QUANTUM TELEPORTATION

Vella Mirco – S4803218
Haider Ali – S4811831

Nei computer quantistici non è possibile clonare gli stati, ovvero sostanzialmente copiarne le informazioni. Questo a causa del teorema no-cloning, che afferma in particolare che non sia possibile copiare uno stato quantistico sconosciuto a priori, in quanto non siamo a conoscenza attualmente di un operatore generico capace di clonare stati.

Tuttavia è possibile teletrasportare le informazioni contenute in uno stato tramite il sopracitato algoritmo di teletrasporto quantistico, che sfrutta uno stato entangled per lo scambio di informazioni.
Con questo algoritmo possiamo trasmettere l'intero stato senza distruggerlo.

Per implementarlo abbiamo creato due versioni diverse, una simulabile, "QuantumTeleportation_REAL", e una eseguibile, "QuantumTeleportation_MOD", della quale discuteremo dopo.

Per iniziare, dopo aver importato le librerie utili, abbiamo creato i tre qubit che entrano in gioco per il teletrasporto e i tre registri classici utilizzati per le misurazioni.

In [3]:
```
#Creazione del circuito per il teletrasporto
qc = QuantumRegister(3) #Tre qubit
crz = ClassicalRegister(1) #Primo bit classico
crx = ClassicalRegister(1) #Terzo bit classico
final = ClassicalRegister(1) #Bit finale classico -> conterrà 0 se il teletrasporto funziona
teleportation_circuit = QuantumCircuit(qc, crz, crx, final) #Compongo il circuito
#Circuito iniziale
teleportation_circuit.draw()
```

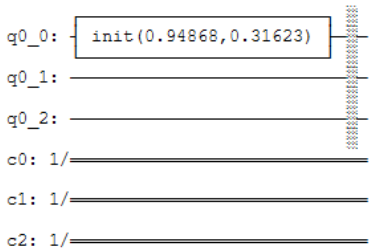
Out[3]:
```
q0_0:

q0_1:

q0_2:

c0: 1/

c1: 1/

c2: 1/
```

Dopodiché abbiamo composto il circuito, denominato "teleportation_circuit".
Fondamentale per questo algoritmo, come già detto, è lo stato entangled condiviso tra Alice e Bob, che creiamo attraverso l'applicazione di una porta di Hademard sul

secondo qubit e una porta CNOT sul terzo bit (controllato dal secondo). Prima di ciò prepariamo anche il qubit da teletrasportare, con le probabilità date (in questo caso l'immagine presenta l'inizializzazione con 0.9 e 0.1).

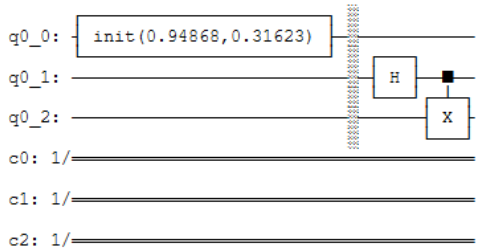
In [4]:
```
initial_state = [sqrt(0.9), sqrt(0.1)]  #AlphaQuadro=0.9 BetaQuadro=0.1
#initial_state = [sqrt(0.3), sqrt(0.7)]  #AlphaQuadro=0.3 BetaQuadro=0.7
init_state = Initialize(initial_state) #Uso questa funzione per poter dopo applicare un disentangler
init_state.label = "init"

teleportation_circuit.append(init_state, [0]) #Inizializzo primo bit con le probabilità sopra descritte
teleportation_circuit.barrier()
#Vediamo come siamo messi
teleportation_circuit.draw()
```

Out[4]:



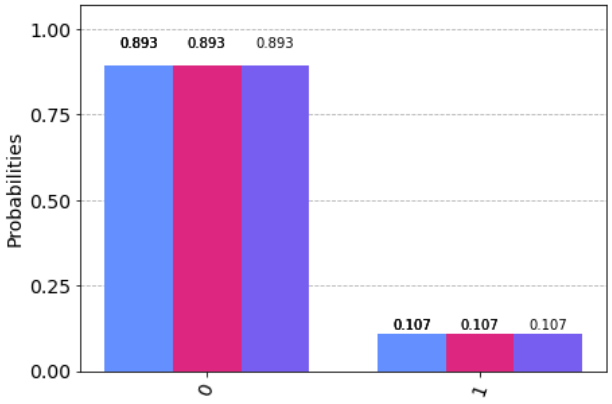
In [5]:
```
#PRIMO STEP -> creo bit entangled
#Creo bit entangled di A e B
teleportation_circuit.h(1) #Hademard sul secondo qubit generale
teleportation_circuit.cx(1,2) #CNOT  su terzo bit  (2) controllato dal secondo (1)
#Vediamo come siamo messi
teleportation_circuit.draw()
```

Out[5]:

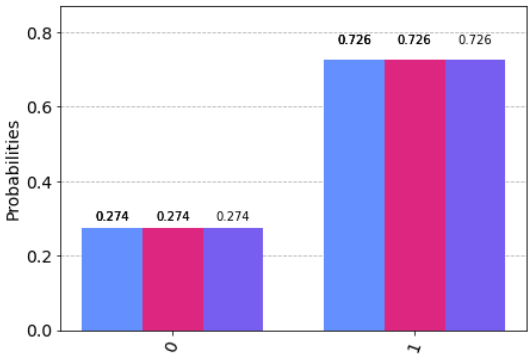


L'inizializzazione del qubit da teletrasportare prevede che una possibile misurazione su di esso dia all'incirca il 90% delle volte 0 e il 10% delle volte 1, vediamo se è così.

Dal grafico vediamo che effettivamente i risultati sono in linea con le attese.



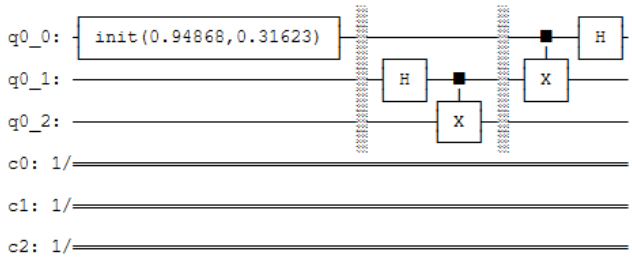
È ovvio che cambiando le probabilità, quindi in particolare adottando 0.3 e 0.7 come da specifica, vediamo come il risultato sia diverso ma sempre coerente con le aspettative.



Torniamo ora all'algoritmo di teletrasporto; abbiamo uno stato entangled condiviso tra Alice e Bob e un qubit in possesso di Alice; il prossimo passo è collegare quest'ultimo allo stato condiviso, in modo da permettere il "passaggio" di informazioni.

In [6]:
```
#SECONDO STEP -> "collego" il messaggio al qubit entangled
teleportation_circuit.barrier() #Barriera
teleportation_circuit.cx(0, 1) #CNOT su secondo bit (1) controllato dal primo bit (0)
teleportation_circuit.h(0) #Hademard sul messaggio
#Circuito aggiornato
teleportation_circuit.draw()
```

Out[6]:

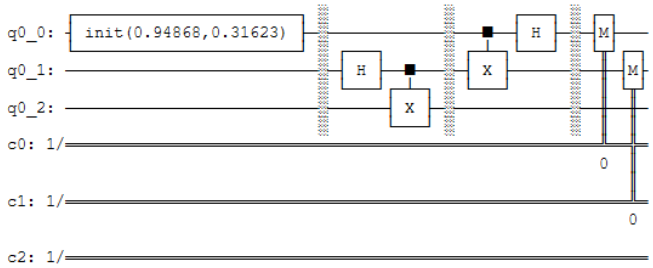


Piccola nota, le barriere presenti nel circuito servono sia a rendere il circuito più ordinato sia ad impedire al compilatore di modificare il codice, rendendolo più efficiente ma modificandone le caratteristiche.

Arrivati a questo punto si passa alle misurazioni, effettuate da Alice sui suoi due qubit. Dato che i qubit di Alice e Bob sono entangled, una misura porta ad un collasso dello stato di Bob. Le misure di Alice vengono riportate su due registri classici.

In [7]:
```
#TERZO STEP -> misura di A
#A effetuua una misura sui due qubit che ha, il risultato sarà nei registri classici
teleportation_circuit.barrier()
teleportation_circuit.measure(0,crz) #Misura del primo qubit nel registro classico crz
teleportation_circuit.measure(1,crx) #Misura del secondo qubit nel registro classico crx
#Circuito aggiornato
teleportation_circuit.draw()
```

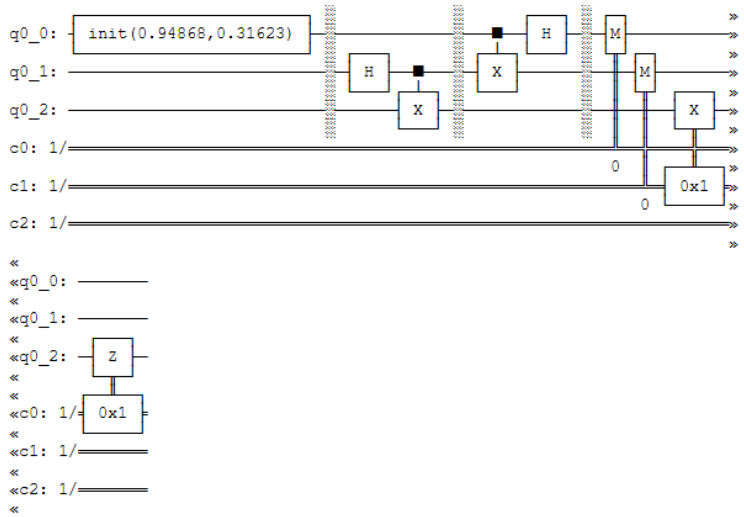
Out[7]:



Bob, quindi, vede cosa è stato misurato da Alice e applica delle porte correttive, con le casistiche riportate nell'immagine.

In [8]:
```
#ULTIMO STEP -> porta correttiva applicata da B
# 00 -> Niente
# 10 -> Z
# 01 -> X
# 11 -> XZ
teleportation_circuit.x(2).c_if(crx, 1) #Applico porta di correzione X se ho 1 nel registro
teleportation_circuit.z(2).c_if(crz, 1) #Applico porta di correzione Z se ho 1 nel registro
#Circuito aggiornato
teleportation_circuit.draw()
```

Out[8]:



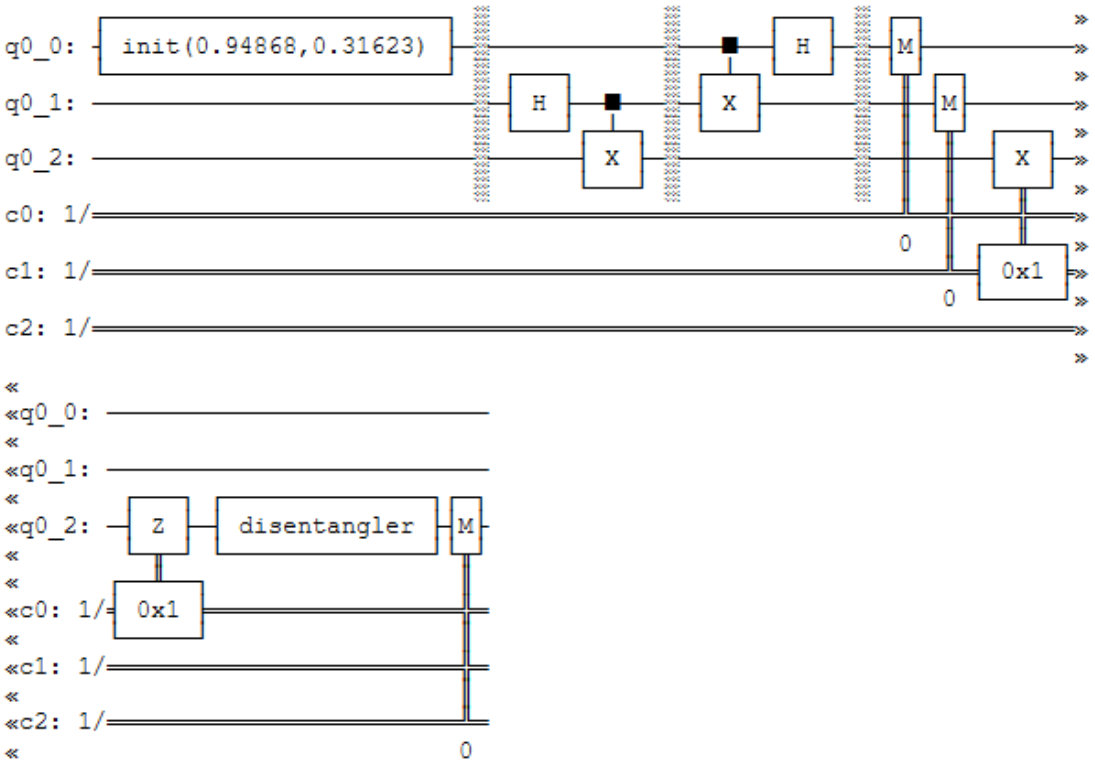
Ora per verificare che il teletrasporto sia andato a buon fine bisogna fare una misura sullo stato di Bob. Per rendere la cosa più chiara applichiamo una trasformazione inversa al qubit teletrasportato, in modo tale da ottenere sempre 0 in caso di successo.

Le ultime operazioni fatte sono quindi le seguenti:

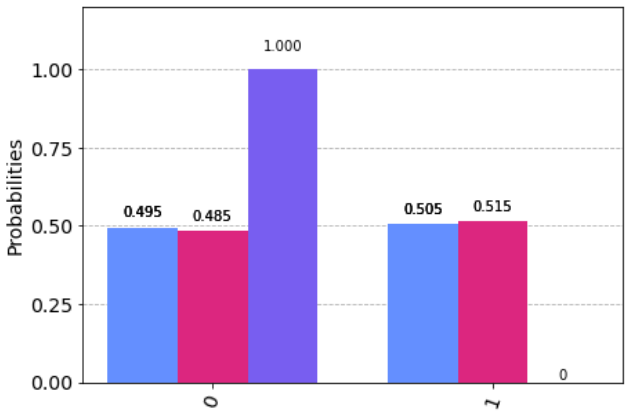
In [9]:
```
#Disentangler
inverse_init_state = init_state.gates_to_uncompute() #Porta che mi dovrebbe dare sempre 0
teleportation_circuit.append(inverse_init_state, [2]) #Applico trasformazione inversa
#Circuito aggiornato
teleportation_circuit.draw()
```

```
#Misura finale, se tutto è andato bene dovrei avere 0 nell'ultimo bit
teleportation_circuit.measure(2,2)
teleportation_circuit.draw()
```

Il circuito completo ha invece questo aspetto:

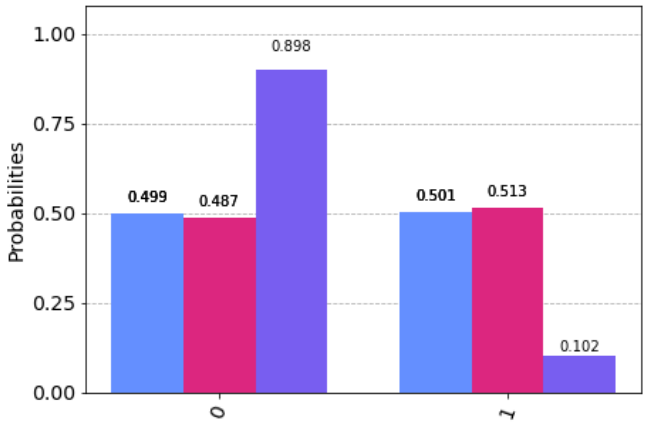


Simulando il circuito otteniamo correttamente il 100% delle volte 0 nel terzo bit, confermando le aspettative:



Senza il disentangler finale otterremmo le stesse probabilità date in fase di inizializzazione, circa:

- 90% ‘1’ / 10% ‘0’
- 30% ‘1’ / 70% ‘0’



VERSIONE ESEGUIBILE

(QuantumTeleportation_MOD)

Arrivati a questo punto ci sembrava utile verificare la potenza di questo algoritmo su un computer quantistico vero, in modo da vedere se fosse in linea con le aspettative. Infatti ci aspettavamo che i risultati presentassero qualche caso di insuccesso, dovuto alle normali interferenze presenti nella realtà.

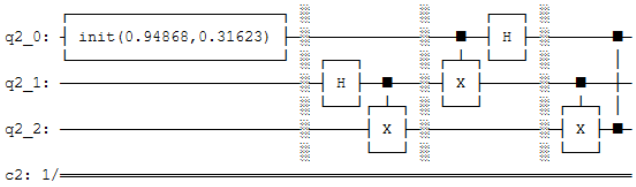
Tuttavia quando abbiamo provato a mandare il codice in esecuzione ad un computer quantistico ci è stato restituito il seguente messaggio:

```
'Instruction bfunc is not supported. Error code: 7001.'
```

Facendo una breve ricerca abbiamo scoperto che la causa dovrebbe essere l’utilizzo delle istruzioni condizionali, a quanto pare non ancora implementate nei computer quantistici; questo imponeva dei cambiamenti al nostro codice.
Abbiamo trovato che secondo un principio chiamato “Principio di misurazione differita” è possibile nel nostro caso rimuovere i comandi if e sostituirli con porte CNOT (funzione cx) e CZ (funzione cz).

```
In [19]: #ULTIMO STEP -> porta correttiva applicata da B
         # 00 -> Niente
         # 10 -> Z
         # 01 -> X
         # 11 -> XZ
         teleportation_circuit.cx(1, 2)
         teleportation_circuit.cz(0, 2)
         #teleportation_circuit.x(2).c_if(crx, 1) #Applico porta di correzione X se ho 1 nel registro
         #teleportation_circuit.z(2).c_if(crz, 1) #Applico porta di correzione Z se ho 1 nel registro
         #Circuito aggiornato
         teleportation_circuit.draw()
```

Out[19]:



Questo è l'unico cambiamento rispetto a prima, oltre all'eliminazione di due registri classici (in quanto non è necessario misurare i qubit di Alice).
Così facendo siamo riusciti a mandare in esecuzione il programma su un computer quantistico reale.

I risultati ottenuti presentano delle oscillazioni dovute alle interferenze:

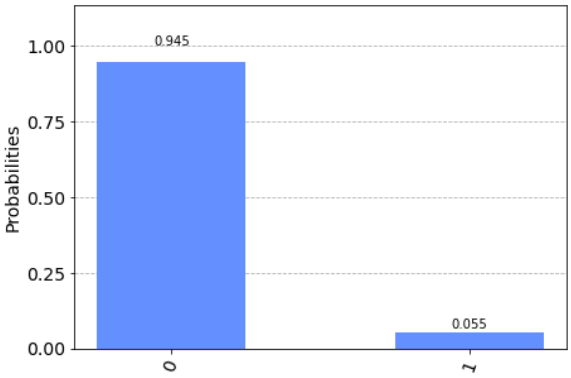


Figura 2 Inizializzato con 0.9 e 0.1

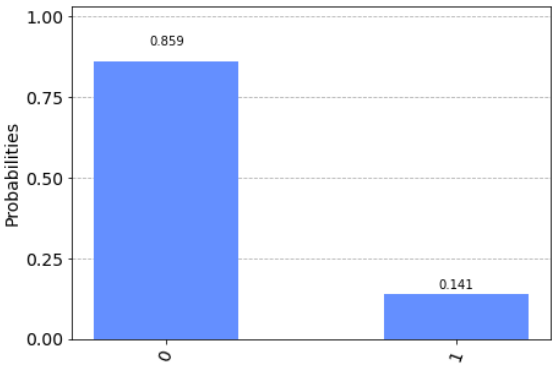


Figura 1 Inizializzato con 0.3 e 0.7

I risultati non risultano influenzati in modo importante dal cambio di probabilità del qubit da teletrasportare, ipotizziamo perché l'algoritmo non è interessato alle probabilità date al qubit di Alice, purchè esse ovviamente sommino a 1.