ドイチ-ジョサのアルゴリズム

関数がバランス関数か定数関数かを判定するアルゴリズム

・定数関数（2bit）：すべての出力が等しい

f(00)=f(01)=f(10)=f(11)=0または1

・バランス関数（2bit）：

f(00), f(01), f(10), f(11)のうち半分が0で半分が1

ここでは中途半端な関数（例：f(00)=1で他の3つは0）は存在しないと考える

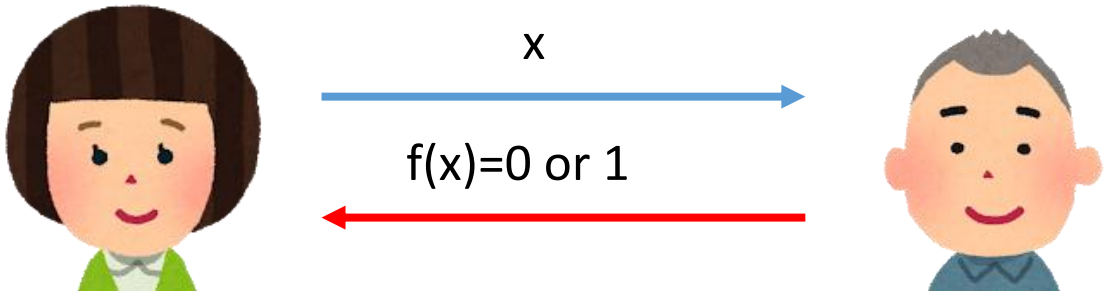
登場人物

Alice：f(x)が定数関数なのかバランス関数なのか知りたいと思っている

Bob：f(x)を知っている（計算できる）

Alice：Bobにxを投げる

Bob：f(x)を返す

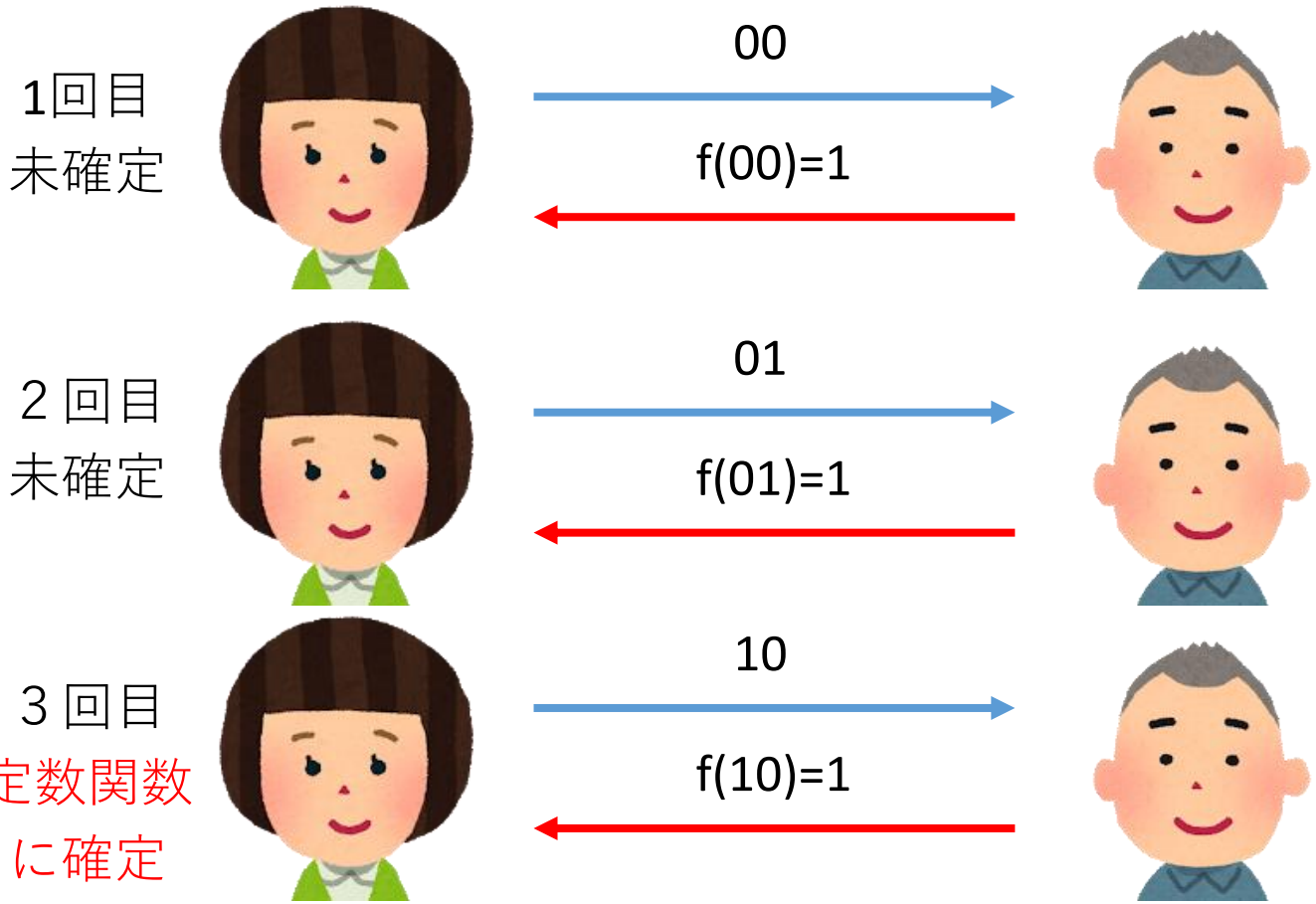


AliceはBobに色々なxを投げて、

帰ってきたf(x)を見て定数関数かバランス関数かを確定させる

古典コンピュータの場合・・・

最悪で2^(ビット数-1)+1回の質問を行わないと、関数を確定させることはできない



量子コンピュータの場合・・・

１回で確定できる

1回目

確定

00と01と10と11が

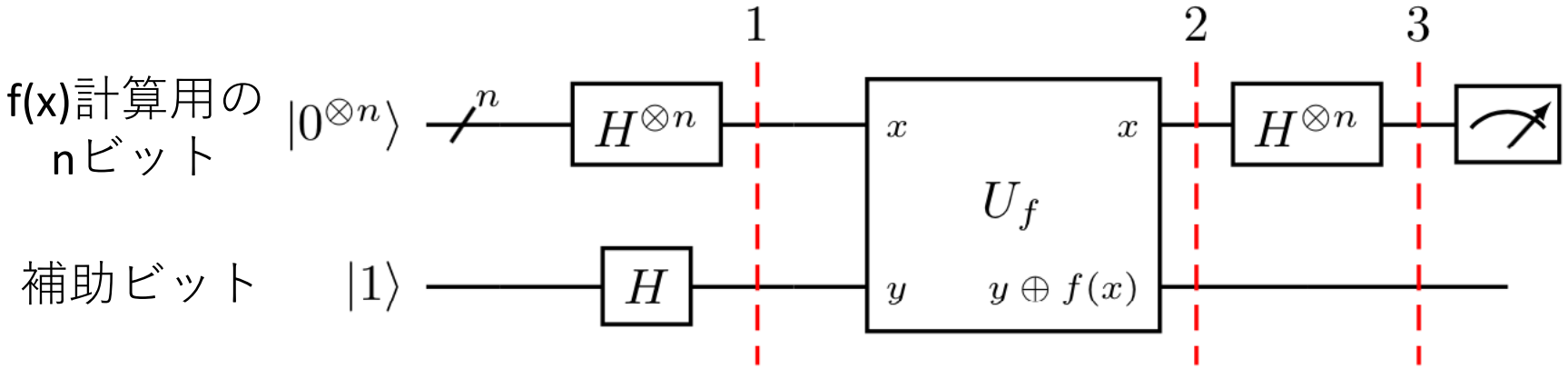
重なり合った量子ビット

測定したビット

Bobから帰ってきたビットが00なら定数関数

それ以外ならバランス関数

ドイチ-ジョサのアルゴリズムの量子回路

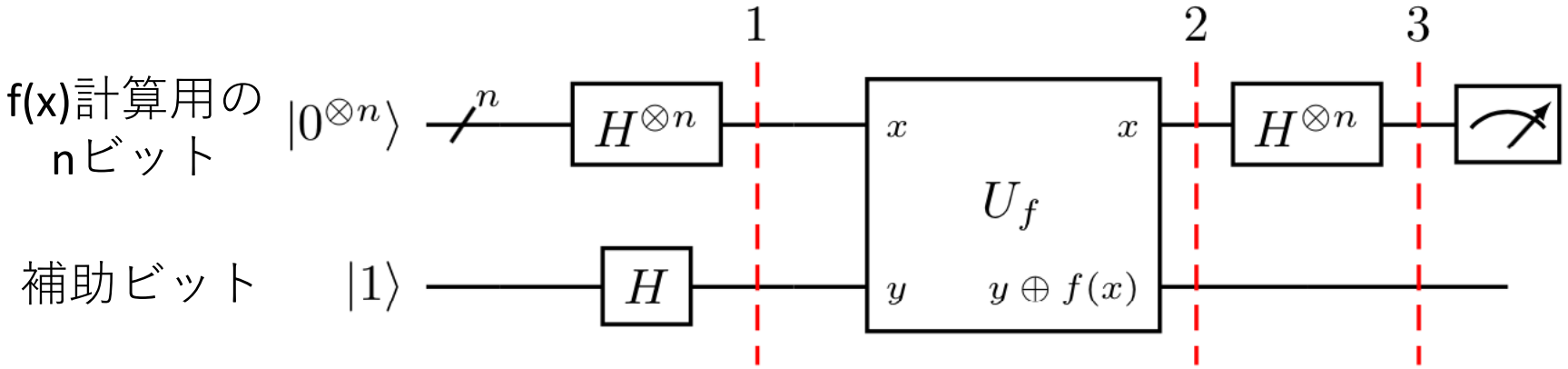


https://qiskit.org/textbook/ja/ch-algorithms/deutsch-jozsa.html

①アダマールゲートで等確率の重ね合わせ状態を作成

$$|\psi_1\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle \left(|0\rangle - |1\rangle\right)$$

ドイチ-ジョサのアルゴリズムの量子回路



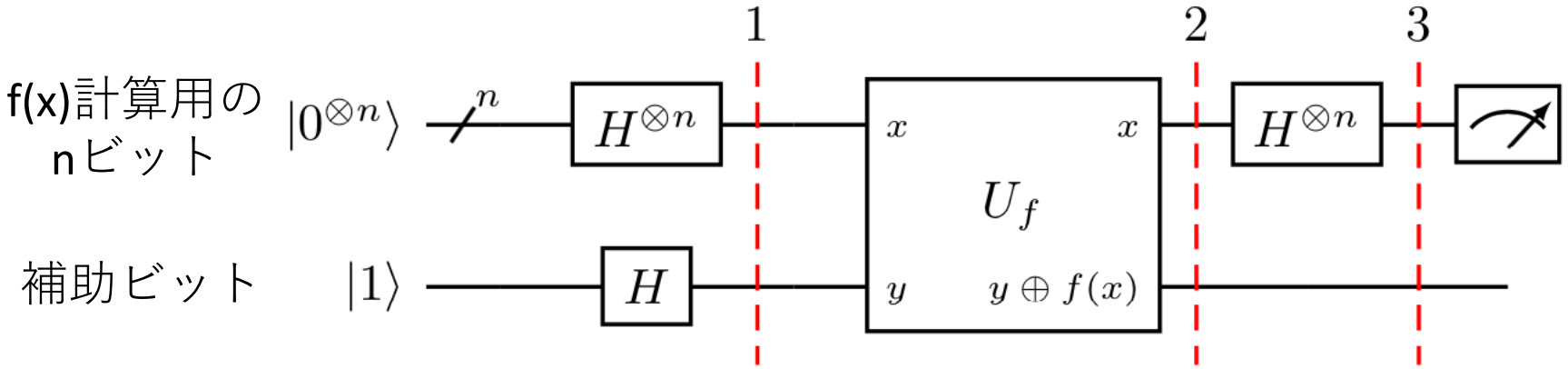
https://qiskit.org/textbook/ja/ch-algorithms/deutsch-jozsa.html

②オラクルUfにて、補助ビット$|b\rangle \rightarrow |b \oplus f(x)\rangle$
（bと f(x)の排他的論理和）

$$|\psi_2\rangle = \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} |x\rangle (|f(x)\rangle - |1 \oplus f(x)\rangle)$$

$$= \frac{1}{\sqrt{2^{n+1}}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle (|0\rangle - |1\rangle)$$

ドイチ-ジョサのアルゴリズムの量子回路

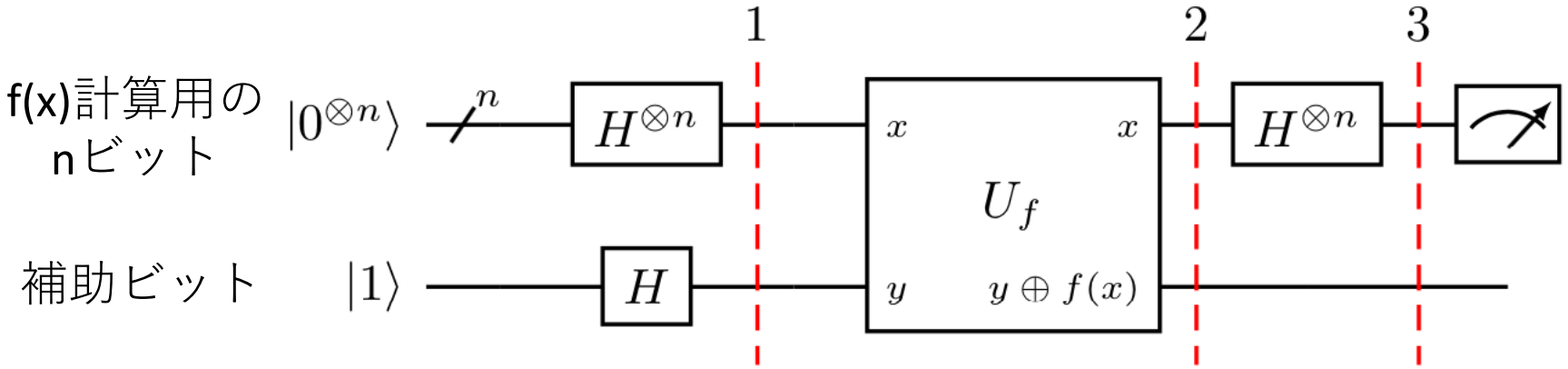


https://qiskit.org/textbook/ja/ch-algorithms/deutsch-jozsa.html

③アダマールゲートをかけて状態を戻す
（②で補助ビットが反転した部分が計算用ビットに反映される
：位相キックバック）

$$|\psi_3\rangle = \frac{1}{2^n}\sum_{x=0}^{2^n-1}(-1)^{f(x)}\left[\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle\right]$$
$$= \frac{1}{2^n}\sum_{y=0}^{2^n-1}\left[\sum_{x=0}^{2^n-1}(-1)^{f(x)}(-1)^{x\cdot y}\right]|y\rangle$$

ドイチ-ジョサのアルゴリズムの量子回路



https://qiskit.org/textbook/ja/ch-algorithms/deutsch-jozsa.html

④測定する

測定結果が$|0 \ldots 0\rangle$（全てのビットが0）になる確率は

定数関数で1

バランス関数で0

関数によって作成するオラクルが変わる

考え方：f(x)=1になっている場合に補助ビットの符号を反転させたい

		x				オラクル
		00	01	10	11	
定数関数	f1(x)	0	0	0	0	何もなし
	f2(x)	1	1	1	1	補助ビットにXゲート
バランス関数	f3(x)	0	0	1	1	q_1→q_2にCXゲート（q_1が共通で1なので）
	f4(x)	0	1	0	1	q_0→q_2にCXゲート（q_0が共通で1なので）
	f5(x)	0	1	1	0	q_0→q_2にCXゲート、q_1→q_2にCXゲート （2つの量子ビットのうち1個が1のときに補助ビットを反転）
	f6(x)	1	0	0	1	q_0にXゲート、q_0→q_2にCXゲート、q_1→q_2にCXゲート、q_0にXゲート
	f7(x)	1	0	1	0	q_0にXゲート、q_0→q_2にCXゲート、q_0にXゲート（q_0が共通で0なので）
	f8(x)	1	1	0	0	q_1にXゲート、q_1→q_2にCXゲート、q_1にXゲート（q_1が共通で0なので）

誤解しやすいところ

アルゴリズムは1人で実装するのでは？

→アリスはオラクルの実装には関わらない

関数の種類によってオラクルは変化しますので、アリスがオラクルを実装したということは、すなわちアリスはアルゴリズムを動かす前から関数の種類を知っているということになります。それでは本末転倒ではないでしょうか。オラクルはブラックボックスです。

アリスは関数の種類によって行動は変わる？

→アリスの行動は変わらない（ボブが実装するオラクルが変わる）

アリスはただ2^nのxをオラクルに投げるだけです。関数はアリスの意思とは関係なく決定されます。ほぼ上と被っている話です。

関数の種類を判別するのがオラクル？

→xに対してf(x)の値を返すのがオラクル

オラクルは判別しているわけではありません。ただf(x)という値を返しているだけです。古典コンピュータでは一度問い合わせるときに一つのxしか尋ねることができませんが、量子コンピュータであれば2^nのxの重ね合わせ状態を問合せます。ので、結果的にオラクルが関数の種類を判別しているということと同値になります。

参考：https://qiita.com/mi2valley/items/f39777cd1f9592a753eb