Playing "20 questions" with a quantum computer

David A. Meyer

Project in Geometry and Physics
Department of Mathematics
University of California/San Diego
La Jolla, CA 92093-0112
dmeyer@math.ucsd.edu

http://math.ucsd.edu/~dmeyer/

MATH 196/296 Student Colloquium
31 October 2001

Introduction

Computers which exploit quantum principles in their logic may be able to solve certain problems more efficiently than is possible classically.

R. P. Feynman, *International Journal of Theoretical Physics* **21** (1982) 467.
D. Deutsch, *Proceedings of the Royal Society of London A* **400** (1985) 97.
P. W. Shor, *SIAM Journal of Computing* **26** (1997) 1484.

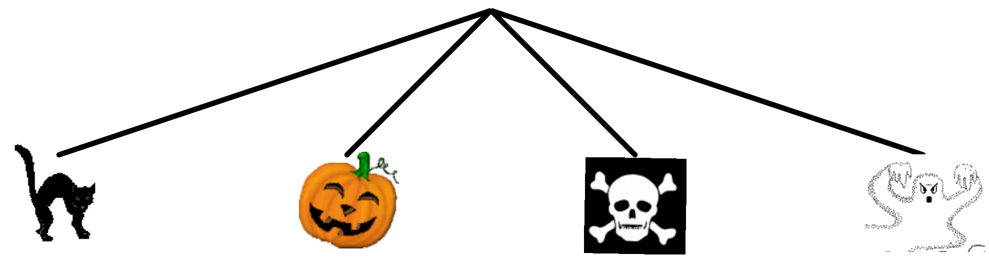
Outline

- example: the game of "20 questions"
- optimal classical strategies
- playing "20 questions" with a classical computer
- principles of quantum computing
- playing "20 questions" with a quantum computer

"20 questions"

Player 1 tries to determine what Player 2 is thinking by asking questions.

"Animal, vegetable or mineral?":



images from:

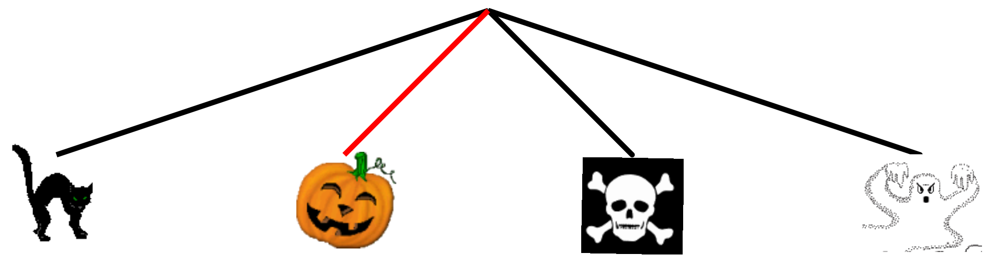
Kid's Domain Free Clipart

"20 questions"

Player 1 tries to determine what Player 2 is thinking by asking questions.

"Animal, vegetable, or mineral?":

Player 2 responds:



images from:

Kid's Domain Free Clipart

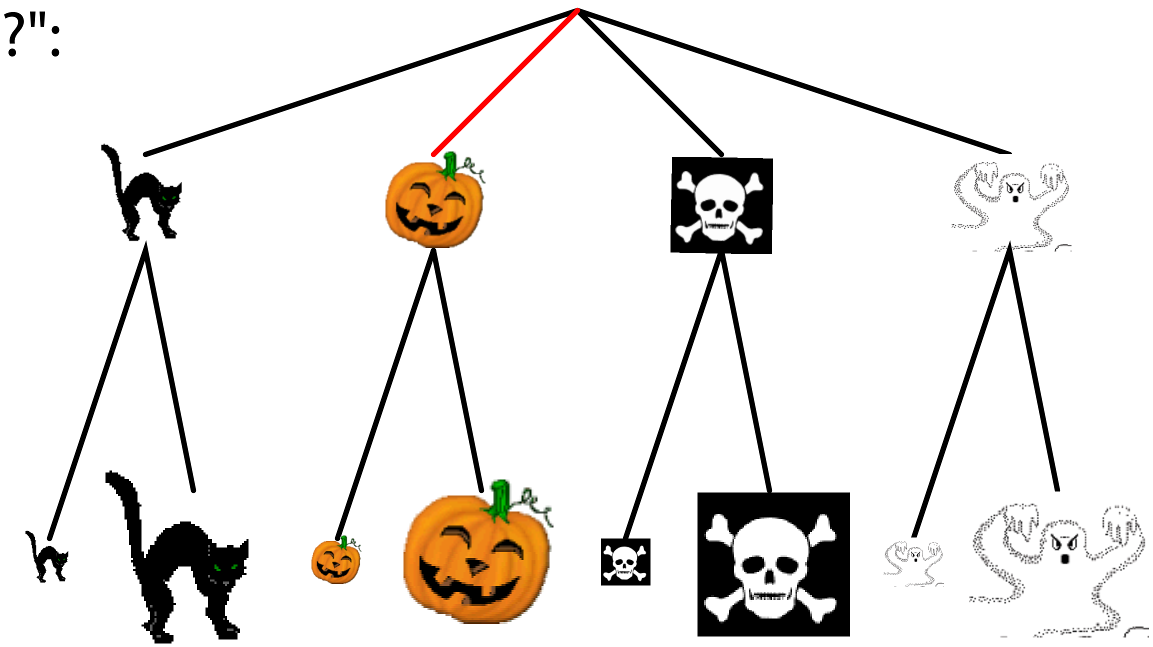
"20 questions"

Player 1 tries to determine what Player 2 is thinking by asking questions.

"Animal, vegetable, or mineral?":

Player 2 responds:

Then **YES/NO** questions like,
"Bigger than a breadbox?":



images from: Kid's Domain Free Clipart

"20 questions"

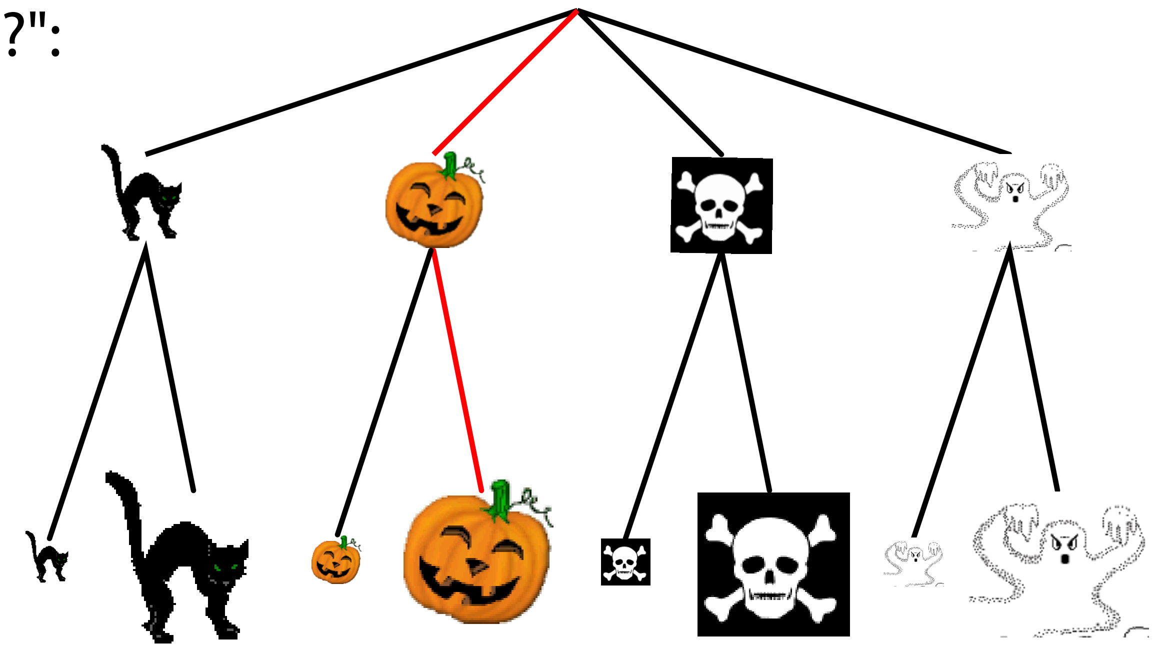
Player 1 tries to determine what Player 2 is thinking by asking questions.

"Animal, vegetable, or mineral?":

Player 2 responds:

Then **YES/NO** questions like,
"Bigger than a breadbox?":

Player 2 responds:



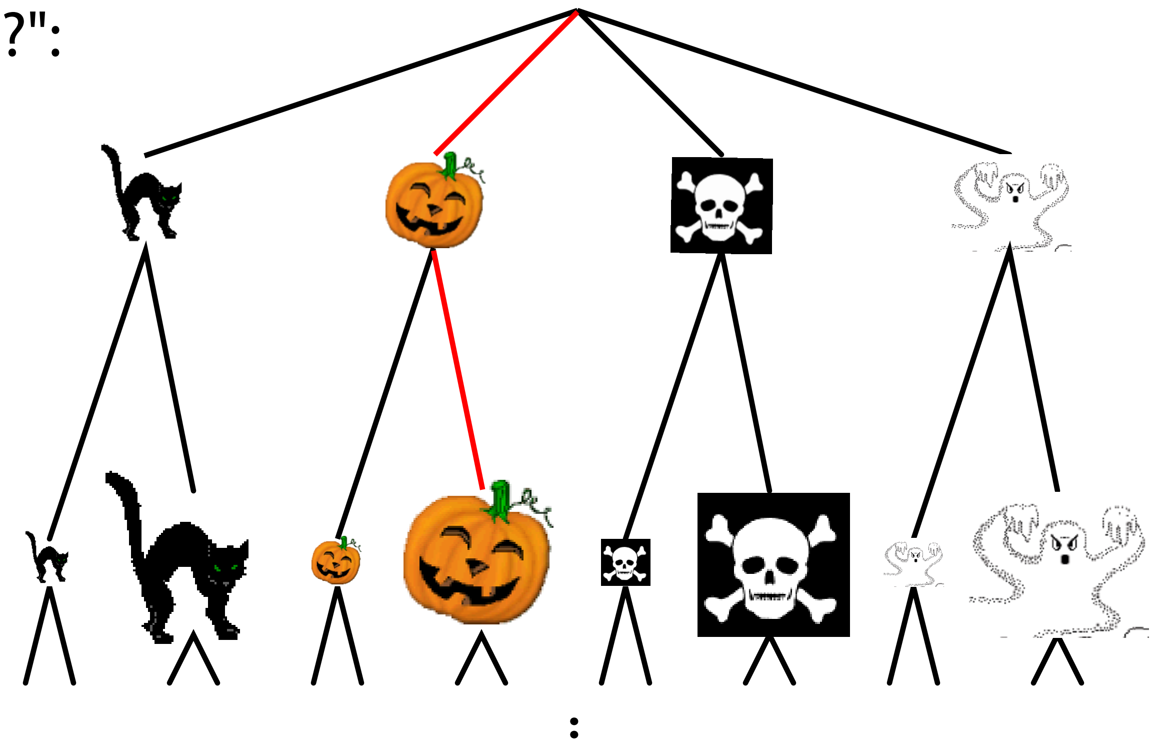
images from:

Kid's Domain Free Clipart

"20 questions"

Player 1 tries to determine what Player 2 is thinking by asking questions.

"Animal, vegetable, or mineral?":



Player 2 responds:

Then **YES/NO** questions like,
"Bigger than a breadbox?":

Player 2 responds:

...

Player 1 wins if s/he uses 20 or fewer questions to determine what Player 2 is thinking.

images from:

Kid's Domain Free Clipart

Search problems

So problem is to identify one out of $N = 2^n$ things w/ fewest questions.

Optimal strategy: each question should maximize amount of information received, independently of response.

Formalize this by defining:

$$\text{entropy } S = \text{lack of information}$$
$$= -\sum_{\text{possibilities } x} \text{prob}(x) \log \text{prob}(x)$$

Initially,

$$S_0 = -N\,(1/N) \log(1/N)$$
$$= -\log(1/N) = \log N\,,$$

assuming uniform distribution of Player 2's choices.

Optimal YES/NO questions

Reduce entropy as much as possible w/ each question.

Suppose fraction α of possibilities get response **YES**; then after question:

$$S_1 = -\alpha \sum_{\alpha N \text{ possibilities}} [1/\alpha N] \log[1/\alpha N] - (1-\alpha) \sum_{(1-\alpha)N \text{ possibilities}} [1/(1-\alpha)N] \log[1/(1-\alpha)N]$$

$$= \alpha \log \alpha N + (1-\alpha) \log (1-\alpha)N.$$

$$\frac{dS_1}{d\alpha} = \log \alpha N + \alpha[1/\alpha N]N - \log (1-\alpha)N + (1-\alpha)[1/(1-\alpha)N](-N) = 0,$$

which implies extremum at $\alpha = 1/2$, actually a minimum since at this value of α,

$$S_1 = (1/2)\log(N/2) + (1/2)\log(N/2) = \log(N/2) < \log N.$$

So optimal question partitions the possibilities evenly.

Optimal classical strategy

Similarly, questions with multiple responses (*e.g.*, "Animal, vegetable, or mineral?") ideally should partition possibilities evenly (assuming uniform prior distribution).

So optimal reduction of entropy is:

$$\log N \xrightarrow{Q_1} \log(N/4) \xrightarrow{Q_2} \log(N/8) \xrightarrow{Q_3} \ldots \xrightarrow{Q_{n-1}} \log(N/2^n) = 0.$$

If all questions were allowed to have 4 responses, optimal questions would reduce entropy like:

$$\log N \xrightarrow{Q_1} \log(N/4) \xrightarrow{Q_2} \log(N/4^2) \xrightarrow{Q_3} \ldots \xrightarrow{Q_{n/2}} \log(N/4^{n/2}) = 0.$$

That is, in worst case Player 1 will require at least $\log_4 N = n/2$ questions.

Classical computer

Standardize by using numerical labels, written as *n*-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an *n*-bit string *x*.

Player 2 responds w/ the Hamming distance from the answer *a* (the number of wrong bits), dist(*x*,*a*) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

1	1	0	0	1	0	1	1	0	1

1	0

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	0	0	1	0	1	1	0	1

0	0

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	0	1	0	1	1	0	1

1	1

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	1	1	0	1	1	0	1

1	0

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	1	0	0	1	1	0	1

1	1

Classical computer

Standardize by using numerical labels, written as *n*-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an *n*-bit string *x*.

Player 2 responds w/ the Hamming distance from the answer *a* (the number of wrong bits), dist(*x*,*a*) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(*x*,*a*) = 5 | 0 | 1 |

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	1	1	1	1	1	0	1

| 1 | 1 |

Classical computer

Standardize by using numerical labels, written as *n*-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an *n*-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5 | 0 | 1 |

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	1	1	0	0	1	0	1

0	1

Classical computer

Standardize by using numerical labels, written as n-bit strings, for possibilities.

Allow every question to have 4 possible responses:

Player 1 names an n-bit string x.

Player 2 responds w/ the Hamming distance from the answer a (the number of wrong bits), dist(x,a) mod 4.

x =	0	1	0	0	1	0	1	1	0	1
a =	0	0	1	1	1	0	0	0	0	1

0	1	1	1	0	0	1	1	0	0

dist(x,a) = 5

0	1

response is 5 mod 4 = 1

Optimal solution must require at least $n/2$ queries; n queries is easy:

0	0	1	1	1	0	0	0	0	1

0	0

Invertible classical computing

Initialize query and response registers to:

0	0	0	0	0	0	0	0	0	0		0	0

Player 1 prepares query:

$\xrightarrow{+x}$	0	1	0	0	1	0	1	1	0	1		0	0

Player 2 responds:

$\xrightarrow{+\text{dist}(x,a)}$	0	1	0	0	1	0	1	1	0	1		0	1

Each of these is an invertible operation, unlike *x* **AND** *y*, for example.

All classical computations can be done w/ (local) invertible operations.

C. H. Bennett, *IBM Journal of Research and Development* **17** (1973) 525.

Invertible computation *via* matrix multiplication

Possible states of computer are, say, $(n+2)$-bit strings; have 2^{n+2} of them.

Operations convert one bit string into another; can think of as matrix multiplication:

$$\underbrace{\overset{\begin{matrix}00 & 01 & 10 & 11\end{matrix}}{\begin{bmatrix}0 & 0 & 0 & 1\\ 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}}}_{T}\begin{bmatrix}0\\1\\0\\0\end{bmatrix}\begin{matrix}00\\01\\10\\11\end{matrix} = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}\begin{matrix}00\\01\\10\\11\end{matrix}$$

Invertible operations are represented by permutation matrices, *i.e.*, those w/ exactly one 1 in each row and column.

Quantum evolution

Evolution in quantum mechanics is represented by multiplication by unitary matrices.

U is unitary if and only if $U(U^*)^{\mathrm{T}} = I = (U^*)^{\mathrm{T}}U$.

Notice that permutation matrices are unitary:

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So evolution in quantum mechanics generalizes invertible classical evolution.

Quantum states in quantum computers

States of quantum systems are linear combinations of classical states.

So a single quantum bit (qubit) can be in any state:

$$a_0 \boxed{0} + a_1 \boxed{1}, \text{ where } |a_0|^2 + |a_1|^2 = 1.$$

The reason for the condition on the coefficients it that when the qubit is measured (relative to this basis), it is always either $\boxed{0}$ or $\boxed{1}$, with probabilities $|a_0|^2$ and $|a_1|^2$, respectively.

Two qubits, like the response register, can be in any state of the form:

$$a_{00} \boxed{0\,|\,0} + a_{01} \boxed{0\,|\,1} + a_{10} \boxed{1\,|\,0} + a_{11} \boxed{1\,|\,1},$$

where again the sum of the squared norms of the coefficients is 1.

Unitary operations in quantum computers

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \; X \boxed{0} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \boxed{1}\,.$$

$$H = 2^{-1/2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \; H \boxed{0} = \left(\boxed{0} + \boxed{1}\right)/2^{1/2}\,.$$

$$T \boxed{0\,|\,0} = \boxed{0\,|\,1}\,.$$

$$F = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} / \, 2\,;$$

$$F \boxed{0\,|\,1} = \left(\boxed{0\,|\,0} + i \boxed{0\,|\,1} - \boxed{1\,|\,0} - i \boxed{1\,|\,1}\right)/2.$$

Tensor products of matrices

$$H \otimes I = \left[\begin{array}{c|c} I & I \\ \hline I & -I \end{array}\right] / 2^{1/2} = \left[\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{array}\right] / 2^{1/2}$$

$$H \otimes H = \left[\begin{array}{c|c} H & H \\ \hline H & -H \end{array}\right] / 2^{1/2} = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{array}\right] / 2$$

A quantum algorithm (M. Hunziker and D. A. Meyer, UCSD preprint (2001).)

Initialize query and response registers to:

$$\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\boxed{0}\quad\boxed{0}\boxed{0}$$

Player 1 prepares query:

$$\xrightarrow{H^{\otimes n}\otimes FT} (\boxed{0}+\boxed{1})\otimes\cdots\otimes(\boxed{0}+\boxed{1})/2^{n/2}$$

$$\otimes(\boxed{0}\boxed{0}+i\,\boxed{0}\boxed{1}-\boxed{1}\boxed{0}-i\,\boxed{1}\boxed{1})/2$$

$$=(\text{all } n\text{-bit strings } \boxed{\mathbf{x}})/2^{n/2}$$

$$\otimes(\boxed{0}\boxed{0}+i\,\boxed{0}\boxed{1}-\boxed{1}\boxed{0}-i\,\boxed{1}\boxed{1})/2$$

Player 2 responds to the quantum query:

$$\xrightarrow{+\mathrm{dist}(x,a)} \sum(-i)^{\mathrm{dist}(x,a)}\,\boxed{\mathbf{x}}\,/\,2^{n/2}\otimes FT\,\boxed{0}\boxed{0}$$

Only 1 quantum question!

Define a new unitary matrix

$$G = \begin{bmatrix} 1 & \mathrm{i} \\ \mathrm{i} & 1 \end{bmatrix} / 2^{1/2},$$

which Player 1 uses to "interpret" the response:

$$\xrightarrow{G^{\otimes n} \otimes I_4} \sum (-\mathrm{i})^{\mathrm{dist}(x,a)} (\mathrm{i})^{\mathrm{dist}(y,x)} \boxed{\mathtt{y}} / 2^n \otimes FT \boxed{0}\boxed{0} = \boxed{\mathtt{a}} \otimes FT \boxed{0}\boxed{0}.$$

Here the sum is over x and y. The 2^n terms which have $y = a$ each contribute $1/2^n$; the rest exactly cancel.

Thus "20 questions" becomes "1 question" with a quantum computer.