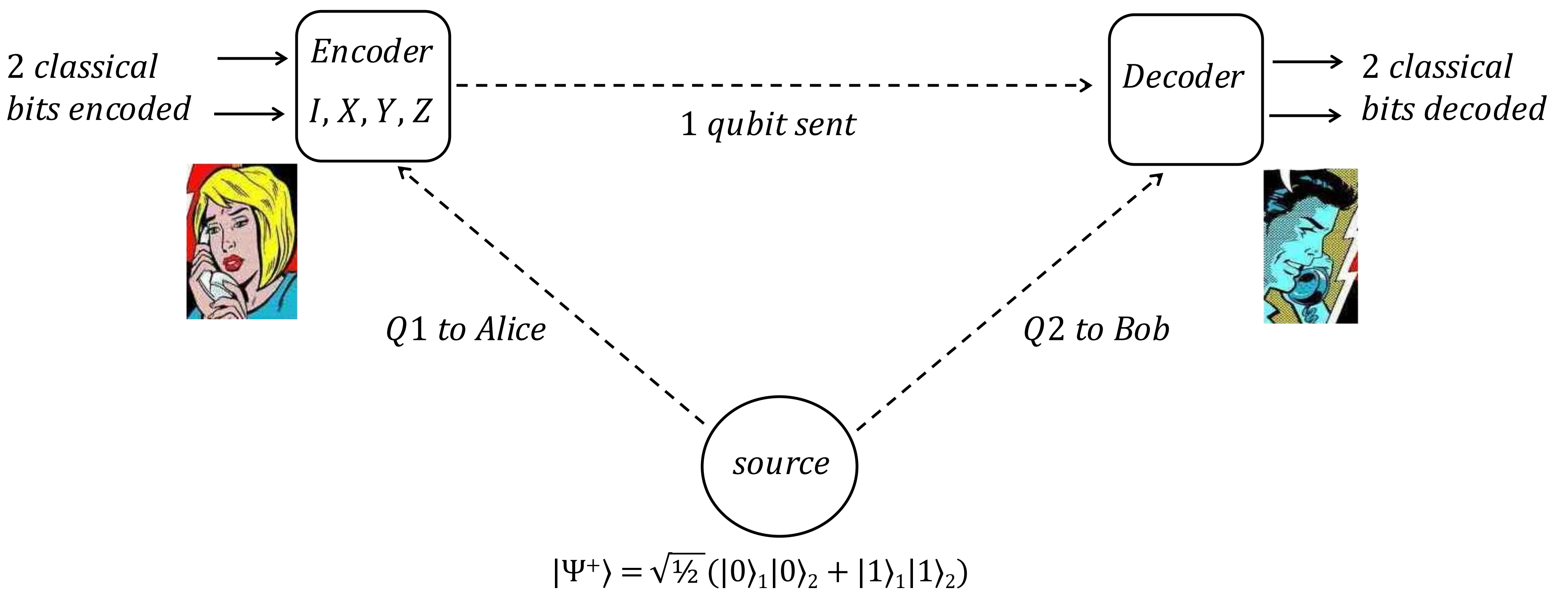
07. QIT, Part II.

1. Quantum Dense Coding

- *Goal*: To use one qubit to transmit two classical bits.
- *But*: One qubit (supposedly) only contains one classical bit's worth of information!
- *So*: How can we send 2 classical bits using just one qubit?
- *Answer*: Use entangled states!

<u>Set-Up</u>:

- Prepare two qubits $Q1$, $Q2$ in an entangled state $|\Psi^+\rangle = \sqrt{½}\,(|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2)$
- Alice gets $Q1$, Bob gets $Q2$.
- Alice manipulates her $Q1$ so that it steers Bob's $Q2$ into a state from which he can read off the 2 classical bits Alice desires to send. All he needs to do this is the post-manipulated $Q1$ that Alice sends to him.



Protocol

1. Alice has a pair of classical bits: either 00, 01, 10, or 11.
   She first encodes the pair in $Q1$ by acting on $Q1$ with one of $\{I, X, Y, Z\}$ according to:

pair	*transform*	*new state*
00	$(I_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 + \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^+\rangle$
01	$(X_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^+\rangle$
10	$(Y_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(-\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^-\rangle$
11	$(Z_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 - \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^-\rangle$

- *Let $Q1$ and $Q2$ be electrons in Hardness states.*
- *Let $\lvert 0\rangle$ be $\lvert soft\rangle$ and $\lvert 1\rangle$ be $\lvert hard\rangle$.*

2. Alice now sends $Q1$ to Bob.

3. After reception of $Q1$, Bob first applies a C_{NOT} transformation to both $Q1$ and $Q2$:

pair	*transform*	*new state*	*Apply C_{NOT}*
00	$(I_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 + \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1 + \lvert 1\rangle_1)\lvert 0\rangle_2$
01	$(X_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^+\rangle$	$\sqrt{½}\,(\lvert 1\rangle_1 + \lvert 0\rangle_1)\lvert 1\rangle_2$
10	$(Y_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(-\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^-\rangle$	$\sqrt{½}\,(-\lvert 1\rangle_1 + \lvert 0\rangle_1)\lvert 1\rangle_2$
11	$(Z_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 - \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^-\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1 - \lvert 1\rangle_1)\lvert 0\rangle_2$

According to the Eigenvalue-Eigenvector Rule, $Q1$ still has no definite value, but $Q2$ now does!

Protocol

4. Bob now applies a Hadamard transformation to $Q1$:

pair	*transform*	*new state*	*Apply C_{NOT}*	*Apply $\mathfrak{H}_1$*
00	$(I_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 + \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1 + \lvert 1\rangle_1)\lvert 0\rangle_2$	$\lvert 0\rangle_1\lvert 0\rangle_2$
01	$(X_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^+\rangle$	$\sqrt{½}\,(\lvert 1\rangle_1 + \lvert 0\rangle_1)\lvert 1\rangle_2$	$\lvert 0\rangle_1\lvert 1\rangle_2$
10	$(Y_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(-\lvert 1\rangle_1\lvert 0\rangle_2 + \lvert 0\rangle_1\lvert 1\rangle_2) = \lvert\Phi^-\rangle$	$\sqrt{½}\,(-\lvert 1\rangle_1 + \lvert 0\rangle_1)\lvert 1\rangle_2$	$\lvert 1\rangle_1\lvert 1\rangle_2$
11	$(Z_1 \otimes I_2)\lvert\Psi^+\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1\lvert 0\rangle_2 - \lvert 1\rangle_1\lvert 1\rangle_2) = \lvert\Psi^-\rangle$	$\sqrt{½}\,(\lvert 0\rangle_1 - \lvert 1\rangle_1)\lvert 0\rangle_2$	$\lvert 1\rangle_1\lvert 0\rangle_2$

According to the EE Rule, $Q1$ and $Q2$ now both have definite values.

5. Bob now measures $Q1$ and $Q2$ to determine the number Alice sent!

   (a) $(Q1 = 0, Q2 = 0) \Rightarrow 00$

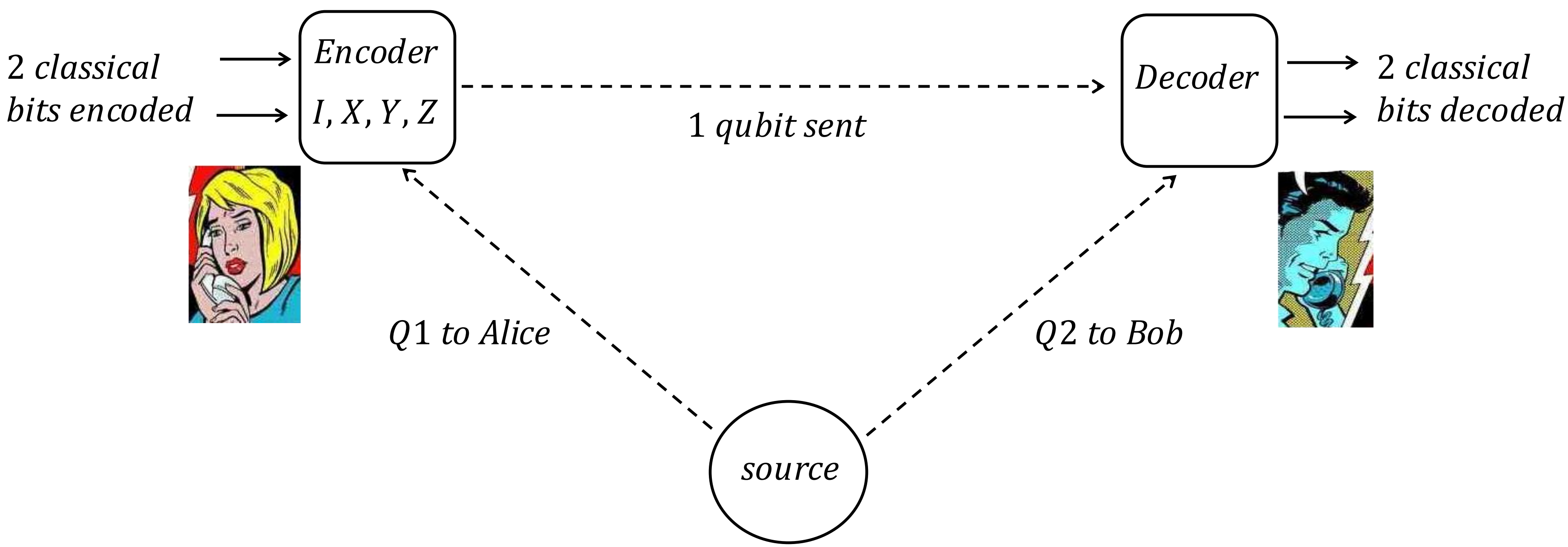
   (b) $(Q1 = 0, Q2 = 1) \Rightarrow 01$

   (c) $(Q1 = 1, Q2 = 1) \Rightarrow 10$

   (d) $(Q1 = 1, Q2 = 0) \Rightarrow 11$

Question: How are the 2 classical bits transferred from Alice to Bob?

- *Not* transferred *via* the single qubit.
- Transferred by the *correlations* present in the 2-qubit entangled state $|\Psi^+\rangle$.
- In order to convey information between Alice and Bob, it need *not* be physically transported from Alice to Bob across the intervening spatial distance.
- The *only* thing required to convey information is to set up a correlation between the sender's data and the receiver's data.



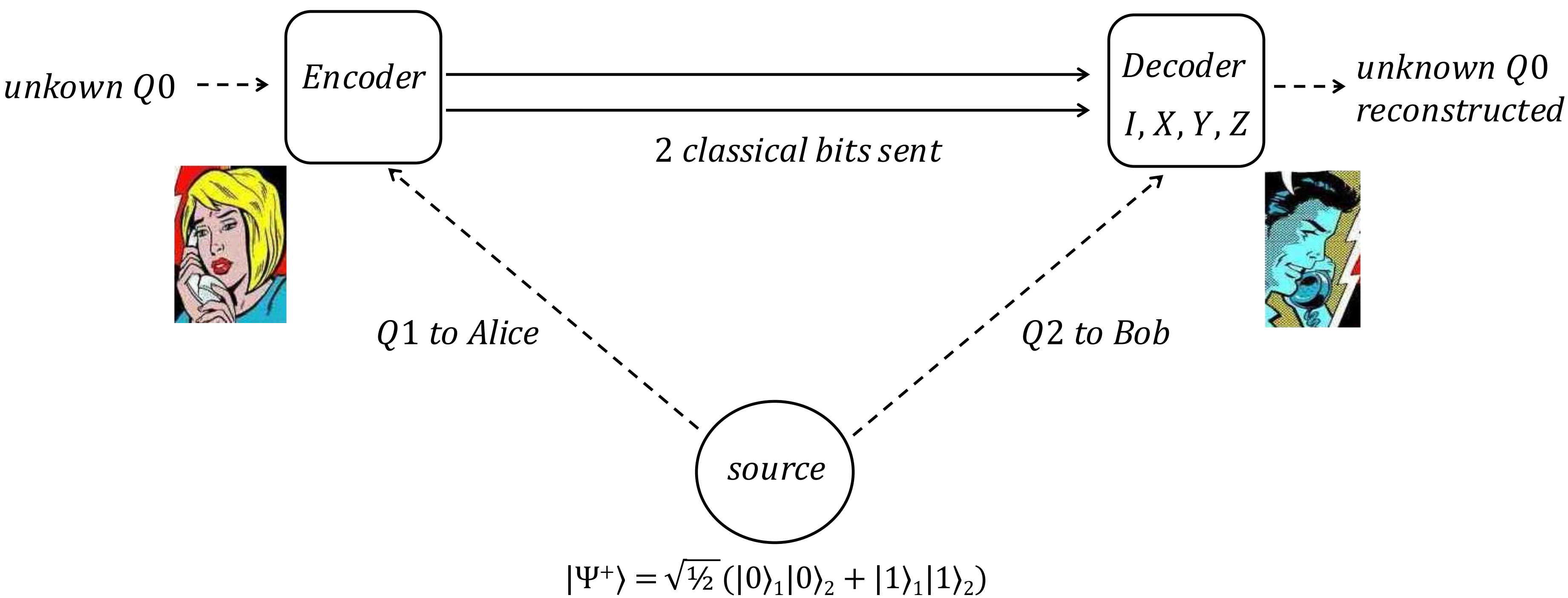
$$|\Psi^+\rangle = \sqrt{1/2}\,(|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2)$$

2. Quantum Teleportation

- *Goal*: To transmit an unknown quantum state using classical bits and to reconstruct the exact quantum state at the receiver.
- *But*: How can this avoid the No-Cloning Theorem?
- *Answer*: Use entangled states!

Set-Up:

- Alice has an unknown $Q0$, $|Q\rangle_0 = a|0\rangle_0 + b|1\rangle_0$, and wants to send it to Bob.
- $Q1$ and $Q2$ are prepared in an entangled state $|\Psi^+\rangle = \sqrt{½}\,(|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2)$. Alice gets $Q1$, Bob gets $Q2$.
- Alice manipulates $Q0$ and $Q1$ so that they steer Bob's $Q2$ into a form from which he can reconstruct the unknown state of $Q0$. All Bob needs to do this are 2 classical bits sent by Alice.



Protocol

1. Alice starts with a 3-qubit system ($Q0$, $Q1$, $Q2$) in the state:

$$|Q\rangle_0|\Psi^+\rangle = \sqrt{½}\,(a|0\rangle_0|0\rangle_1|0\rangle_2 + a|0\rangle_0|1\rangle_1|1\rangle_2 + b|1\rangle_0|0\rangle_1|0\rangle_2 + b|1\rangle_0|1\rangle_1|1\rangle_2)$$

Alice now applies C_{NOT} on $Q0$ & $Q1$, and then a Hadamard transformation on $Q0$:

First C_{NOT} on $Q0$ & $Q1$:

$$(C_{NOT_{01}} \otimes I_2)|Q\rangle_0|\Psi^+\rangle = \sqrt{½}\,(a|0\rangle_0|0\rangle_1|0\rangle_2 + a|0\rangle_0|1\rangle_1|1\rangle_2 + b|1\rangle_0|1\rangle_1|0\rangle_2 + b|1\rangle_0|0\rangle_1|1\rangle_2)$$

Then $\mathfrak{H}$ on $Q0$:

$$(\mathfrak{H}_0 \otimes I_1 \otimes I_2)(''\ '') = ½|0\rangle_0|0\rangle_1(a|0\rangle_2 + b|1\rangle_2) + ½|0\rangle_0|1\rangle_1(a|1\rangle_2 + b|0\rangle_2)$$
$$+ ½|1\rangle_0|0\rangle_1(a|0\rangle_2 - b|1\rangle_2) + ½|1\rangle_0|1\rangle_1(a|1\rangle_2 - b|0\rangle_2)$$

EE Rule: Each of the terms represents a state in which $Q0$ and $Q1$ have definite values, but $Q2$ does not.

2. Alice now measures $Q0$ and $Q1$:

If measurement outcome is:	*...$Q2$ is now in state:*
$\vert 0\rangle_0\vert 0\rangle_1$	$a\vert 0\rangle_2 + b\vert 1\rangle_2$
$\vert 0\rangle_0\vert 1\rangle_1$	$a\vert 1\rangle_2 + b\vert 0\rangle_2$
$\vert 1\rangle_0\vert 0\rangle_1$	$a\vert 0\rangle_2 - b\vert 1\rangle_2$
$\vert 1\rangle_0\vert 1\rangle_1$	$a\vert 1\rangle_2 - b\vert 0\rangle_2$

Protocol

If measurement outcome is:	*...Q2 is now in state:*
$\vert 0\rangle_0 \vert 0\rangle_1$	$a\vert 0\rangle_2 + b\vert 1\rangle_2$
$\vert 0\rangle_0 \vert 1\rangle_1$	$a\vert 1\rangle_2 + b\vert 0\rangle_2$
$\vert 1\rangle_0 \vert 0\rangle_1$	$a\vert 0\rangle_2 - b\vert 1\rangle_2$
$\vert 1\rangle_0 \vert 1\rangle_1$	$a\vert 1\rangle_2 - b\vert 0\rangle_2$

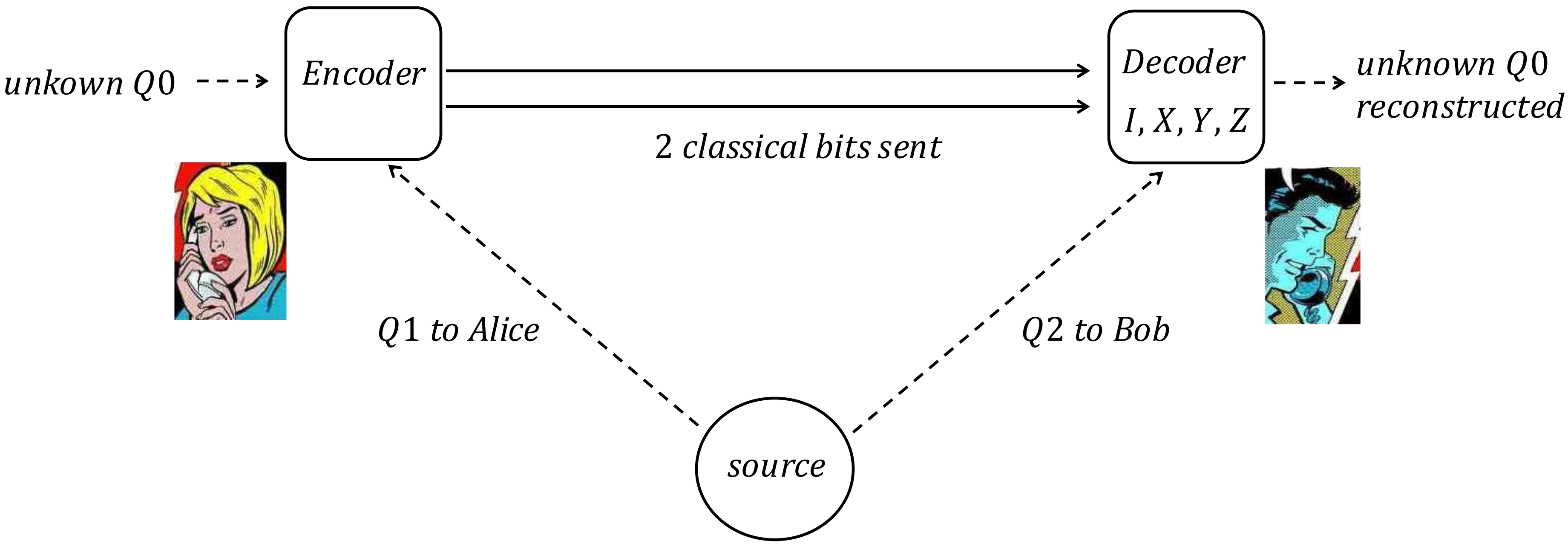
3. Alice sends the result of her measurement to Bob in the form of 2 classical bits: 00, 01, 10, or 11.

4. Depending on what he receives, Bob performs one of $\{I, X, Y, Z\}$ on $Q2$.
   This allows him to turn it into (reconstruct) the unknown $Q0$.

If bits received are	*...then $Q2$ is now in state*	*...so to reconstruct $Q0$, use*
00	$a\vert 0\rangle_2 + b\vert 1\rangle_2$	I_2
01	$a\vert 1\rangle_2 + b\vert 0\rangle_2$	X_2
10	$a\vert 0\rangle_2 - b\vert 1\rangle_2$	Z_2
11	$a\vert 1\rangle_2 - b\vert 0\rangle_2$	Y_2

Question 1: Does Bob violate the *No-Cloning Theorem*? Doesn't he construct a copy of the unknown $Q0$?

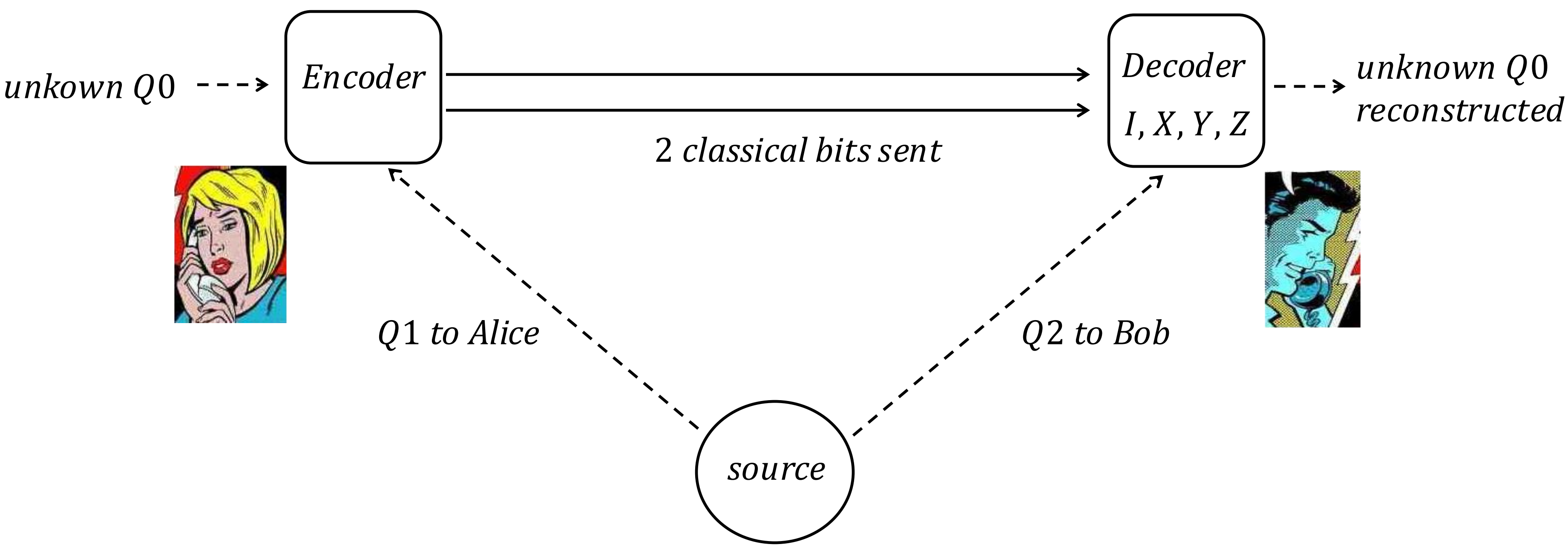
- *No violation occurs.*
- Bob *does* construct a copy: $Q2$ has become an exact duplicate of $Q0$.
- *But*: After Alice is through transforming $Q0$ and $Q1$, the original $Q0$ has now collapsed to either $|0\rangle_0$ or $|1\rangle_0$!
  - *Alice destroys $Q0$ in the process of conveying the information contained in it to Bob!*



$$|\Psi^{+}\rangle = \sqrt{½}\,(|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2)$$

Question 2: How does Bob reconstruct the unknown $Q0$ (that encodes an arbitrarily large amount of information) from just 2 classical bits?

- Information to reconstruct $Q0$ is transferred by the correlations present in the entangled state $|\Psi^{+}\rangle$, *in addition* to the 2 classical bits.
- The 2 classical bits are used simply to determine the appropriate transformation on $Q2$, *after* it has been "steered" into the appropriate state by Alice.



$$|\Psi^{+}\rangle = \sqrt{½}\,(|0\rangle_1|0\rangle_2 + |1\rangle_1|1\rangle_2)$$

3. Quantum Computation.

- *General Goal*: To use the inaccessible arbitrarily large amount of information encoded in qubits to perform computations in "quantum parallel" (*i.e.*, in record time!).

- *Initial (modest) Goal*: To compute all possible values of a function f in a single computation.

- *First Question*: Can classical computations be done using qubits instead of classical bits?
  - *Can transformations on qubits be defined that reproduce the transformations on bits that are needed to implement a classical computer.*

Classical Computation Using Bits

To implement a classical computer, it suffices to have an *AND* transformation and a *NOT* transformation on classical bits defined by the following:

$$0\ AND\ 0 = 0 \qquad NOT\ 0 = 1$$
$$0\ AND\ 1 = 0 \qquad NOT\ 1 = 0$$
$$1\ AND\ 0 = 0$$
$$1\ AND\ 1 = 1$$

- *AND takes two input bits and produces one output bit.*
- *NOT takes one input bit and produces one output bit.*

- *Initial problem*: Transformations on qubits are *reversible*: the number of input qubits *always* must equal the number of output qubits.

Why? Qubit transformations are operators on vector spaces. And an operator defined on an n-dim vector space (e.g., n-qubit space) that acts on n-dim vectors (e.g., n qubits) can only spit out n-dim vectors.

Solution: The Controlled-controlled-*NOT*, CC_{NOT}, operator.

- Changes the third target qubit if the first two qubits are $|1\rangle|1\rangle$, and leaves it unchanged otherwise.

$$CC_{NOT}|0\rangle|0\rangle|0\rangle = |0\rangle|0\rangle|0\rangle \qquad CC_{NOT}|0\rangle|1\rangle|1\rangle = |0\rangle|1\rangle|1\rangle \qquad CC_{NOT}|1\rangle|1\rangle|0\rangle = |1\rangle|1\rangle|1\rangle$$

$$CC_{NOT}|0\rangle|0\rangle|1\rangle = |0\rangle|0\rangle|1\rangle \qquad CC_{NOT}|1\rangle|0\rangle|0\rangle = |1\rangle|0\rangle|0\rangle \qquad CC_{NOT}|1\rangle|1\rangle|1\rangle = |1\rangle|1\rangle|0\rangle$$

$$CC_{NOT}|0\rangle|1\rangle|0\rangle = |0\rangle|1\rangle|0\rangle \qquad CC_{NOT}|1\rangle|0\rangle|1\rangle = |1\rangle|0\rangle|1\rangle$$

$$CC_{NOT} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad |0\rangle|0\rangle|0\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ |0\rangle|0\rangle|1\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ \cdots, |1\rangle|1\rangle|1\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

- *Claim:* CC_{NOT} implements *AND* and *NOT* on qubits.
  - To implement *AND*, act with CC_{NOT} on a 3-qubit state in which the last qubit is $|0\rangle$:

    $$CC_{NOT}|x\rangle|y\rangle|0\rangle = |x\rangle|y\rangle|x\ AND\ y\rangle$$

  - To implement *NOT*, act with CC_{NOT} on a 3-qubit state in which the first two qubits are $|1\rangle|1\rangle$:

    $$CC_{NOT}|1\rangle|1\rangle|x\rangle = |1\rangle|1\rangle|NOT\ x\rangle$$

So: Any classical computation can be done using qubits instead of bits.

- *In particular*: Any classical function that takes n input bits and produces k output bits can be implemented using arrays of primitive CC_{NOT} "gates".

How to Construct a Qubit-Based Function Calculator

- Let $|x\rangle_{(n)}$ represent n input qubits that encode the number x.
  - *Example*: $|1\rangle|1\rangle|0\rangle$ represents 6.

- Let $|0\rangle_{(k)}$ represent k $|0\rangle$ qubits (the output register).
- Let $|f(x)\rangle_{(k)}$ represent k output qubits that encode the number $f(x)$.

- Define an operator U_f that acts on $(n+k)$ qubits in the following way:

$$U_f|x\rangle_{(n)}|0\rangle_{(k)} = |x\rangle_{(n)}|f(x)\rangle_{(k)}$$

- *Now*: Feed U_f a *superposition* of all possible numbers x it can take as input.
- *Result*: A superposition of all possible values of the function in a *single* computation!

Two Steps:

1. Prepare as input a superposition of all possible numbers x that can be encoded in n bits:
   (i) Start with an n-qubit state $|0\rangle_1|0\rangle_2\cdots|0\rangle_n$
   (ii) Now apply a Hadamard transformation to each qubit:

$$(\mathfrak{H}_1\otimes\mathfrak{H}_2\otimes\cdots\otimes\mathfrak{H}_n)|0\rangle_1|0\rangle_2\cdots|0\rangle_n$$

$$=\left(\sqrt{½}\right)^n\{(|0\rangle_1+|1\rangle_1)(|0\rangle_2+|1\rangle_2)\cdots(|0\rangle_n+|1\rangle_n)\}$$

$$=\left(\sqrt{½}\right)^n\{\underbrace{|0\rangle_1|0\rangle_2\cdots|0\rangle_n}+|0\rangle_1|0\rangle_2\cdots|1\rangle_n+\cdots+\underbrace{|1\rangle_1|1\rangle_2\cdots|1\rangle_n}\}$$

The first term encodes the binary number for 0*, or* $|0\rangle_{(n)}$

Each term in between is the binary number for each number between 0 *and* 2^n-1.

The last term encodes the binary number for 2^n-1*, or* $|2^n-1\rangle_{(n)}$

$$=\left(\sqrt{½}\right)^n\sum_{x=0}^{2^n-1}|x\rangle_{(n)}$$

So the entire sum is a superposition that encodes all numbers x *such that* $0\le x<2^n$.

Two Steps:

2. Now attach a k-qubit output register $|0\rangle_{(k)}$ and apply U_f:

$$U_f\left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|0\rangle_{(k)} = \left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} U_f|x\rangle_{(n)}|0\rangle_{(k)}$$

$$= \left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|f(x)\rangle(_{k)}$$

A superposition of all possible values $f(x)$, for $0 \leq x < 2^n$, of the function f. And we've effectively calculated them all with just a single application of U_f.

The Catch: None of these values of f is accessible until we make a measurement!

The Task for Quantum Algorithm construction

Given a problem, first construct an appropriate superposition of solutions; and then manipulate the superposition so that the relevant terms aquire high probability.

Two Steps:

2. Now attach a k-qubit output register $|0\rangle_{(k)}$ and apply U_f:

$$U_f\left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|0\rangle_{(k)} = \left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} U_f|x\rangle_{(n)}|0\rangle_{(k)}$$

$$= \left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|f(x)\rangle(_{(k)}$$

A superposition of all possible values $f(x)$, for $0 \le x < 2^n$, of the function f.

Example: Let $f(x) = x^2, n = 2, k = 4$

$$\left(\sqrt{1/2}\right)^2 \sum_{x=0}^{3} |x\rangle_{(2)}|x^2\rangle_{(4)} = \tfrac{1}{2}\{(|0\rangle|0\rangle)(|0\rangle|0\rangle|0\rangle|0\rangle) + (|0\rangle|1\rangle)(|0\rangle|0\rangle|0\rangle|1\rangle) + (|1\rangle|0\rangle)(|0\rangle|1\rangle|0\rangle|0\rangle) + (|1\rangle|1\rangle)(|1\rangle|0\rangle|0\rangle|1\rangle)\}$$

- *A superposition of all possible values of x^2, for $0 \le x < 4$.*
- *Takes the form of an entangled 6-qubit state: Input 2-qubit state is in a superposition, output 4-qubit state is in a superposition, and both superpositions are entangled.*

Example: Shor's Factorization Algorithm (1994)

- Factors large integers into primes in *polynomial* time.

> - *Polynomial time*: The number of steps required to complete the algorithm for a given input is of the order n^c, $c > 1$, where n is the complexity of the input.
> - *Exponential time*: The number of steps required to complete the algorithm for a given input is of the order c^n, $c > 1$, where n is the complexity of the input.

- Current classical algorithms require exponential times.

Why is fast prime factorization important?

- Classical RSA Encryption (Rivest, Shamir & Adleman 1978).
  - *public encryption key* = product pq of two (very large) primes.
  - *private decryption key* = p, q separately
  - *Thus*: Factorizing pq (in your lifetime) would let you break RSA encryption (standard encryption for web transactions).

Two essential facts underlie Shor's algorithm:

(i) Factorizing a large integer is equivalent to determining the period r of an associated periodic function $f(x+r) = f(x)$.

(ii) A *discrete Fourier transform* maps a function $g(x)$ of period r on the domain $(0, 2^n - 1)$ to a function $G(c)$ which has approximately non-zero values only at multiples of $2^n/r$.

Protocol

- By Fact (i), to factorize a given large integer, suppose we've determined that we need to find the period r of an appropriate periodic function $f(x)$.

Step 1

- Construct a superposition of all possible solutions of $f(x)$ for $0 \leq x < 2^n$.

$$U_f\left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|0\rangle_{(k)} = \left(\sqrt{1/2}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)}|f(x)\rangle_{(k)}$$

Our Good Friend the qubit-based function calculator!

2^n terms!

superposition of 2^n terms

superposition of 2^n terms

input and output are entangled!

Step 2

- Measure $f(x)$; *i.e.*, compute *one* value of it, say $f(x_0)$.

$$\left(\sqrt{\tfrac{1}{2}}\right)^n \sum_{x=0}^{2^n-1} |x\rangle_{(n)} |f(x)\rangle_{(k)} \xrightarrow[\textit{collapse}]{} C \sum_{x=0}^{2^n-1} g(x)|x\rangle_{(n)} |f(x_0)\rangle_{(k)}$$

superposition of 2^n terms   *superposition of 2^n terms*   *superposition of $2^n/r$ terms*   *single term*

where $g(x) = 1$ for $x = x_0 + kr$, and zero otherwise (for k an integer).

- The output register has collapsed to a single term $|f(x_0)\rangle_{(k)}$.
- The input register $|x\rangle_{(n)}$ is still in a superposition of all those values of x for which $f(x) = f(x_0)$.
  - *Initially there were 2^n input terms; now there are $2^n/r$.*
- <u>*Also*</u>: $g(x)$ has the same period r as $f(x)$, since $g(x) = g(x_0 + kr)$.

<u>So</u>: To find the period of $f(x)$, we now need to find the period of $g(x)$.

Step 3

- Act on the input register with a *quantum Fourier transformation*:

$$C\sum_{x=0}^{2^n-1} g(x)|x\rangle_{(n)}|f(x_0)\rangle_{(k)} \xrightarrow[\textit{quantum FT}]{} C'\sum_{c=0}^{2^n-1} G(c)|c\rangle_{(n)}|f(x_0)\rangle_{(k)}$$

superposition of $2^n/r$ terms

Still a superposition of $2^n/r$ terms, but now $c = j2^n/r$ terms are "favorably" weighted

where $G(c)$ is the *discrete Fourier transform* of $g(x)$.

- By Fact (ii), $G(c)$ is approximately non-zero only for $c = j2^n/r$, for integer j.

- <u>*Which means*</u>: The input superposition has now been "favorably" weighted to produce values of $c = j2^n/r$ when measured.

- <u>*Which means*</u>: If we measure the input register, we will most likely get a value for $j2^n/r$. From this value, we can extract a value for r.

Two Interpretive Issues

(1) *How are quantum computers different from classical computers?*

<u>Claim</u>: Apart from *hardware* differences (quantum 2-state systems *vs*. classical 2-state systems), the essential difference between a quantum computer and a classical computer is that the former are ideally much more *efficient* than the latter.

- A quantum computer can compute anything that a classical computer can.
  - *<u>Recall</u>: Any computation implemented using bits can be implemented using qubits.*

- A classical computer can compute anything that a quantum computer can.
  - *Any computation implemented using qubits can be implemented using bits and a probabilistic algorithm.*
  - *<u>Intuitively</u>: There are probabilistic classical 2-state systems that can simulate the output of quantum 2-state systems, (although perhaps not as efficiently).*

(2) *Is quantum information different from classical information?*

Claim: No *fundamental* difference between classical and quantum information: just a difference in types of *sources*.

Information = What is produced by an information source that is required to be reproducible at the receiver if the transmission is to be counted a success.

Two Types of Information Source

I. Classical information source

- *Abstractly*: Produces letters from a set $\{x_1, x_2, ..., x_n\}$ with probabilities $p_i = p(x_i)$.
- Messages = sequences of letters. *Ex*: $x_7x_3x_4$...
- *Concretely*: Produces *physical systems* (e.g., on-off switches) in *classical states* $\{x_1, x_2, ..., x_n\}$.
- Output = sequence of classical states. *Ex*: $x_7x_3x_4$...

II(a). Quantum information, Non-Entanglement Source

- Produces *physical systems* (e.g., electrons) in *non-entangled quantum states* $\{|\psi_1\rangle, |\psi_2\rangle, ..., |\psi_n\rangle\}$.
- Output = sequence of quantum pure states. *Ex*: $|\psi_7\rangle|\psi_3\rangle|\psi_4\rangle$...

Timpson, C. (2008) "Philosophical Aspects of Quantum Information Theory", in D. Rickles (ed.) *The Ashgate Companion to the New Philosophy of Physics*, Ashgate.

II(b). Quantum information, Entanglement Source

- Produces *physical systems* (*i.e.*, electrons) in *entangled quantum states* which include other systems inaccessible to the source.
- Output = sequence of quantum entangled states.

Example of II(b):

$B = \{B_1, B_2, ...\} = \{\textit{electrons produced by source}\}$

$A = \{A_1, A_2, ...\} = \{\textit{electrons entangled with source electrons}\}$

$C = \{C_1, C_2, ...\} = \{\textit{"target" electrons at receiver}\}$

- *Suppose*: Electron B_i is produced at source in entangled state $|\psi\rangle_{A_iB_i}$ with electron A_i.
- *Goal*: To reproduce this entangled state at receiver, but between A_i and C_i: $|\psi\rangle_{A_iC_i}$
- *In general*: If source produces sequence of states

$$|\psi\rangle_{A_iB_i}|\psi'\rangle_{A_jB_j}|\psi''\rangle_{A_kB_k}\ldots,$$

then successful transmission occurs if receiver reproduces sequence of states

$$|\psi\rangle_{A_iC_i}|\psi'\rangle_{A_jC_j}|\psi''\rangle_{A_kC_k}\ldots\ .$$

Measures of information, depending on source

- The *Shannon Entropy*:

$$H(X) = -\sum_i p_i \log_2 p_i$$

  - $X = \{x_1, \ldots, x_n\}$, where x_i is a state produced by a classical information source, and p_i is a probability distribution over such states.

Specifies the minimal number of bits required to encode the output of a classical information source (Shannon 1948).

Aside!

Ex: Let $X = \{A, B, C, D\}$

- To encode X, need 2 bits per letter.
- *So*: Need $2N$ bits to encode an N-letter message.

$A = 00, \quad B = 01,$
$C = 10, \quad D = 11$

- *Suppose*: We have a probability distribution over X.
  - *Ex*: $p_A = 1/2, \quad p_B = 1/4, \quad p_C = p_D = 1/8$

Claim 1: There are $2^{NH(X)}$ possible N-letter messages.

$$\log_2\left(\begin{array}{l}\textit{\# possible N-letter}\\ \textit{messages}\end{array}\right) = \log_2\left(\frac{N!}{(p_A N)!\,(p_B N)!\,(p_C N)!\,(p_D N)!}\right) = NH(X)$$

Number of ways to arrange N distinct letters into 4 bins with capacities $p_A N$, $p_B N$, $p_C N$, $p_D N$.

Claim 2: 2^x messages require x bits to encode them.

"$\log_2 x = y$" means "$x = 2^y$"

- *So*: Instead of $2N$ bits, we only need $NH(X)$ bits, where

$$NH(X) = -N\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{4}\log_2\frac{1}{4} + \frac{1}{8}\log_2\frac{1}{8} + \frac{1}{8}\log_2\frac{1}{8}\right) = 1.75N$$

Measures of information, depending on source

- The *Shannon Entropy*:

$$H(X) = -\sum_i p_i \log_2 p_i$$

Specifies the minimal number of bits required to encode the output of a classical information source (Shannon 1948).

- $X = \{x_1, \ldots, x_n\}$, where x_i is a state produced by a classical information source, and p_i is a probability distribution over such states.

- The *von Neumann Entropy*:

$$S(\rho) = -\mathrm{Tr}(\rho \log_2 \rho) = -\sum_i p_i \log_2 p_i$$

Specifies the minimal number of qubits required to encode the output of a quantum information source (Schumacher 1995).

- $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$,where $|\psi_i\rangle$ is a vector state produced by a quantum information source, and p_i is a probability distribution over such states.