Introducción a Hadoop y Spark

Procesamiento de Datos a Gran Escala

José Antonio Álvarez Ocete
Francisco Javier Sáez Maldonado

29 de octubre de 2021

Índice

1. Parte 1: Programación GPGPU **1**
  1.1. Recursos de la GPU 1
  1.2. Suma de 2 vectores 2
  1.3. Suma de 2 matrices 2
  1.4. Stencil1d: Estudiar el efecto de la memoria compartida 2

2. Parte 2: Programación con QisKit: (Computación cuántica) **2**
  2.1. Puertas Cuánticas 2
  2.2. Generación de números aleatorios con un Computador Cuántico 4
  2.3. Entrelazamiento 5
  2.4. Sumador de 2 qbits 5

3. Ejercicios opcionales de la práctica 2 **5**

Dividiremos esta práctica en 2 partes fundamentales.

1. En la primera, se concentrará en realizar diversas pruebas con los elementos más básicos de la programación en CUDA.

2. En la segunda, TODO!!!!!!!!!!

1. Parte 1: Programación GPGPU

1.1. Recursos de la GPU

Para programar en GPU, el primer paso que debemos dar es conocer las características (y por tanto, los recursos) que el dispositivo del que disponemos nos ofrece. En este caso práctico, vamos a realizar las pruebas utilizando Google Colaboratory. Google pone a disposición del usuario tanto GPUs como TPUs que son más que suficientes para realizar algunas pruebas.
Lo primero que haremos es estudiar qué recursos tiene la GPU que nos ofrecen

1.2. Suma de 2 vectores

1.3. Suma de 2 matrices

1.4. Stencil1d: Estudiar el efecto de la memoria compartida

2. Parte 2: Programación con QisKit: (Computación cuántica)

Para esta parte de la práctica utilizaremos el Quantum Composer de IBM. Puesto que el tenemos un número limitado de procesos a ejecutar (únicamente 5) veremos los resultados en el simulador sin llegar a medirlo en muchos casos.

2.1. Puertas Cuánticas

TODO: Añadir enunciado

Puerta CNOT La única operación no trivial aplicable sobre un único bit es la negación: la puerta NOT. De la misma forma, es natural preguntarse cuál es el equivalente a la puerta NOT en el mundo cuántico. Dado que un qubit está descrito por dos amplitudes α y β:

$$|\varphi\langle = \alpha|0\langle + \beta|1\langle,$$

la puerta NOT será un intercambio entre las posiciones de estas amplitudes, obteniéndose así:

$$|\varphi\langle = \beta|0\langle + \alpha|1\langle.$$

La matriz unitaria que describe esta transformación es sencilla:

$$X = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Vemos la implementación de esta puerta en el Quantum Composer de IBM:

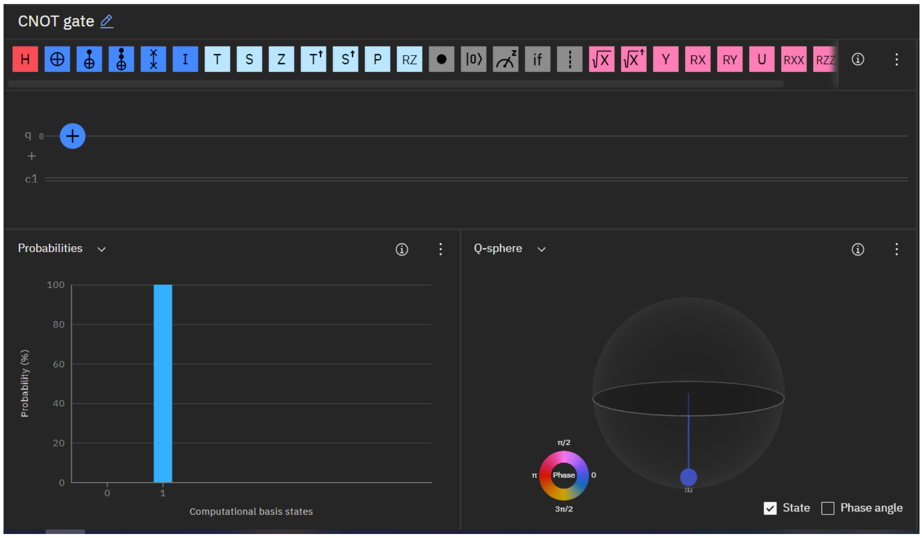

Figura 1: Circuito con puerta de Hadamard

Recordemos que en el Quantum Composer de IBM todos los qubits empiezan siempre en el estado $|0\langle$. Tras aplicarlo a nuestro qubit una puerta X obtendremos $|1\rangle$.

Puertas de Pauli

Puerta Hadamard Finalmente presentamos la puerta de Hadamard para un único bit. Está descrita por la siguiente matriz unitaria:

$$H_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Uno de sus usos más comunes es la superposición de qubits. Si aplicamos esta puerta al estado $|0\langle$ obtenemos el estado de Bell:

$$H|0\langle = \frac{1}{\sqrt{2}}|0\langle + \frac{1}{\sqrt{2}}|1\langle = |+\langle$$

Mientras que si se la aplicamos al estado $|1\langle$ obtenemos:

$$H|1\langle = \frac{1}{\sqrt{2}}|0\langle - \frac{1}{\sqrt{2}}|1\langle = |-\langle$$

Que también supone una superposición exacta de $|0\langle$ y $|1\langle$ puesto que $|1/\sqrt{2}|^2 = |-1/\sqrt{2}|^2 = 1/2$.
Podemos estudiar el comportamiento de esta puerta utilizando el Quantum Composer de IBM:

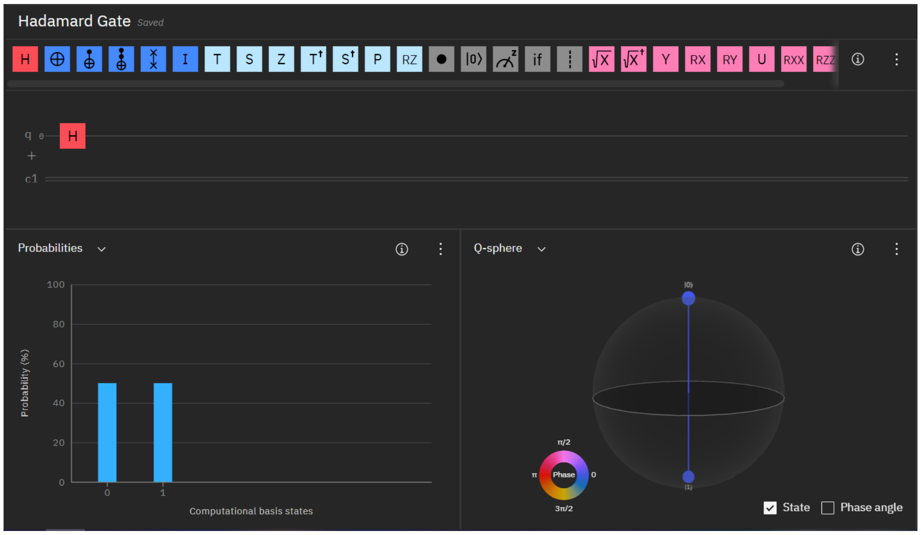

Figura 2: Circuito con puerta de Hadamard

Mirando tanto la esfera de Bloch como las probabilidades vemos que tenemos la misma probabilidad de medir 0 y 1.

2.2. Generación de números aleatorios con un Computador Cuántico

TODO: añadir enunciado
Sabemos que utilizando la puerta de Hadarmad H explicada en el apartado anterior ponemos un qubit $|0\langle$ en superposición:

$$H|0\langle= \frac{1}{\sqrt{2}}|0\langle+\frac{1}{\sqrt{2}}|1\langle$$

Si ahora medimos este qubit obtendremos $|0\langle$ con probabilidad $|1/\sqrt{2}|^2 = 1/2$, y $|1\rangle$ con probabilidad $1/2$. Esto es, hemos creado un generador de bits aleatorios utilizando un único qubit. Para crear un generador de 3 bits utilizaremos un sistema de 3 qubits. Inicialmente en el estado $|000\rangle$, aplicaremos una puerta Hadamard a cada qubit de forma independiente, poniendo así cada qubit en superposición:

$$\hat{H}_3|000\langle= \frac{|000\langle+|001\langle+|010\langle+|011\langle+|100\langle+|101\langle+|110\langle+|111\langle}{\sqrt{8}}|0\langle$$

Donde la puerta H_3 tranformación de Hadamard para tres qubits. Se puede definir recursivamente de la siguiente forma:

$$H_m = H_1 \times H_{m-1}, H_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Así obtenemos la puerta Hadamard para tres qubits:

$$H_3 = \frac{1}{2^{3/2}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}$$

Pasamos a realizar un estudio empírico del circuito diseñado. Lo implementamos utilizando el Quantum Composer de IBM:
Este simple circuito cuántico pone todos los qubits en superposición y después mide el resultado. Comprobamos los resultados ejecutando un trabajo con este circuito en el ordenador cuántico de IBM:

2.3. Entrelazamiento

2.4. Sumador de 2 qbits

3. Ejercicios opcionales de la práctica 2