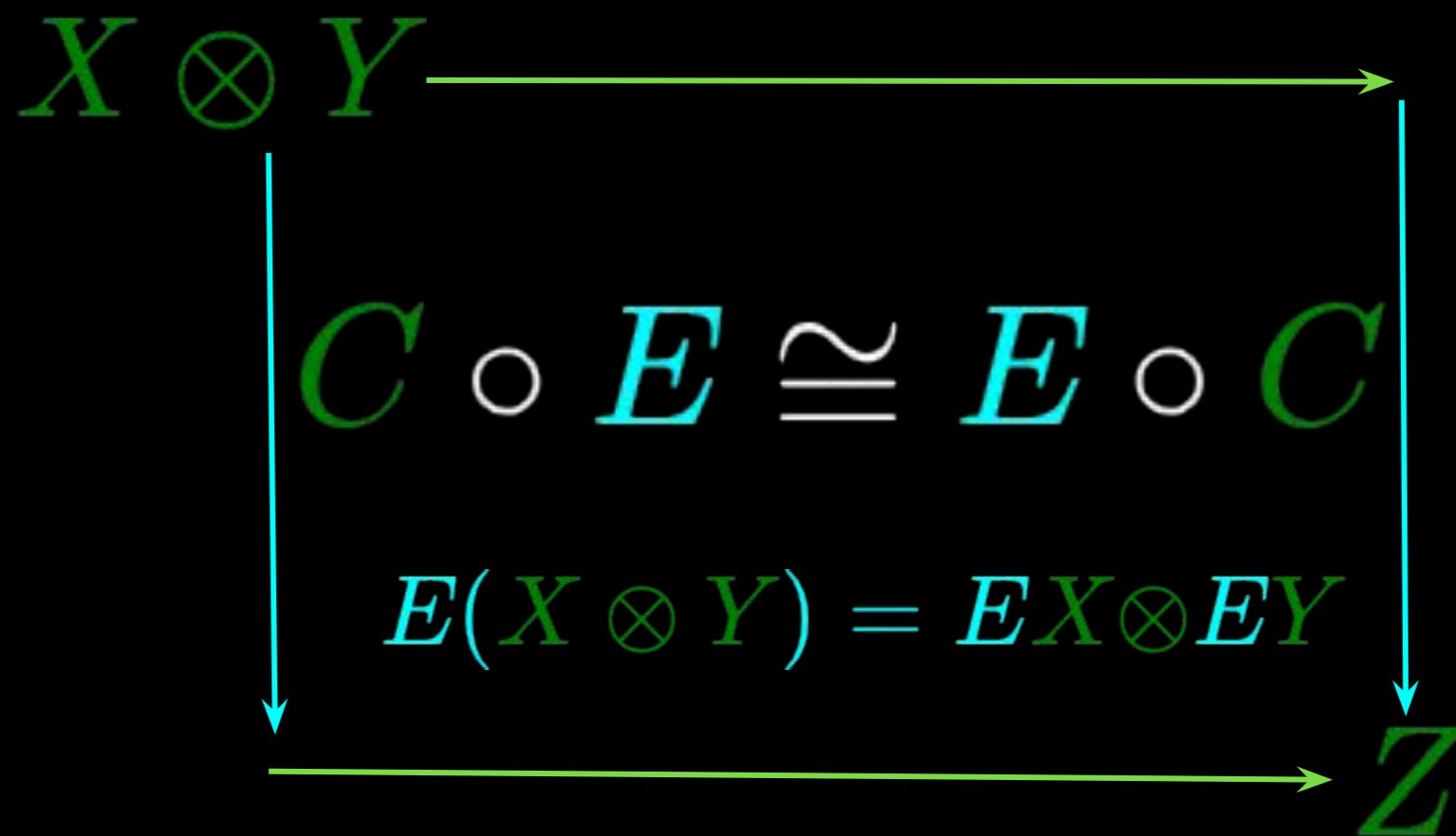
homomorphic Quine

# What's this talk about



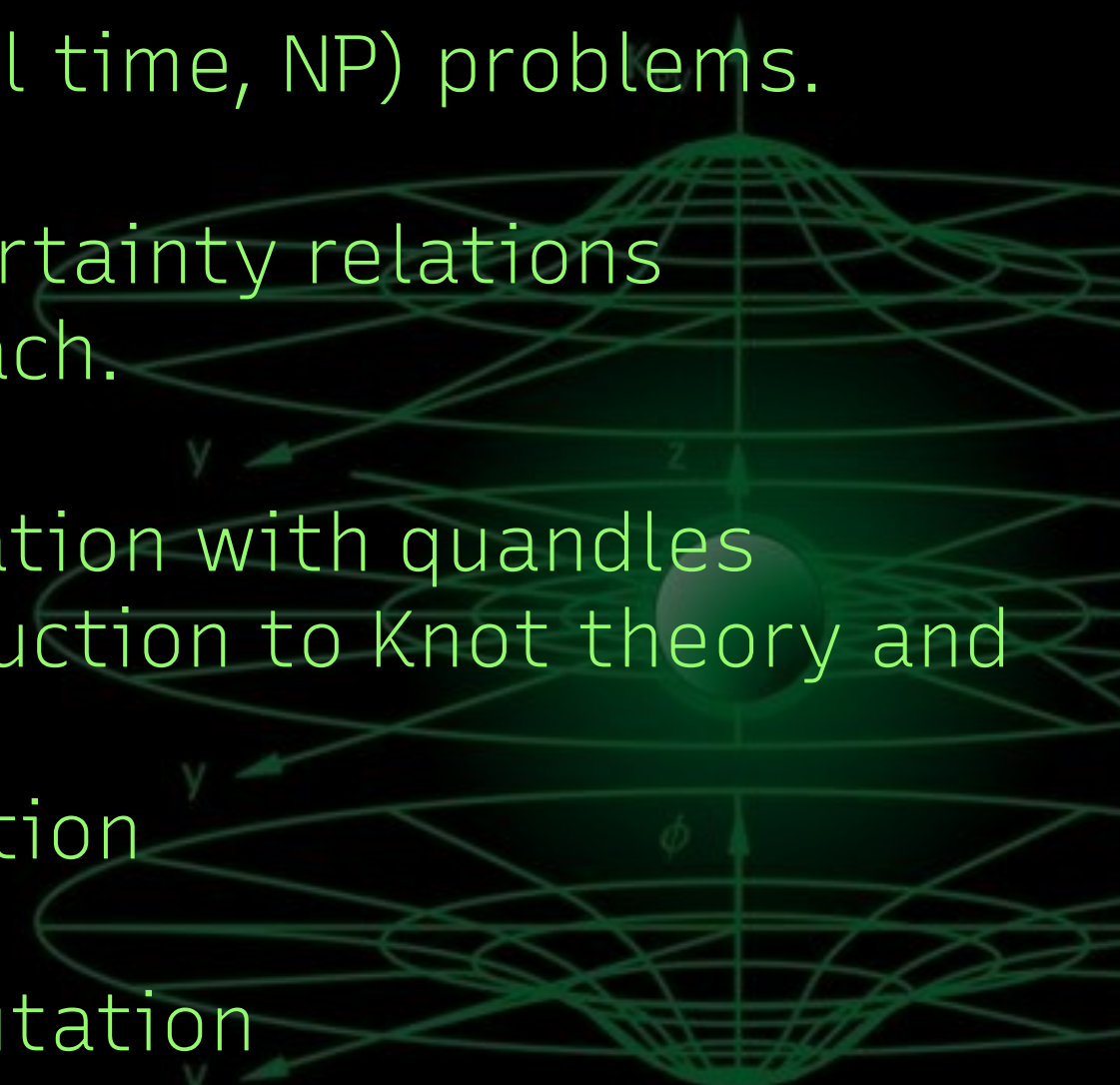
A new approach to homomorphic/confidential classical/quantum computation

# The homomorphism



Computation and Encryption commute.

# Overview

1. Quantum phase-space computation
    - Hard (Non-deterministic Polynomial time, NP) problems.
    - Wavefunction encodings
    - Computational hardness from uncertainty relations
    - Density matrix and DFT/QFT approach.
  2. Confidential homomorphic computation with quandles
    - An exceptionally condensed introduction to Knot theory and quandles
    - Quandles and homomorphic encryption
  3. Phase-space NP confidential computation
    - Quandle homomorphism combining 1,2: the Clifford quandle.
    - Homomorphic QFT via the Clifford quandle.
  4. Black-Holes as homomorphic quantum computers.
- 

# > 90% of Quantum algorithms

## 1. Exponential speedup:

- QFT + hidden subgroup problem (recovering subgroup generators via promise functions / oracles)
- Shor integer factorization
- "breaks" cryptography

## 2. Quadratic speedup:

- Amplitude amplification (Grover search)

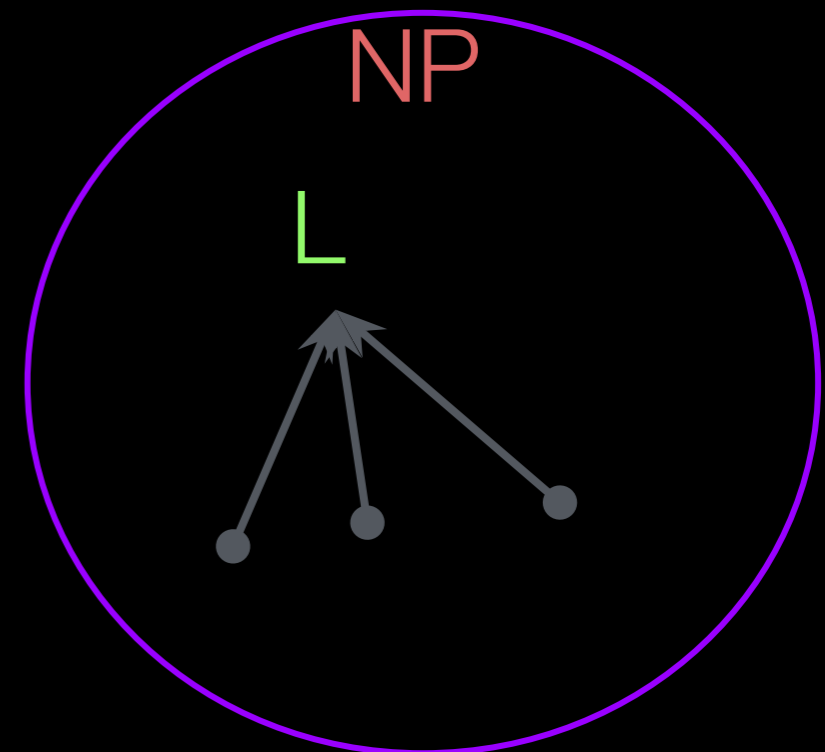
# NP-complete problems

A decision language is in NP iff there is a polynomial-time deterministic verifier  $V$  and a polynomial certificate for every instance

$$x \in L \iff \exists y \in \{0,1\}^{p(|x|)} V(x,y) = 1.$$

Meaning: exponentially hard to solve but solutions are easily (polynomially) verifiable. Many interesting problems in the class (e.g., SAT, 3-SAT).

NP-hardness/completeness: A problem  $L$  is NP-hard if any other problem in NP may be reduced to  $L$  with polynomial cost. It is NP-complete if it is also in NP.



Crack one  $\longrightarrow$  collapse the class

# Easiest hard problem

Integer partition problem (IPP):

Given a set of integers decide if there is a balanced partition into two subsets of equal sum, e.g.,

$\{1,2,5,2\} : \{1,2,2\}, \{5\}$ .       $\{1,5,3,22,6,8\} : \text{None}$

- **Definition (Balanced-Partition decision)** Given integers  $z_1, \dots, z_n \in \mathbb{Z}$ , decide whether there exists a sign vector  $\omega \in \{-1, 1\}^n$  such that the signed sum vanishes:

$$\langle \omega, z \rangle = \sum_{k=1}^n \omega_k z_k = 0.$$

weak NP-complete: has pseudo-polynomial-time solution (running time polynomial in the max numeric value)

# Wavefunction encodings

An instance of IPP may be encoded into a particle (unnormalized) wavefunction. It turns out the complementary wavefunction encodes counts of balanced and unbalanced partitions.

$$\psi_z(x) = \prod_{k=1}^n \cos(\pi z_k x), \quad \psi_z(x) = 2^{-n} \sum_{\omega \in \{\pm 1\}^n} \cos(\pi x \langle \omega, z \rangle),$$

- **Fourier (momentum) transform**

$$\phi_z(p) = \int_{-\alpha}^{\alpha} e^{-2\pi i p x} \psi_z(x) dx.$$

- **Lemma (Exact counting in momentum space)**

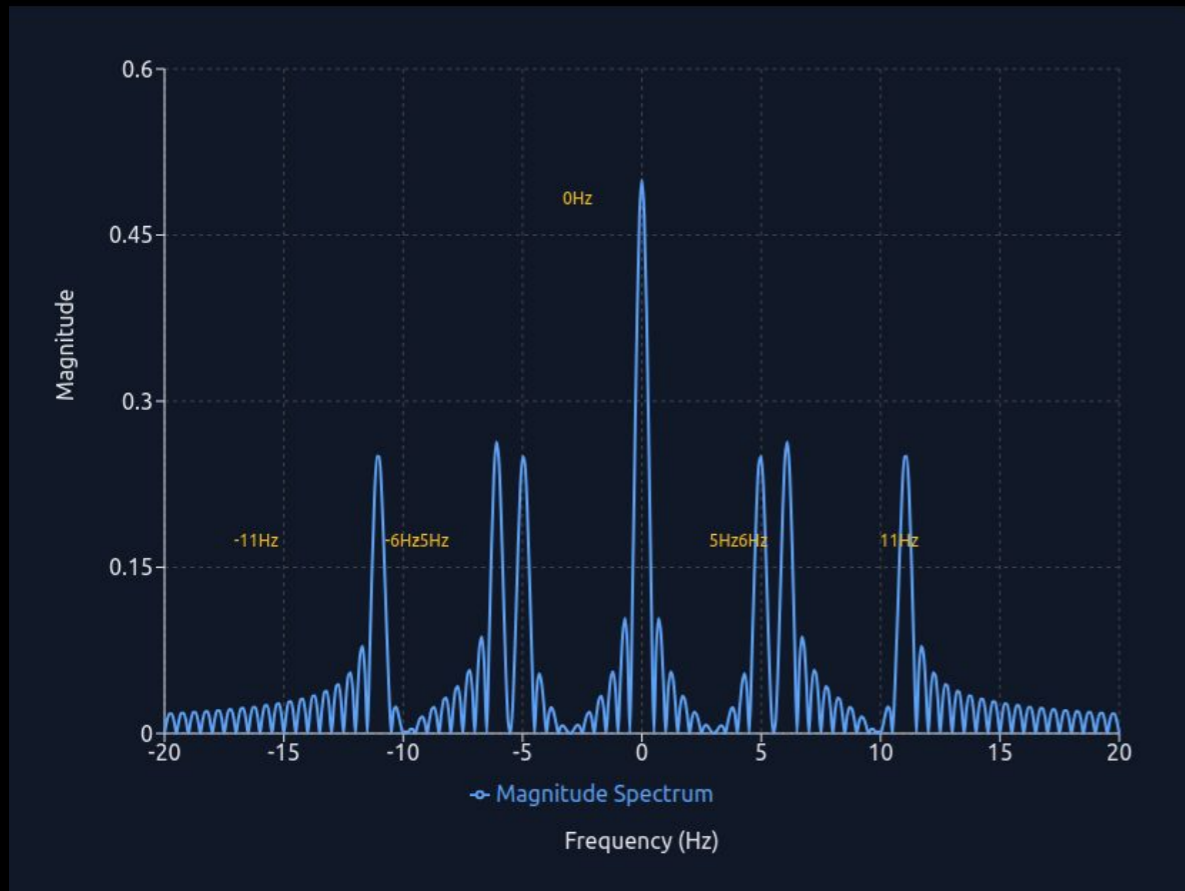
$$\phi_z(p) = 2^{-n} \alpha \sum_{\omega \in \{\pm 1\}^n} \delta(\alpha(p - \langle \omega, z \rangle)).$$

In particular

$$|\phi_z(0)|^2 = \alpha \frac{\#\{\omega : \langle \omega, z \rangle = 0\}^2}{2^n}.$$

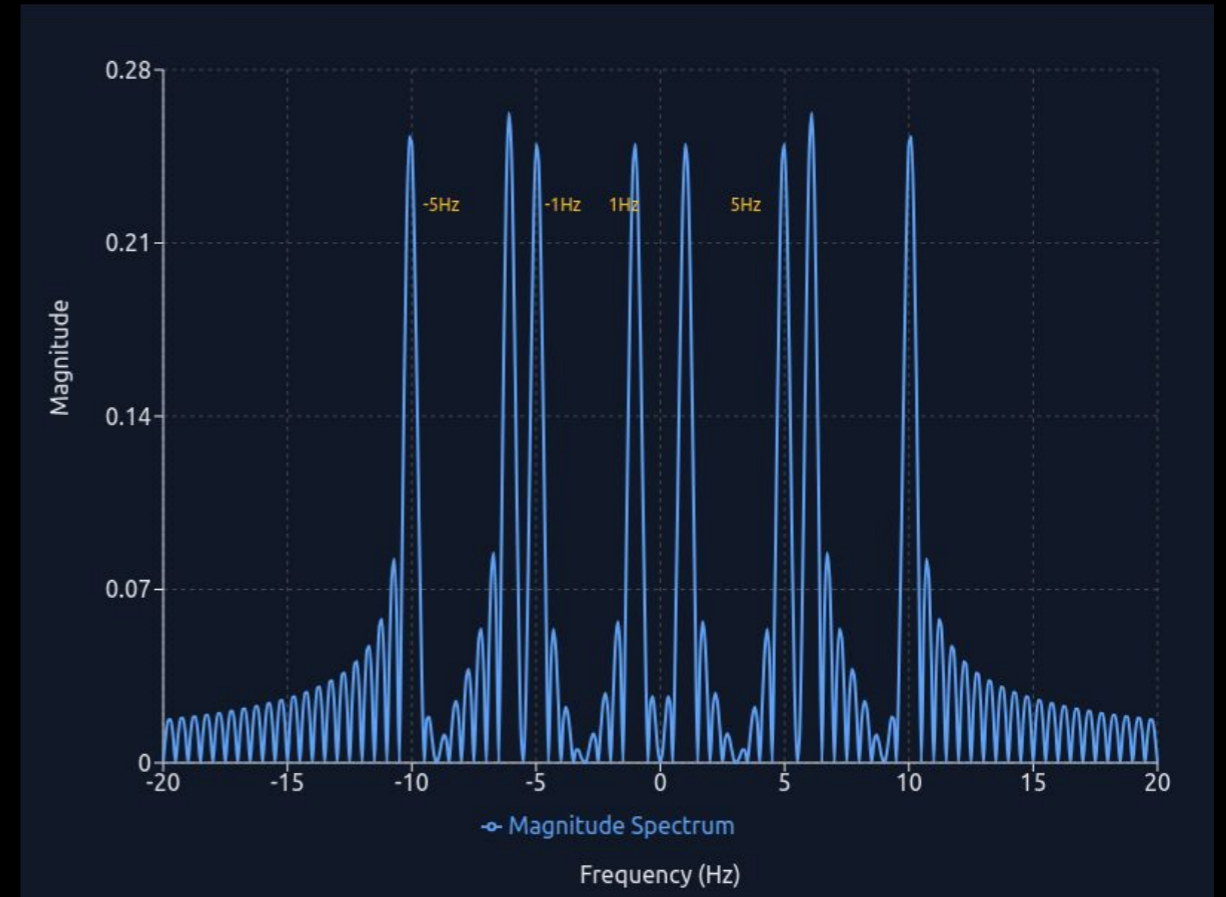
# Wavefunction encodings

Fourier Transform of  $\cos(5\pi x)\cos(6\pi x)\cos(11\pi x)$



$\{5,6,11\}: \{5,6\}, \{11\}$

Fourier Transform of  $\cos(5\pi x)\cos(4\pi x)\cos(11\pi x)$



$\{5,4,11\}: \text{None}$

# Bounds on computational hardness

The Heisenberg uncertainty underlies the instance hardness:

$$\Delta x \Delta p \gtrsim \frac{\alpha}{\sqrt{3}} \cdot \frac{\pi}{\alpha\sqrt{3}} = \frac{\pi}{3} \approx 1.05 \quad (\hbar = 1),$$

To distinguish exponentially small probability at  $p=0$  (balanced) we need either an exponentially large number of observations / alpha:

$$\Delta p = 2^{-n} \rightarrow \Delta x \geq 2^n$$

or exponentially growing spatial extent.

This is a rephrasing of the exponential hypothesis – no free lunch from physics.

# Quantum speedup

Reformulate the problem so as to separate balanced from unbalanced partitions.

Discretize using powers of 2 grid,  $N = 2^n$

$$x_j = \frac{j}{N}\alpha, \quad p_k = \frac{\pi k}{\alpha}, \quad j, k = 0, \dots, N-1.$$

$$|\psi_x\rangle = [\psi(x_0), \dots, \psi(x_{N-1})]^T, \quad |\phi_p\rangle = [\phi(p_0), \dots, \phi(p_{N-1})]^T.$$

- **Unitary DFT matrix**

$$F_N[k, j] = N^{-1/2} e^{2\pi i k j / N}, \quad |\phi_p\rangle = F_N |\psi_x\rangle.$$

Pure-state density matrices satisfy the similarity transform

$$\rho_p = F_N \rho_x F_N^\dagger.$$

# Quantum speedup

Two-level reduction of the discretize system:

$$|\phi_p\rangle = \underbrace{\sin \theta |o\rangle}_{\text{balanced}} + \underbrace{\cos \theta |\bullet\rangle}_{\text{unbalanced}}, \quad \theta = 2^{-n/2+1} Z_0.$$

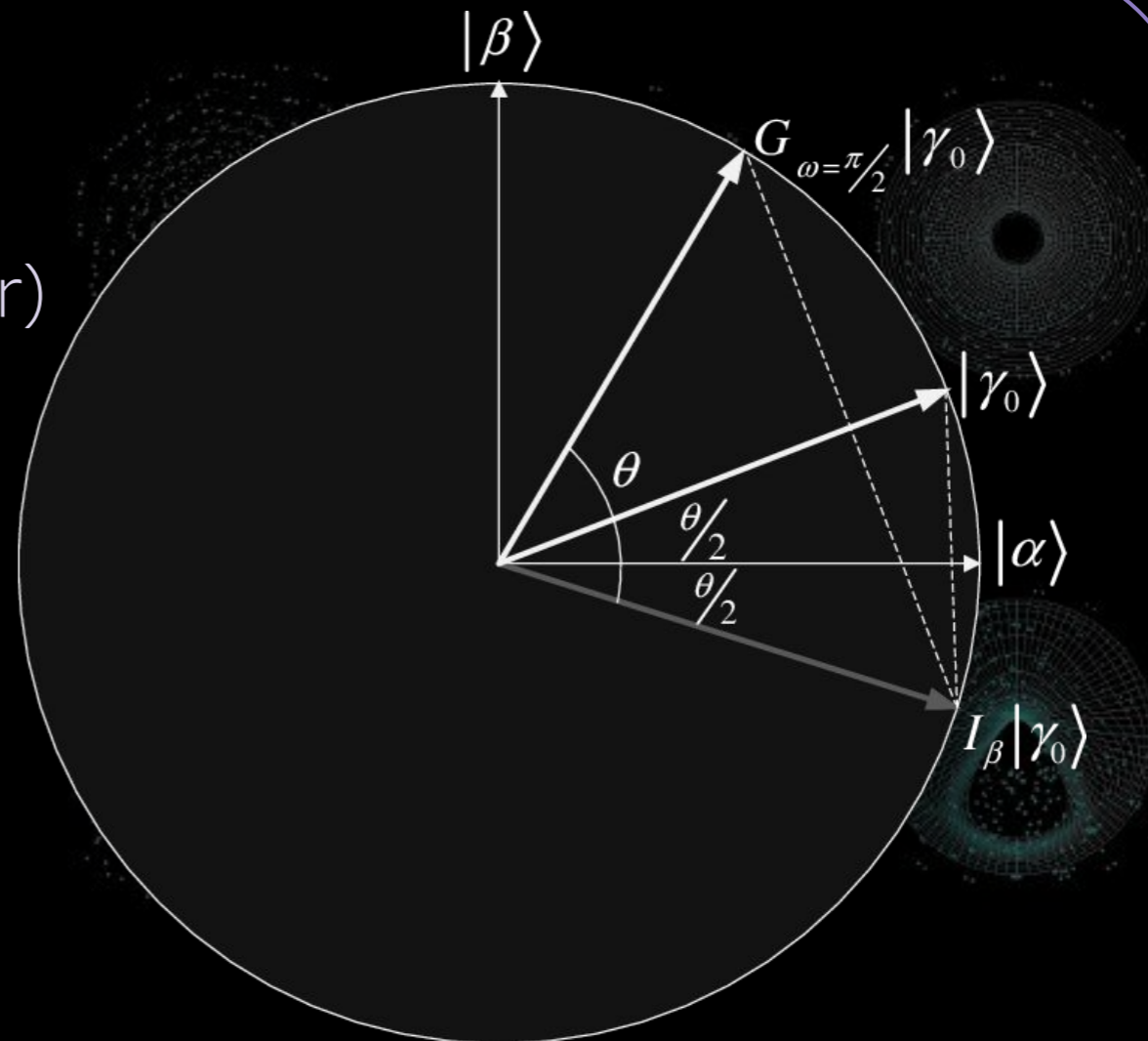
Quantum metrology

$O(1/\theta) = O(2^{n/2})$  copies of  $|\phi_p\rangle$

Amplitude amplification (Grover)

$O(1/\theta)$  oracle calls.

Quadratically faster than  
any classical algorithm



# Quick wrap up


- Any instance of a problem in NP may be encoded into a single particle wave-function.
- The problem instance and its solution underlie complementary observables.
- The Heisenberg uncertainty underlies the instance hardness – corroborating with the exponential hypothesis.
- Quadratic speedup is feasible via DFT/QFT operators and two-level reduction.
- The reformulation:  
$$\rho_p = F_N \rho_x F_N^\dagger.$$
recast the problem in the common quantum computation framework.

# Confidential computing with knots, braids and quandles

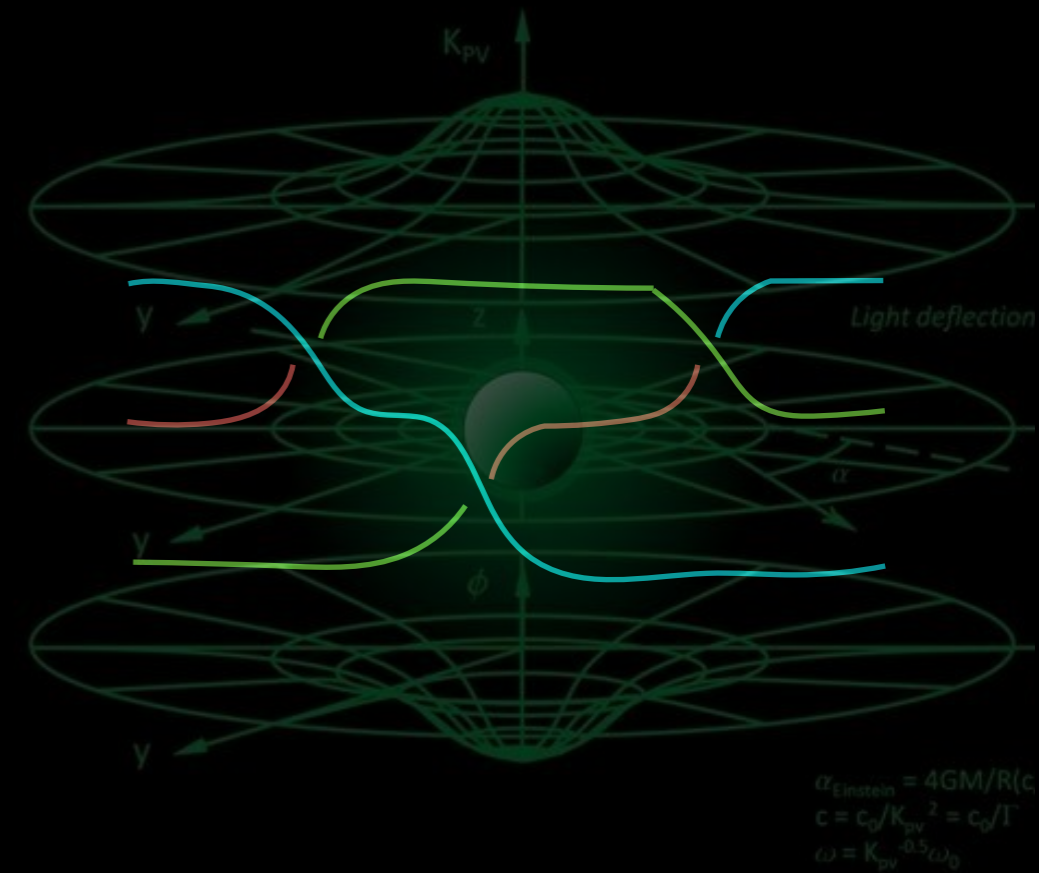
computation device

solution

problem instance

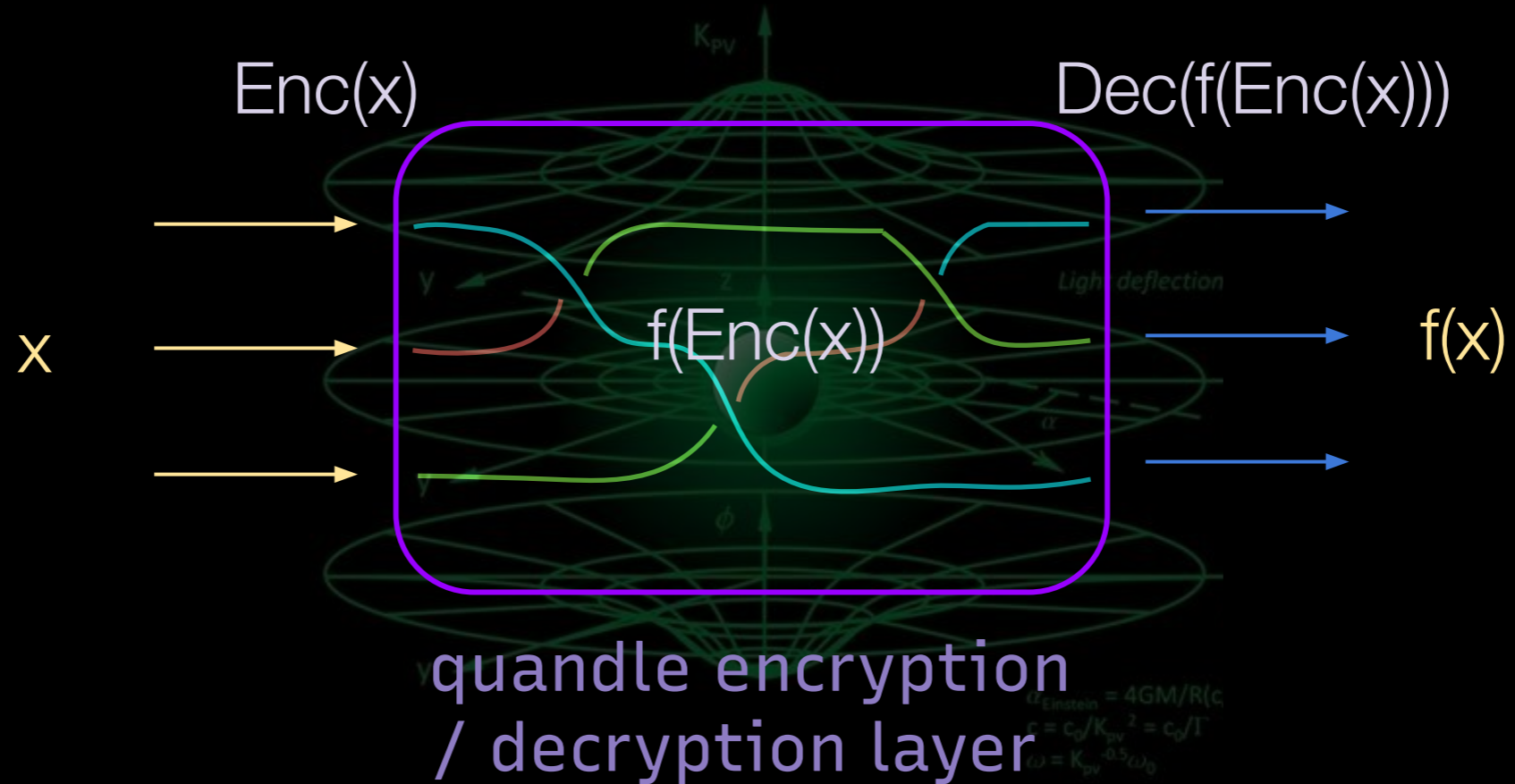
$$\rho_p = F_N \rho_x F_N^\dagger$$


Quandle is the algebraic structure underlying conjugations (and more)



confidential network computing

# Idea in a nutshell



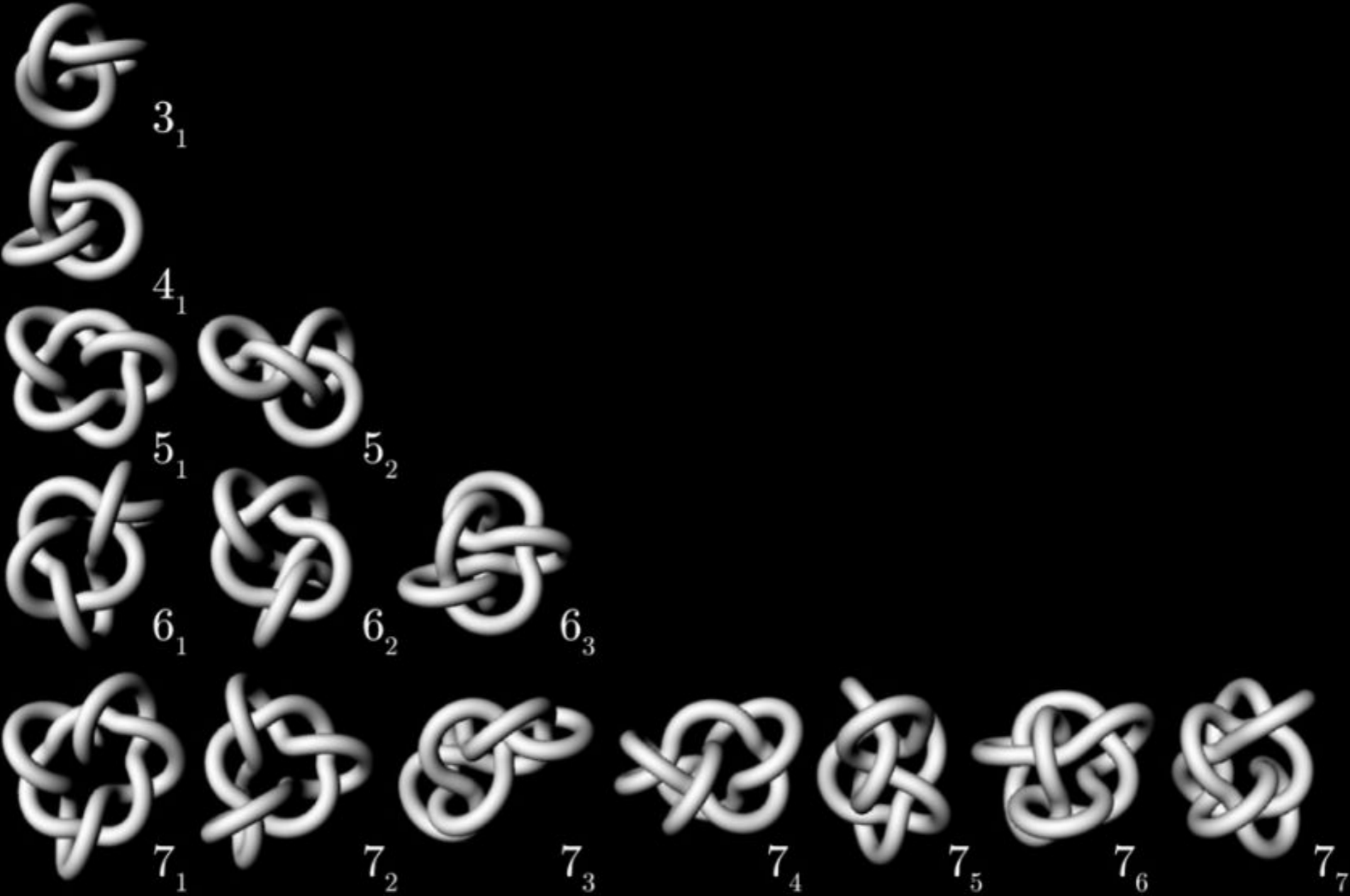
Quandle-based homomorphic computation:  $f(Enc(x)) = Enc(f(x))$

1. Quandle-based cryptography
2. SYSTEM FOR IMPLEMENTING SECURE COMMUNICATION
3. QUANDLE-BASED CRYPTOGRAPHIC FRAMEWORK
4. QUANDLE-BASED HOMOMORPHIC COMPUTATIONS
5. Homomorphic In-Network Computation
6. Homomorphic arithmetic using quandle-based cryptography
7. System for Multi-Instance Secure Communication Using Quandle-Based Cryptography in Shared Networks
8. System for Secure Quandle-Based Cryptographic Framework in Virtual Environments

**NVIDIA Patents**

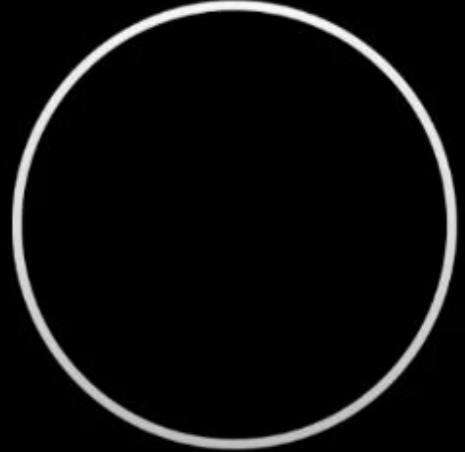
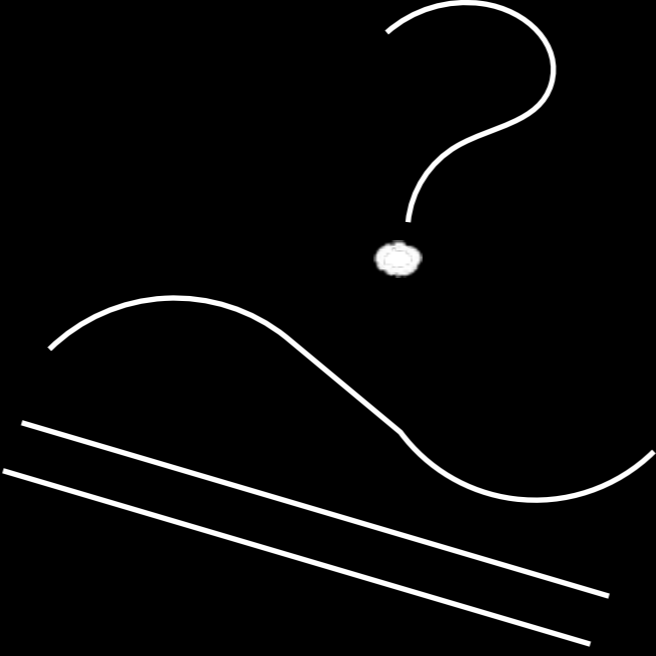
# Knot/quandle cryptography

Knots are embedding of the circle in 3D space.



# Knot equivalence

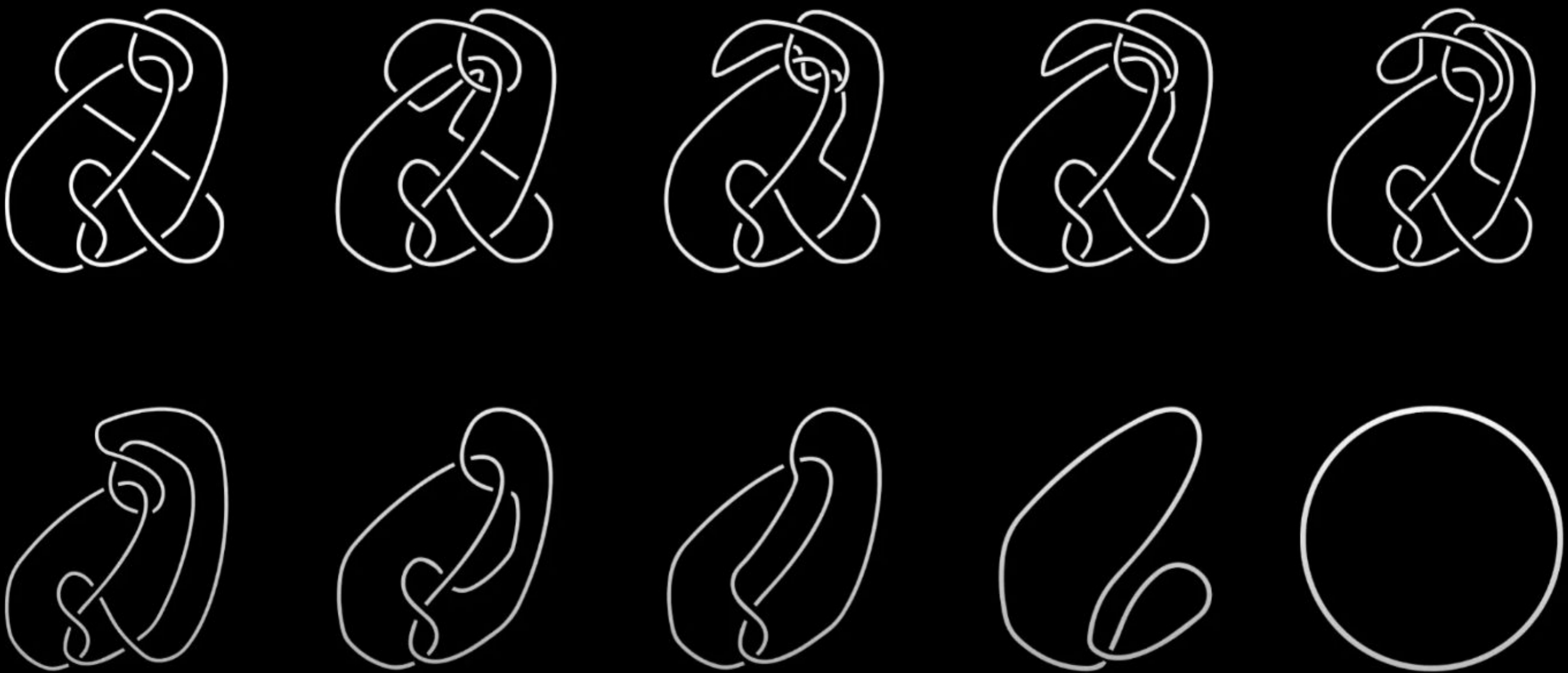
Commonly, knots are taken as their projections in 2D, aka knot diagrams. Over- and under-crossing information is recorded in the diagram. The key question is then:



Are they equivalent?

# Knot equivalence

Commonly, knots are taken as their projections in 2D, aka knot diagrams. Over- and under-crossing information is recorded in the diagram. The key question is then:



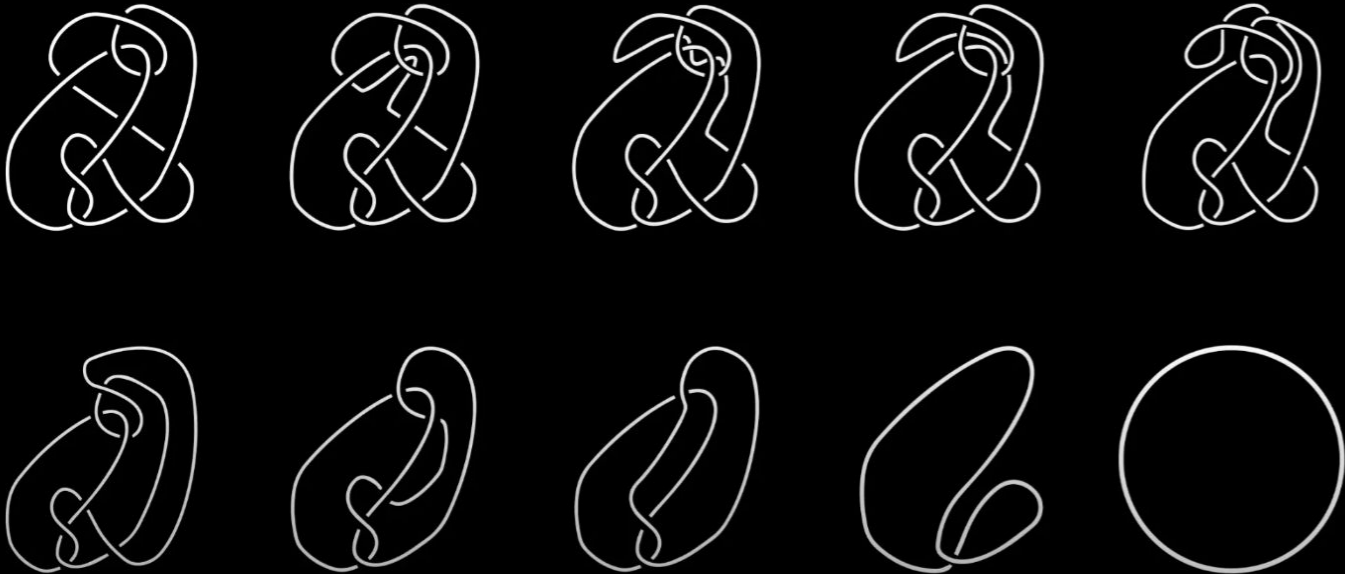
# Knots equivalence (Reidemeister theorem)

Two knot diagrams are equivalent iff they are related by three classes of local deformations.

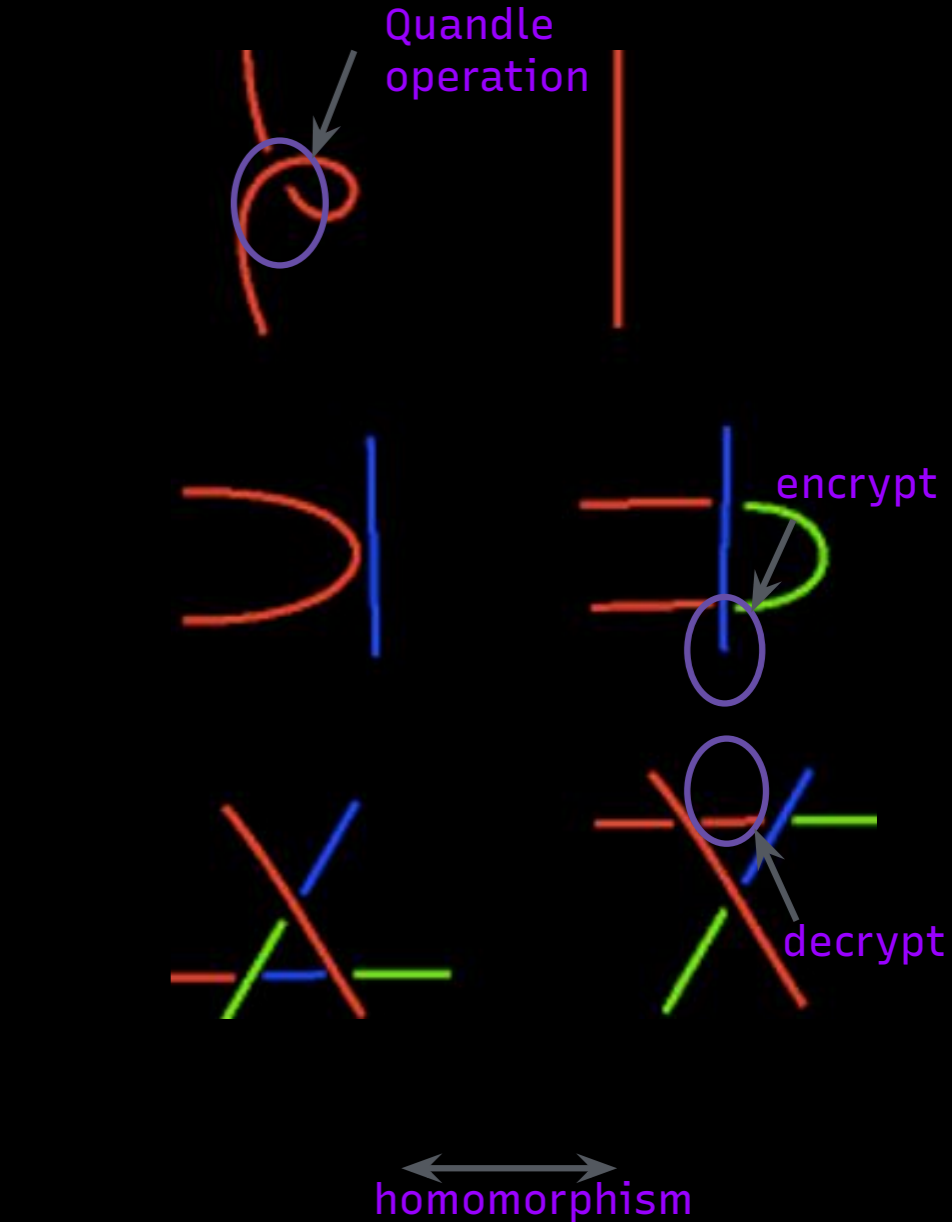
Move 1: "Twist"    Move 2: "Poke"    Move 3: "Slide"



Untying the unknot via the 1,2,3 moves:



The **quandle axioms** are the algebraic manifestation of the Reidemeister moves.



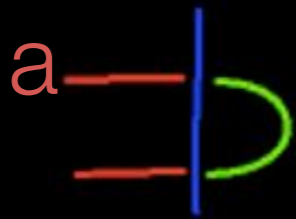
# Quandles

The operator  $\triangleright$  together with the colors define an algebraic structure known as a quandle.



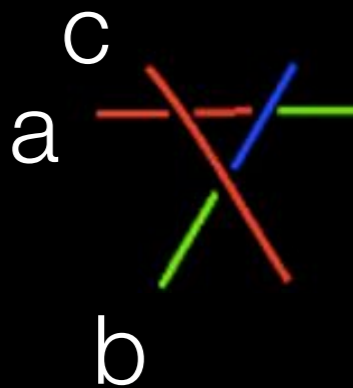
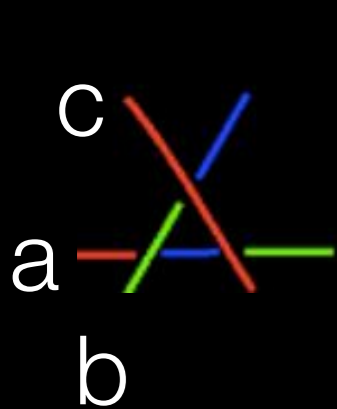
$$a \triangleright a = a$$

idempotence



$$(a \triangleright b) \triangleleft b = a$$

bijectivity

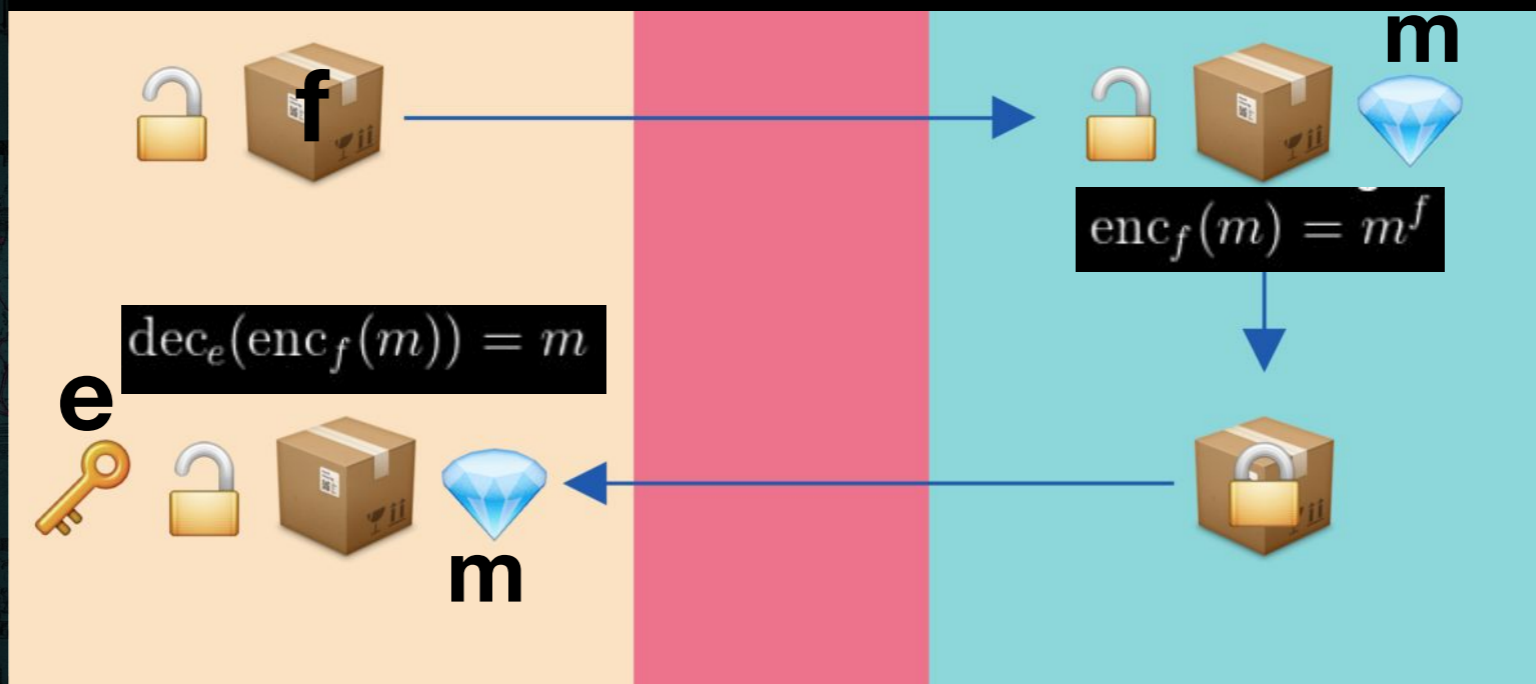


$$(a \triangleright b) \triangleright c = (a \triangleright c) \triangleright (b \triangleright c)$$

distributivity

# RSA cryptosystem

Is the most widely used asymmetric public-key cryptography. Asymmetric - as it uses different keys for encoding and decoding.



RSA trapdoor (information without which decryption is hard)

$$ef \equiv 1 \pmod{\varphi(n)}, \quad (m^f)^e \equiv m \pmod{n}, \quad m \in \mathbb{Z}/n\mathbb{Z}$$

when  $n$  is large, the totient function is hard to compute unless its prime decomposition is known, rendering the private key ( $e$ ) a trapdoor.

$$\text{enc}_f(m) = m^f \pmod{n},$$

encrypt using pub key  $f$

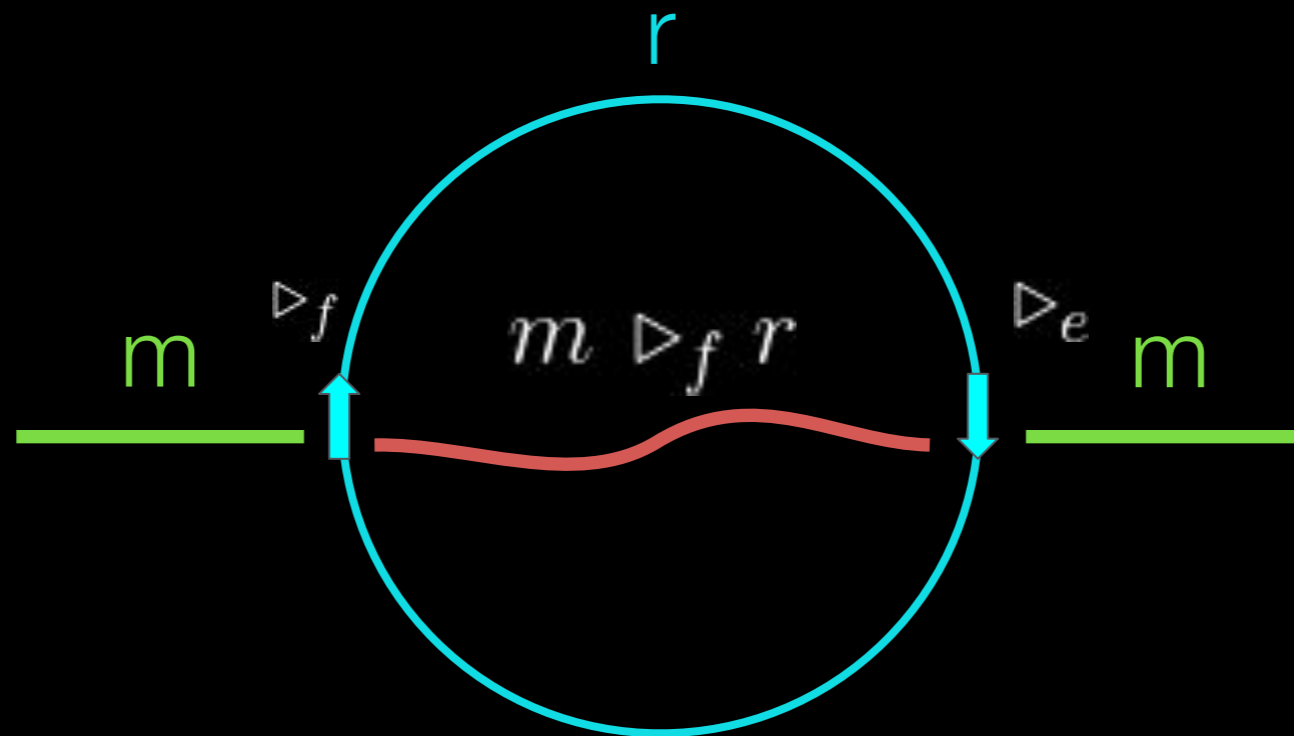
$$\text{dec}_e(m) = m^e \pmod{n},$$

decrypt using private key  $e$

$$\text{dec}_e(\text{enc}_f(m)) = m \pmod{n}$$

# RSA as a quandle

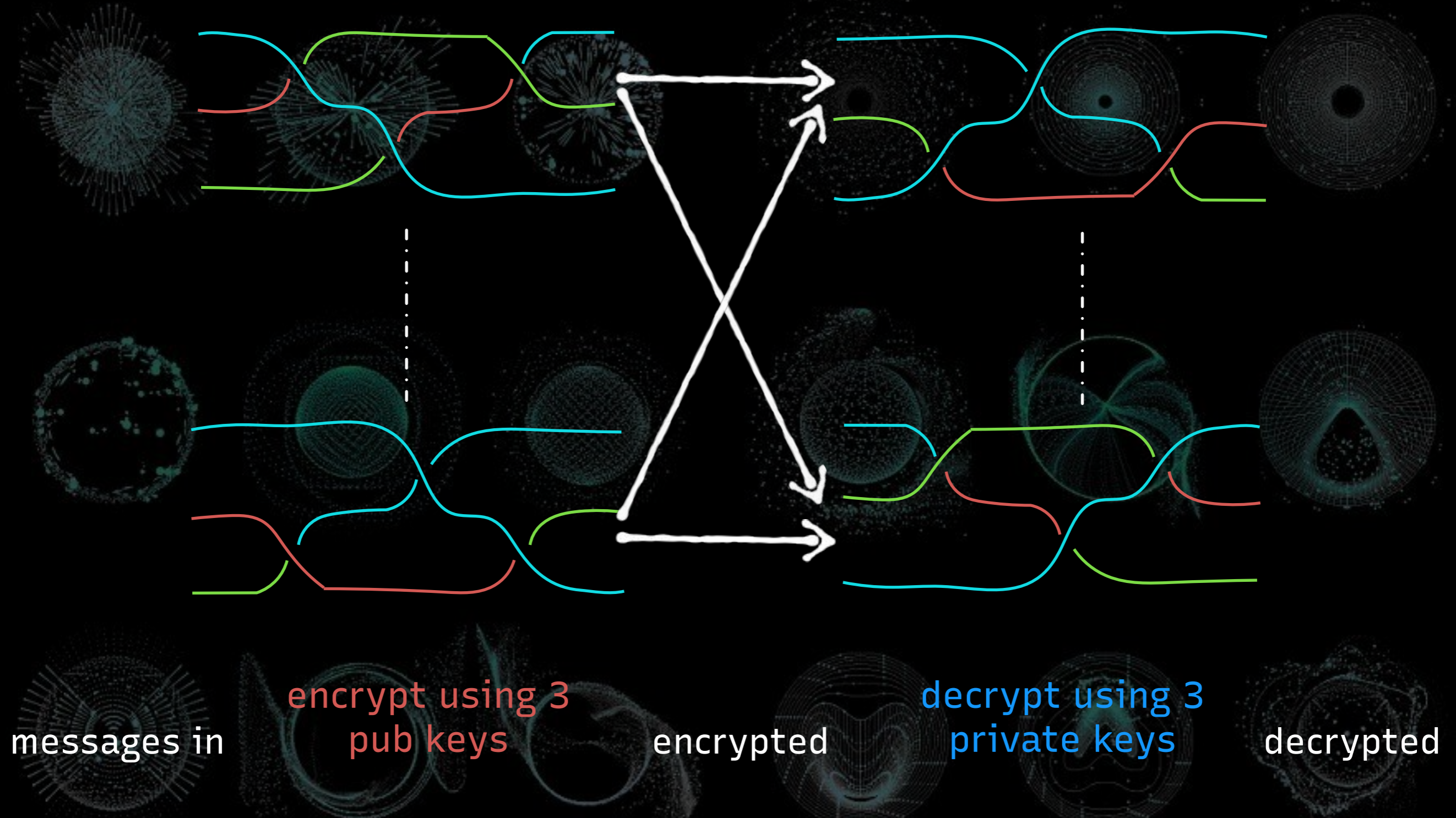
We have defined,  $a \triangleright_e b := a^e b^{1-e} \pmod n$  over the ring  $\mathbb{Z}/n\mathbb{Z}$  (also over the rationals), and have proven it is a quandle. The quandle operator and its inverse differ only by the key role, private or public.



$$\triangleleft_e = \triangleright_f : (m \triangleright_f r) \triangleright_e r = m \pmod n$$

the RSA cryptosystem is a special case of the quandle's bijectivity with the auxiliary message  $r=1$ .

# Quandle cryptographic protocols



dotted edges show equivalences by R3. There are two topologically/algebraically equivalent ways of decrypting the output of any of the two encryption braids.

# Hardness: unbraiding/unknotting + RSA



public key

```
<root>
<!--Braid word representation-->
<braid>
  X      X
  |      |
  X      X
</braid>

<!--Generators/crossing keys, signatures: 1 = public, -1 = private -->
<keys>
  privatel.pem,
  privatel.pem,
  publicl.pem
</keys>

<!--Input labels-->
<inputs>
  1 2 3
</inputs>

<!--Output labels-->
<outputs>
  1 2 3
</outputs>

<!--Encryption braiding scheme execution-->
<execute>
  <in:1>tmp/file1</in:1>
  <in:2>tmp/file2</in:2>
  <in:3>tmp/file3</in:3>
  <out:1>tmp/out1</out:1>
  <out:2>tmp/out2</out:2>
  <out:3>tmp/out3</out:3>
</execute>
</root>
```



private key

```
<root>
<!--Braid word representation-->
<braid>
  X      X
  |      |
  X      X
</braid>

<!--Generators/crossing keys, signatures: 1 = public, -1 = private -->
<keys>
  publicl.pem,
  publicl.pem,
  privatel.pem
</keys>

<!--Input labels-->
<inputs>
  1 2 3
</inputs>

<!--Output labels-->
<outputs>
  1 2 3
</outputs>

<!--Decryption braiding scheme execution-->
<execute>
  <in:1>tmp/out1</in:1>
  <in:2>tmp/out2</in:2>
  <in:3>tmp/out3</in:3>
  <out:1>tmp/decrypted1</out:1>
  <out:2>tmp/decrypted2</out:2>
  <out:3>tmp/decrypted3</out:3>
</execute>
</root>
```



# Homomorphic encryption

$$\text{Enc}(f(m_0, m_1)) = f(\text{Enc}(m_0), \text{Enc}(m_1))$$

Quandle cryptography is homomorphic in the sense:

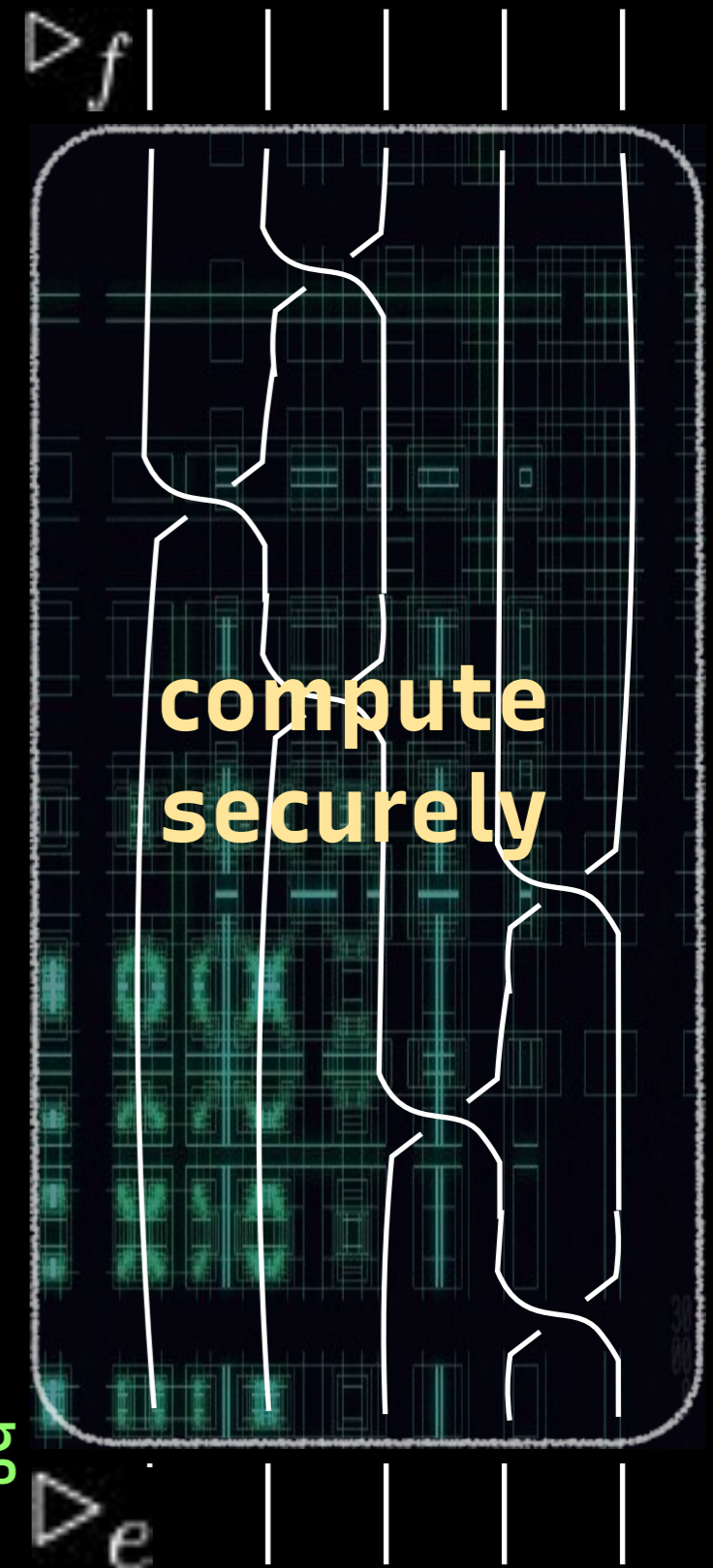
Multiplication:  $(a_0 a_1) \triangleright (b_0 b_1) = (a_0 \triangleright b_0)(a_1 \triangleright b_1)$

Distributivity:  $(a \triangleright_e b) \triangleright_g c = (a \triangleright_g c) \triangleright_e (b \triangleright_g c)$

Key-multiplicity:  $(\cdot \triangleright_{e_1} c) \triangleright_{e_2} c = \cdot \triangleright_{e_1 e_2} c$

Additivity:  $(pa) \triangleright_e r + (qb) \triangleright_e r = (pa + qb) \triangleright_e r$

the last one is perhaps the most surprising. Here,  $p$  and  $q$  are the zero-divisors in the ring (the prime factors of the modulo)



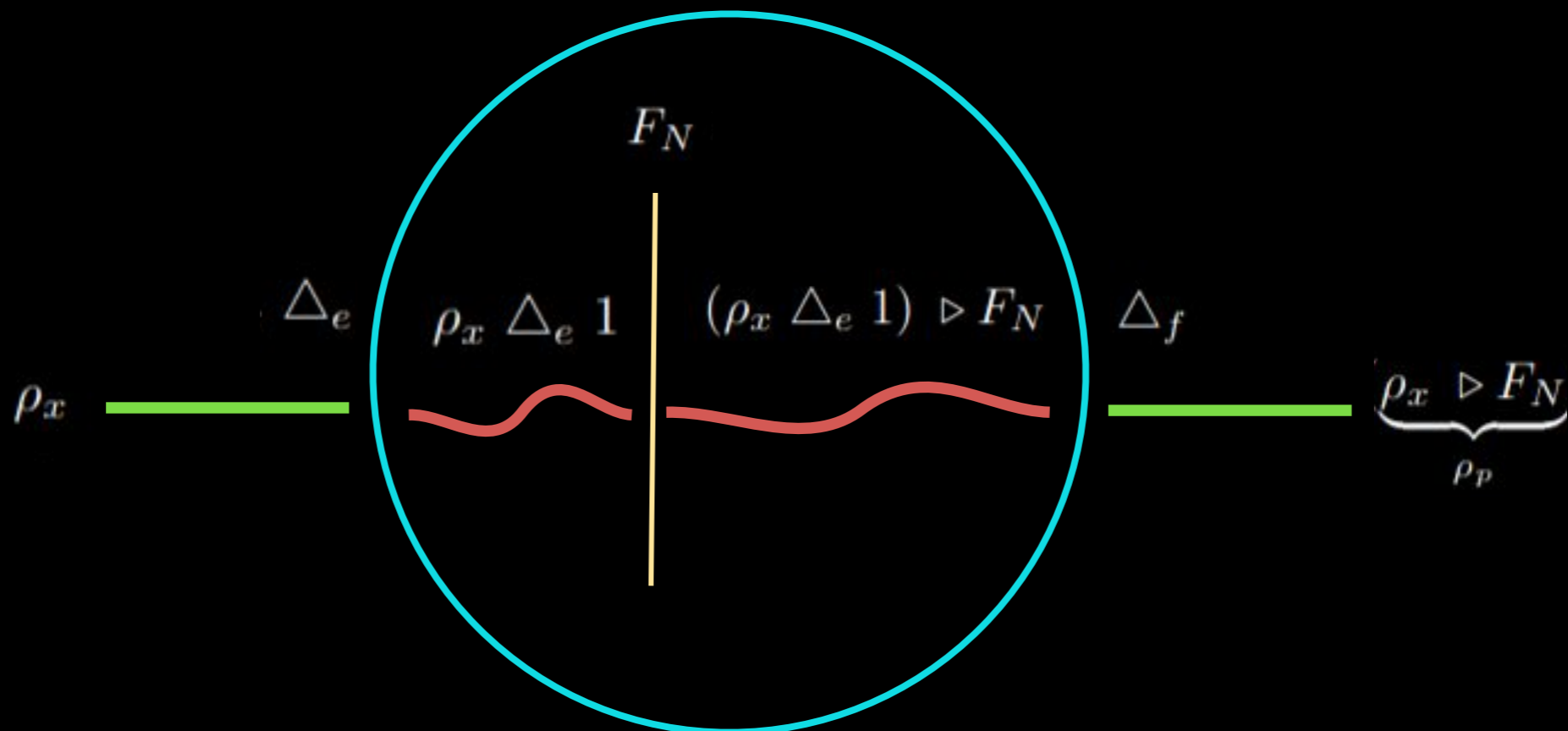
# Quick wrap up

rôle	quandle	carrier set	operation
quantum data-path	conjugation	density operators on $\mathbb{C}^N$	$X \triangleright_c U := UXU^\dagger$
classical key-path	log-linear ( "RSA" )	$\mathbb{Z}_n^*$ ( $n = pq$ )	$a \Delta_e b := a^e b^{1-e} \text{ mod } n$

- We have encoded NP-complete problems in density operator and used conjugation quandle (Fourier transform) to compute solution, and
- We have used a log-linear quandle over integer rings to generalize the RSA cryptosystem.
- These are two quandles defined over quite different algebras.
- We would like to combine them by a single quandle homomorphism.

# Phase-space NP confidential computation

rôle	quandle	carrier set	operation
quantum data-path	conjugation	density operators on $\mathbb{C}^N$	$X \triangleright_c U := UXU^\dagger$
classical key-path	log-linear ( "RSA" )	$\mathbb{Z}_n^*$ ( $n = pq$ )	$a \Delta_e b := a^e b^{1-e} \text{ mod } n$



$$(\rho_x \Delta_e 1) \triangleright F_N = \underbrace{(\rho_x \triangleright F_N)}_{\rho_p} \Delta_e 1$$

"Encrypt (RSA) then evaluate (Fourier)" = "Evaluate then encrypt"

# The homomorphism

We shall find a mapping preserving the quandle structure that obeys

$$(\rho_x \triangle_e 1) \triangleright F_N = \underbrace{(\rho_x \triangleright F_N)}_{\rho_p} \triangle_e 1$$

$$\Phi : \mathbb{Z}_N^2 \longrightarrow \mathcal{P}_N / \{\text{global phase}\}, \quad t = \begin{pmatrix} t_Z \\ t_X \end{pmatrix} \longmapsto P(t) = Z^{t_Z} X^{t_X} \quad (1)$$

is compatible both with the "RSA-scaling"  $t \mapsto et$  and with conjugation by every Clifford gate  $G$ :

$$\Phi(et) = \Phi(t)^e, \quad \Phi(M_G t) = G \Phi(t) G^\dagger. \quad (2)$$

$\mathcal{P}_N = \langle X, Z \rangle$  is the  $N$ -dimensional Heisenberg-Weyl ("Pauli") group:

# The homomorphism

Every Clifford gate (e.g., DFT flips bases  $X$  and  $Z$ ) has a Pauli operator:

$\mathcal{P}_N = \langle X, Z \rangle$  is the  $N$ -dimensional Heisenberg-Weyl ("Pauli") group:

- **Weyl operators** on an  $N$ -dimensional qudit

$$Z|k\rangle = \omega^k |k\rangle, \quad X|k\rangle = |k+1\rangle, \quad \omega = e^{2\pi i/N}. \quad (3)$$

- **Commutation rule**  $XZ = \omega^{-1}ZX \rightarrow Z^a X^b = \omega^{ab} X^b Z^a$ .
- **Pauli (Heisenberg–Weyl) operator** labelled by  $t = (t_Z, t_X) \in \mathbb{Z}_N^2$ :

$$P(t) := Z^{t_Z} X^{t_X}. \quad (4)$$

# The fundamental identity

Under symplectic action (unitary preserving transformation) the Pauli operator:

The  $2 \times 2$  matrix  $M_G \in \text{SL}_2(\mathbb{Z}_N)$  is the usual symplectic action of  $G$  on Pauli exponents (e.g. for a Hadamard  $M_H = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ ).

$$GZG^\dagger = Z^\alpha X^\beta, \quad GXG^\dagger = Z^\gamma X^\delta, \quad (\alpha, \beta, \gamma, \delta) \in \mathbb{Z}_N$$

modulo global phase factors. Hence,

$$\begin{aligned} P(M_G t) &= P \left( \underbrace{\begin{bmatrix} \alpha & \gamma \\ \beta & \delta \end{bmatrix}}_{M_G} \begin{bmatrix} t_z \\ t_x \end{bmatrix} \right) = Z^{\alpha t_z + \gamma t_x} X^{\beta t_z + \delta t_x} \\ &= (Z^\alpha X^\beta)^{t_z} (Z^\gamma X^\delta)^{t_x} = GZ^{t_z} X^{t_x} G^\dagger = GP(t)G^\dagger \quad (5) \end{aligned}$$

# The fundamental identity

We thus get the homomorphic identity:

Fundamentally,

$$\begin{aligned} \boxed{(P(t) \Delta_e 1) \triangleright G} &= GP(t)^e G^\dagger \\ &= GP(et)G^\dagger = P(M_G \cdot et) = (Z^\alpha X^\beta)^{etz} (Z^\gamma X^\delta)^{etx} \\ &= \left[ (Z^\alpha X^\beta)^{tz} (Z^\gamma X^\delta)^{tx} \right]^e = P(M_G t)^e = [GP(t)G^\dagger]^e \\ &= \boxed{(P(t) \triangleright G) \Delta_e 1} \quad (6) \end{aligned}$$

- Indeed, within the Heisenberg-Weyl group, "encrypt then evaluate" = "evaluate then encrypt"
- The encryption (log-linear quandle RSA) is at the level of the Lie algebra.
- The computation (conjugate Clifford gates) is at the level of the corresponding Lie group.

# Homomorphic QFT via Clifford quandle

The homomorphic NP phase-space computation (QFT) schematics:

**Public key:**  $(n, e, g, N)$     **Secret key:**  $(p, q, f)$ .

generator  $g \in (\mathbb{Z}/n\mathbb{Z})^*$  of exact order  $N = 2^n$

Client

Pauli mask  $\mathbf{t} = (t_Z, t_X) \in \mathbb{Z}_N^2$

encrypt NP instance

$$P(\mathbf{t}) = Z^{t_Z} X^{t_X}$$

$$\tilde{\rho} = P(\mathbf{t}) \rho_x P(\mathbf{t})^\dagger$$

encrypt mask

$$c = g^{t_Z \cdot e} \pmod{n}$$

decrypt mask

$$t_Z = c^f \pmod{n}$$

decrypt NP solution

$$\rho_p = P(-\mathbf{t}) \tilde{\rho} P(-\mathbf{t})^\dagger$$

$(\tilde{\rho}, c)$

encrypted instance

Server

solve homomorphically

$$\tilde{\rho} \leftarrow F_N \tilde{\rho} F_N^\dagger$$

update mask

$$c \leftarrow c^{\alpha_F} \pmod{n}$$

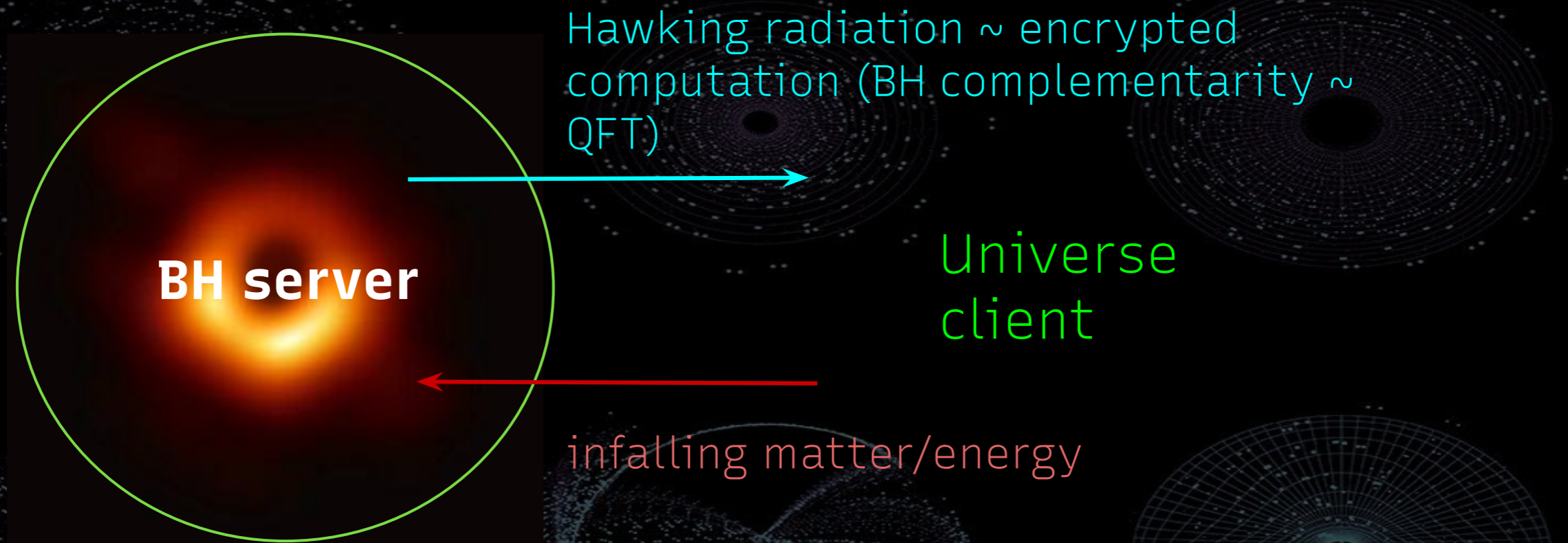
$(\tilde{\rho}, c)$

encrypted solution

# Summary

- NP and #P problems are solvable via DFT/QFT. Exponential hypothesis is respected.
- Quandles provide a framework for both encryption and quantum computation.
- The Clifford quandle (defined on Heisenberg-Weyl "Pauli" group) captures both DFT/QFT (qudit) computation and homomorphic RSA encryption, hence suitable for phase-space homomorphic computation.
- Issue 1: The scheme is not quantum universal. Some quantum gates are not in the Clifford group. Interestingly, the standard n-qubit QFT is not Clifford.
- Issue 2: The scheme works with basic RSA but quandle encryption is far more diverse. How would a PQC extension look like?

# Black holes $\sim$ homomorphic servers



Without the trapdoor information (e.g., private key) Hawking radiation is highly scrambled, exhibiting short-range correlations which are hard to decode.

```

                                     AA
                                   AAAAA
                                AAAAA
                               AAAAA
                              AAAAA
                             AAAAA
                            AAAAA
                           AAAAA
                          AAAAA
                         AAAAA
                        AAAAA
                       AAAAA
                      AAAAA
                     AAAAA
                    AAAAA
                   AAAAA
                  AAAAA
                 AAAAA
                AAAAA
               AAAAA
              AAAAA
             AAAAA
            AAAAA
           AAAAA
          AAAAA
         AAAAA
        AAAAA
       AAAAA
      AAAAA
     AAAAA
    AAAAA
   AAAAA
  AAAAA
 AAAAA
AAAAA

```

options:

- h, --help show this help message and exit
- fps FPS Frames per second (default: 10)
- width WIDTH Output width in pixels (maintains aspect ratio)

```
(venv) crypsis@crypsisZ:~/PythonDev/TerminalCapture$ python3.11 webm2gif.py trefoil.webm trefoil.gif -fps 30
```

```
usage: webm2gif.py [-h] [--fps FPS] [--width WIDTH] input output
```

```
webm2gif.py: error: unrecognized arguments: -fps 30
```

```
(venv) crypsis@crypsisZ:~/PythonDev/TerminalCapture$ python3.11 webm2gif.py trefoil.webm trefoil.gif --fps 30
```

```
Generating color palette...
```

```
Converting to GIF...
```

```
Successfully converted trefoil.webm to trefoil.gif
```

```
(venv) crypsis@crypsisZ:~/PythonDev/TerminalCapture$
```