UNIVERSITÀ DEGLI STUDI ROMA TRE

Dipartimento di Ingegneria

Corso di Laurea in Ingegneria Informatica

Tesi Di Laurea

Sviluppo di un simulatore per la crittografia quantistica a variabili continue

Laureando

Kevin Santodonato

Matricola 548019

Relatore

Prof. Matteo Rosati

Anno Accademico 2022/2023

Questa è la dedica

Ringraziamenti

Vorrei prendere un momento per esprimere la mia gratitudine a tutte le persone che hanno contribuito al raggiungimento di questo traguardo.

Innanzitutto, desidero ringraziare il mio Relatore, il *Professor Matteo Rosati* per la usa estrema disponibilità e il suo costante sostegno.

Un ringraziamento speciale va alla mia famiglia per essermi sempre stata vicino, soprattuto nei momenti piú difficili di questo percorso di studi.

Vorrei inoltre ringraziare il mio amico e collega *Davide Tedesco* per aver provveduto a spronarmi a dare sempre il massimo.

Infine, ma non per importanza, vorrei ringraziare i colleghi e gli amici che hanno contribuito alla mia crescita personale e professionale.

Indice

Elenco delle figure

Introduzione

La crittografia, da migliaia di anni, ha costituito il fondamento delle comunicazioni segrete e della protezione dei dati sensibili. Nel corso della storia umana, dall'antico Egitto fino alla metà del secolo scorso, la crittografia veniva considerata un'arte e tutti i sistemi crittografici e cifrari che venivano realizzati erano considerati sicuri fino al momento in cui qualcuno riusciva a trovare una falla nel sistema e decifrare il messaggio senza una chiave.

Nel 1949, grazie a Claude Shannon, inizia l'era moderna della crittografia la quale, da questo punto in poi della storia, viene considerata oltre che un'arte anche una scienza. Shannon nel '49 pubblica un articolo denominato "Communication Theory of Secrecy Systems"[Sha49] nel quale comprova, per la prima volta, la sicurezza del cifrario di Vernam (comunemente chiamato One Time Pad) attatraverso dimostrazioni matematiche.

Il cifrario di Vernam si basa su un concetto molto semplice: per trasmettere una messaggio cifrato tra due parti, che vengono comunemente chiamate Alice e Bob, viene prodotta una chiave della lunghezza del messaggio che si vuole trasmettere. La chiave deve essere condivisa un modo sicuro tra le due parti, ad esempio Alice e Bob possono incontrarsi e scambiare la chiave. Successivamente la chiave viene utilizzata da Alice per codificare il messaggio e da Bob per decodificarlo, a questo punto la chiave va scartata e per trasmettere un nuovo messaggio è necessario produrre una nuova chiave.

Questo tipo di cifrario rientra nella classe dei crittosistemi simmetrici (a chiave privata) nei quali per codifica e decodifica viene utilizzata la stessa chiave. Di questa classe di crittosistemi la sicurezza è

garantita da quanto dimostrato da Shannon però si può subito notare un problema non indifferente: la chiave deve essere condivisa in qualche modo tra Alice e Bob. Condividere una chiave, in alcune situazioni, potrebbe risultare un'operazione molto dispendiosa ed è per questo che nel 1976 sono state gettate le basi per una nuova classe di crittosistemi denominati crittosistemi asimmetrici[DH76].

L'idea alla base dei crittosistemi asimmetrici (a chiave pubblica) è quella di utilizzare due chiavi diverse per la codifica e la decodifica. La prima implementazione di un protocollo che rientra in questa classe è stata realizzata nel 1978 e prende il nome di RSA [RSA78], il quale oggigiorno è ampiamente utilizzato. In crittosistemi a chiave pubblica se Bob vuole ricevere dei messaggi codificati con una chiave pubblica per prima cosa deve generare per sé stesso una chiave privata. Da questa chiave produce una chiave pubblica che rivela ad Alice la quale utilizza la chiave pubblica per codificare il messaggio e trasmetterlo a Bob. Una volta ricevuto il messaggio Bob lo decodifica utilizzando la propria chiave privata.

La sicurezza di questi sistemi, a differenza di quelli a chiave privata, non si basa su dimostrazioni matematiche ma solamente sulla complessità computazionale. Questo perché per la produzione delle chiavi vengono utilizzare delle funzioni matematiche molto semplici da calcolare ma molti difficili da invertire. In termini di complessità computazionale difficile sta a significare che il tempo necessario per eseguire un'operazione cresce esponenzialmente con la lunghezza della chiave, mentre per operazioni semplici il tempo di esecuzione è polinomiale rispetto la lunghezza della chiave. Ad esempio, la sicurezza del protocollo RSA si basa proprio sulla difficoltà di fattorizzazione di interi molto grandi.

Non essendoci prove matemetiche che garantiscano la sicurezza dei crittosistemi a chiave pubblica non è da escludere la probabilità che sia possibile realizzare un algoritmo classico che abbia una complessità computazionale polinomiale per la fattorizzazione di grandi interi. Di fatto la sicurezza di questa classe di protocolli è già minacciata da un algoritmo per computer quantistici, ideato da Peter Shor nel 1994, che permette la fattorizzazione di grandi interi con complessità polinomiale [Sho94].

Per questo al giorno d'oggi è sempre più importante la ricerca su protocolli di distribuzione quantistica di chiavi (QKD) che utilizzano principi della fisica quantistica con lo scopo di condividere chiavi su canali pubblici. Questi protocolli rientrano nella classe di crittosistemi simmetrici perché la codifica e la decodifica dei messaggi trasmessi viene effettuata con una chiave privata comune ad Alice e Bob. Il potere della QKD risiede nel fatto che grazie alla fisica quantistica si ha la possibilità di determinare se la condivisione dei messaggi necessari per la produzione della chiave è avvenuta in modo sicuro o meno, in parole povere si è in grado di determinare se una spia (Eve) è venuta in possesso di questi messaggi.

Il primo protocollo ad essere stato proposto per la distribuzione quantistica di chiavi è il BB84 che prende il nome dai suoi ideatori Charles H. Bennett e Gilles Brassard che lo hanno realizzato nel 1984 [BB14]. Questo protocollo per realizzare la comunicazione utilizza due canali di trasmissione, uno classico e pubblico e l'altro quantistico e privato. Alice trasmette sul canale quantistico a Bob una sequenza di fotoni polarizzati in modo casuale in due basi, ognuna delle quali ha due gradi di polarizzazione. In ricezione Bob effettuerà la misura della polarizzazione del fotone utilizzando una base disponibile. In trasmissione potrebbe succedere che Eve tenti a sua volta di misurare la polarizzazione del fotone alterandola e questo porterà Bob a ricevere una fotone polarizzato in modo differente rispetto a quello inviato da Alice. Questa alterazione da parte di Eve porterà Alice e Bob a determinare la presenza di una spia durante una fase di confronto successiva alla trasmissione [Zha18].

In questa tesi studieremo un protocollo di QKD che utilizza una codifica non nei singoli fotoni ma piuttosto nella fase di un segnale quantistico. La tesi è strutturata nel seguente modo: nel capitolo 1 introdurremo gli stati coerenti con le loro proprietà e la misura di segnali quantistici, nel capitolo 2 discuteremo dei protocolli QKD ed in particolare del protocollo CV-QKD con modulazione gaussiana, nel capitolo 3 illustreremo una possibile implementazione del protocollo discusso nel capitolo precedente ed infine, nel capitolo 4 discuteremo le conclusioni.

Capitolo 1

Descrizione quantistica di un segnale ottico

1.1 Stati coerenti e le loro proprietà

I segnali elettromagnetici possono essere descritti sia attraverso onde che attraverso particelle, ovvero fotoni. Ad esempio, il segnale prodotto da un LASER è ben rappresentato da uno stato quantistico detto stato coerente. Esso comunemente è rappresentato $|\alpha_j\rangle = |q_j + ip_j\rangle$ attraverso la notazione introdotta da Paul Dirac che prende il nome di *bra-ket*. Dal punto di vista della fisica classica, il numero complesso α è strettamente legato all'ampiezza di oscillazione del campo elettrico corrispondente al segnale. Esso è costituito da due componenti reali, q e p, dette componenti di quadratura. Dal punto di vista della fisica quantistica, tuttavia, non è possibile assegnare un valore preciso a queste grandezze. Infatti, i valori di q e p che utilizziamo per rappresentare lo stato coerente sono solo valori medi. Quando proviamo a fare una misurazione delle grandezze fisiche associate, che indichiamo con $\hat{q}$ e $\hat{p}$, esse si comportano come variabili aleatorie che rispondono alla stessa distribuzione di probabilità gaussiana che, in oppurtune unità di misura, ha varianza unitaria. Il motivo per cui non rappresenta-

no un valore esatto è perché, a differenza della fisica classica, nella fisica quantistica le quantitá, anche scalari, sono degli operatori che assumono una valore esatto solamente in fase di misura.

Ogni stato coerente presenta un numero medio di fotoni che può essere calcolato nel seguente modo:

$$\langle n_j \rangle = |a_j|^2 = q_j^2 + p_j^2 \tag{1.1}$$

Il numero medio di fotoni di uno stato coerente è associato all'energia del segnale di luce che viene inviato, maggiore è il numero di fotoni maggiore è l'energia del segnale.

Come detto precedentemente, gli stati coerenti si presentano come una distribuzione di probabilità gaussiana e possono essere rappresentati graficamente in un piano cartesiano, che riporta i valori misurabili di q e p, come una nuvola di punti i quali avranno appunto probabilità gaussiana di essere estratti.

1.2 Misurazione dei segnali e incertezza quantistica

Partendo dal caso ideale, in cui il canale di comunicazione è perfetto e quindi non aggiunge rumore durante la trasmissione, Bob riceverà uno stato coerente con la forma vista in figura 1.1 e proverà a misurarlo, con una data incertezza. Da notare che l'incertezza della misura è presente anche nel caso di un canale di comunicazione ideale, questo perché non è dovuta solamente al rumore esterno, come ci si aspetterebbe in un canale di comunicazione classico, ma è una caratteristica intriseca di uno stato quantistico la quale non può essere rimossa in base al principio di indeterminazione di Heisenberg. Questa incertezza si presenta proprio in fase di misura, perché come detto nella sezione 1.1, il valore di quadratura q e p sono solamente valori medi mentre il valore Q o P che otteniamo da una misura 2.1.1 si comportano come variabili aleatorie con distribuzione di probabilità gaussiana.

In un trasmissione reale, l'aggiunta di rumore esterno è inevitabile e in questo rumore è presente anche quello prodotto da un eventuale spia (Eve) che tenta anch'essa di misurare il segnale modifi-

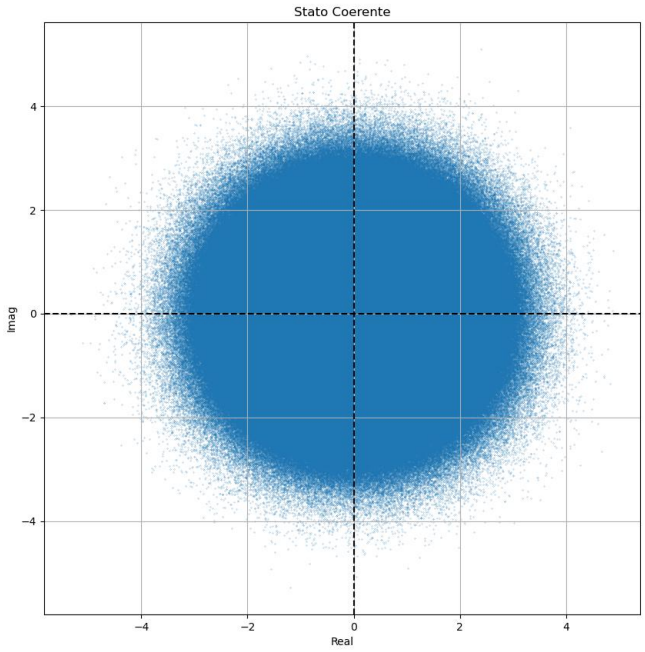


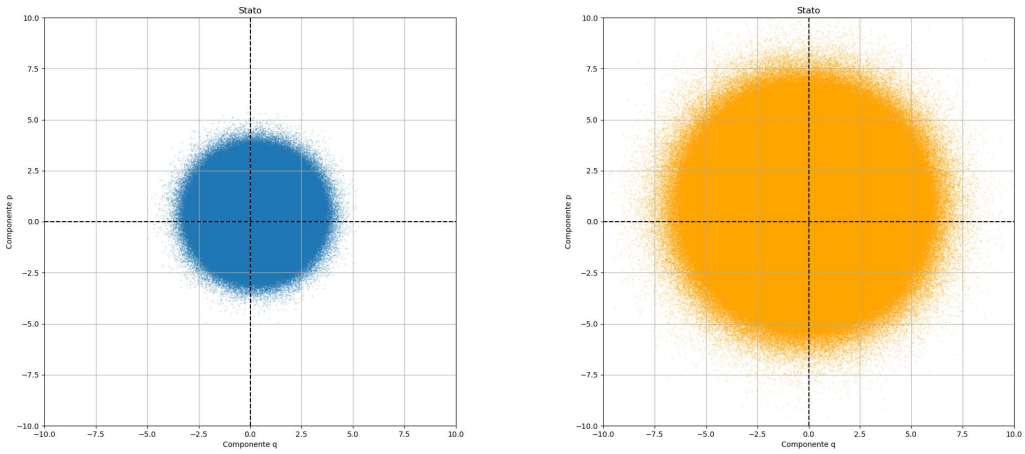
Figura 1.1: Questa figura è una gaussiana vista dall'alto che viene ottenuta simulando molte volte la produzione dello stesso stato coerente e la misura della grandezza $\hat{q}$ o $\hat{p}$. La gaussiana è rappresentata su un piano cartesiano riportante sulle ascisse le misure della componente q mentre sulle ordinate le misure della componente p

candolo, questo rende la misura di Bob ancora più incerta come possiamo notare nella figura 1.2.

Per concludere possiamo dire che i protocolli di QKD utlizzano due fondamentali caratteristiche dei segnali quantistici:

- vi è un'incertezza nella misura anche in assenza di rumore esterno;
- una misura di un segnale introduce rumore nel segnale stesso.

Tali proprietà risultano cruciali per determinare l'eventuale intromissione di Eve nella comunicazione e per nascondere i bit trasmessi da Eve.



(a) (b)

Figura 1.2: Esempio di rappresentazione dello stesso stato: (a) lo stato non è stato alterato da Eve; (b) Eve ha misurato lo stato per poi re-inviarlo a Bob

Capitolo 2

Breve storia della QKD

La QKD (Quantum Key distribution), o Distribuzione Quantistica delle Chiavi, è una tecnologia di crittografia quantistica che permette la distribuzione di chiavi crittografiche con un livello di sicurezza inattacabile anche da computer quantistici. La QKD può essere suddivisa in due macro-classi DV-QKD e CV-QKD. Nella DV-QKD (Discrete-Variable Quantum Key Distribution) le informazioni vengono trasmesse utilizzando stati quantistici discreti, ad esempio la polarizzazione dei fotoni, che viene misurata attraverso apposite apparecchiature per ogni fotone.

Il primo esempio di protocollo DV-QKD, come accennato nell'introduzione, è il BB84. In questo protocollo Alice trasmette sul canale quantistico a Bob una sequenza di fotoni polarizzati in modo casuale in due basi, ognuna delle quali ha due gradi di polarizzazione. Nella prima base il fotone può essere polarizzato a 0 o 90 gradi e verrà considerato come un bit 0, mentre nella seconda a 45 o 135 gradi e verrà considerato come bit 1. In ricezione Bob effettuerà la misura della polarizzazione del fotone scegliendo una base con probabilità uniforme dalle due basi disponibili. Alla fine della trasmissione Bob sarà in possesso dei gradi di polarizzazione misurati i quali verranno convertiti in 0 e 1 seguendo la stessa convenzione di Alice. Da notare che parte dei dati di Bob differirà da quelli inviati da Alice perchè Bob ha effettuato la misura utilizzando la base sbagliata. Successivamente Bob comunicherà ad

Alice quale base ha utilizzato per la misura di ogni fotone ed Alice scarterà tutti quei fotoni la cui base di trasmissione e misura non combaciano, e lo comunicherà a Bob. Fatto ciò Alice e Bob saranno in possesso, almeno in teoria, della stessa stringa random di bit. Quello che potrebbe succedere è che durante la trasmissione Eve intercetti i fotoni inviati da Alice per effettuarne una misura e poi inviare il risultato della propria misura a Bob. Tuttavia questo altererebbe il risultato della misura di Bob perchè Eve non può sapere con quale base è stato polarizzato il fotone, e di conseguenza non può sapere con quale base misurarlo. Misurando un fotone con la base sbagliata nel momento in cui lo ritrasmetterà a Bob, egli riceverà un fotone polarizzato in modo differente rispetto a quello inviato da Alice. Questo permetterà ad Alice e Bob di determinare la presenza di una spia andando a confrontare una porzione di bit misurati con la stessa base utilizzata in trasmissione e quindi ritenuti uguali. Durante il confronto se si riscontrano bit diversi si determina la presenza di una spia. Se Eve ha intercettatto troppi fotoni non viene prodotta nessuna chiave e si ricomincia il protocollo da capo, in caso contrario si utilizzano i bit rimasti segreti come chiave crittografica[Zha18].

Nella CV-QKD (Continous-Variable Quantum Key Distribution), a differenza della DV-QKD per la trasmissione di informazioni vengono utilizzate variabili continue delle onde elettromagnatiche come ad esempio la fase.

In entrambe le classi di protocolli la sicurezza é garantita dai princípi delle fisica quantistica che ci permettono di rilevare eventuali intercettazioni da parte di Eve (spia) nella trasmissione su canale quantistico tra Alice (mittente) e Bob (destinatario).

La rilevazione è dovuta al fatto che durante la misura da parte di Eve lo stato quantistico viene alterato facendo diminuire l'energia del segnale se questo non viene amplificato [1] e introducendo del rumore, quindi dell'incertezza. All'atto pratico Eve intercetta lo stato coerente, rappresentato come in figura 1.1, ne effettua una misura per poi ritrasmetterlo sul canale. Bob riceverà uno stato coerente con del rumore aggiunto e un minor numero di fotoni, ciò comporta che la gaussiana che lo rappre-

[1] Un'amplificazione, pur riportando il livello di energia del segnale a quello iniziale, introdurrà ulteriore rumore.

senta sarà caratterizzata da una varianza superiore a quella stimata e un valor medio più vicino allo zero. Per questo motivo, si effettuano numerosi run di trasmissione quantistica. In una fase preliminare si utilizza un certo numero di run per stimare i parametri di rumore del canale e, da essi, decidere se la trasmissione sta avvenendo in modo sicuro oppure no (si veda Sez. 2.1.2). Nel caso in cui il controllo non vada a buon fine, la trasmissione corrente viene abortita perché non è possibile garantire la sicurezza della chiave crittografica e si comincia con un nuovo round, se invece il controllo va a buon fine, il protocollo assicura che un'eventuale spia Eve non possa ottenere sufficiente informazioni sulla chiave condivisa fra Alice e Bob.

2.1 CV-QKD con modulazione gaussiana

Il protocollo di distribuzione quantistica di chiavi a variabili continue con modulazione gaussiana è un protocollo molto indicato per lo sviluppo e l'utilizzo su larga scala nel mondo reale data la sua affinità con le infrastrutture oggi già esistenti, come ad esempio, i canali di comunicazione in fibra ottica. Come enunciato all'inizio del capitolo con questo protocollo si effettuano misure su variabili continue delle onde elettromagnetiche e in particolare nei protocolli con modulazione gaussiana gli stati coerenti da trasmettere sul canale di comunicazione vengono estratti da una distribuzione normale centrata in zero e una certa deviazione standard che viene definita in base ad alcuni fattori.

Il protocollo può essere suddiviso in due sezioni: la prima sezione ha effettivamente a che fare con segnali quantistici mentre la seconda sezione opera con segnali e dati classici. La prima sezione comprende la scelta e la trasmissione degli stati coerenti su fibra ottica da parte di Alice e termina nel momento in cui Bob effettua la misura; la seconda sezione comprende il sifting, la stima dei parametri e la riconciliazione.

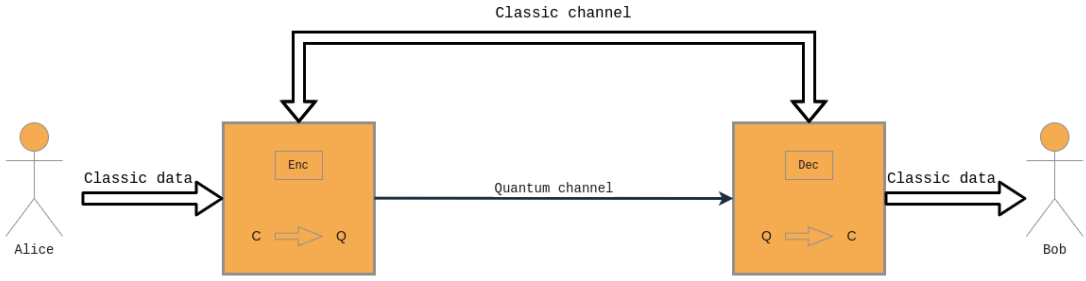


Figura 2.1: Schema di comunicazione tra Alice e Bob Sezione.

2.1.1 Produzione, trasmissione e ricezione del segnale quantistico

Alice prepara gli stati coerenti andando a estrarre i valori delle componenti di quadratura q e p da una distribuzione normale centrata in zero ($q \sim \mathcal{N}(0, V_{mod})$ e $p \sim \mathcal{N}(0, V_{mod})$) [LPF+18]. Da notare che la distribuzione normale dalla quale Alice estrae le componenti di quadratura non ha nulla a che fare con le distribuzioni normali che caratterizzano gli stati coerenti ma rappresenta il modo in cui i dati vengono modulati. Dopo la preparazione Alice trasmette a Bob gli stati coerenti attraverso un canale di comunicazione, ad esempio una fibra ottica, che può essere anch'esso rappresentato come un canale quantistico.

Il canale è caratterizzato da un certo valore di trasmittanza T e di rumore ξ. A causa della trasmittanza durante la trasmissione il segnale perde potenza quindi le gaussiane che descrivono lo stato coerente saranno centrate in un valore più vicino allo zero rispetto al momento della preparazione, mentre il rumore che viene introdotto fa aumentare la varianza delle gaussiane così da accrescere l'incertezza nelle misure da parte di Bob.

Per dare una rappresentazione matematica di quello che avviene ad uno stato coerente durante la trasmissione, possiamo andare a cosiderare un generico stato scelto da Alice $|\alpha_A\rangle = |q_A + ip_A\rangle$. Come abbiamo detto nella sezione 1.1 le componenti di quadratura rispondono ad una distribuzione di probabilià normale. Bob andrà ad effettuare una misura su uno stato non più coerente, a causa

della presenza di rumore ambientale. Tuttavia, lo stato quantistico ricevuto da Bob può ancora essere rappresentato utilizzando i valori medi di $\hat{q}$ e $\hat{p}$ e le loro varianze. Per lo stato ricevuto da Bob queste quantità prendono i seguenti valori:

$$\begin{aligned} q_B = Mean(\hat{q}_B) = \sqrt{T} q_A, \quad Var(\hat{q}_B) = 1 + \xi \\ p_B = Mean(\hat{p}_B) = \sqrt{T} p_A, \quad Var(\hat{p}_B) = 1 + \xi \end{aligned} \tag{2.1}$$

Pertanto, a tutti gli effetti, il canale ha come risultato quello di attenuare i valori medi e aumentare la varianza delle componenti di quadratura.

Per la misura dello stato possono essere adottati due metodi diversi:

- **misura omodina**: Bob decide se misurare la componente di quadratura q o p; la decisione viene presa attraverso una distribuzione di probabilità uniforme. Per ciascuna componente di quadratura i valori misurati seguono le seguenti distribuzioni gaussiane:

$$\begin{aligned} Q_B \sim \mathcal{N}(\sqrt{T} q_A, 1 + \xi) \\ P_B \sim \mathcal{N}(\sqrt{T} p_A, 1 + \xi) \end{aligned} \tag{2.2}$$

  Per rimarcare quanto appenda detto, solo una delle due componenti verrà misurata.

- **misura eterodina**: questa misura permette di misurare entrambe le componenti contemporaneamente, però c'è un prezzo da pagare perché effettuando questa misura viene aggiunto del rumore e la varianza delle distribuzioni normali ha un incremento di 1. Anche i questo caso i valori misurati di entrambe le componenti seguono le seguenti distribuzioni di probabilità gaussiane:

$$\begin{aligned} Q_B \sim \mathcal{N}(\sqrt{T} q_A, 2 + \xi) \\ P_B \sim \mathcal{N}(\sqrt{T} p_A, 2 + \xi) \end{aligned} \tag{2.3}$$

In caso di misura omodina è necessaria un'operazione addizionale cioé il sifting. Infatti, a questo punto del protocollo Alice è in posseso del doppio dei dati di Bob (in quanto Bob misura solo

una della due componenti per ogni round) e per far sì che abbiano la stessa quantità di dati correlati Bob rivela ad Alice, su canale classico pubblico, quale componente ha misurato ad ogni round; Alice prende nota della componenti misurate e scarta le altre[MFZG18]. Notare che questa operazione non compromette la sicurezza: Eve saprà solo quale tipo di misura ha eseguito Bob ad ogni round, ma non il risultato ottenuto né il valore vero inviato da Alice.

2.1.2 Stima dei parametri

Dopo avere terminato la fase di trasmissione, Alice e Bob rivelano una porzione random di dati in modo tale da confrontare ciò che è stato effettivamente inviato da Alice con le misure di Bob. Da questo confronto sono in grado di stimare l'informazione mutua I_{AB} (informazione mutua tra Alice e Bob) e l'informazione mutua tra Eve e Bob χ_{EB}. Prima della effettiva stima delle mutue informazioni e necessario stimare trasmittanza e rumore del canale e calcolare altre quantità che verranno utilizzare nel calcolo delle mutue informazioni.

Nel'articolo [LPF+18] viene calcolata la segiente espressione per l'informazione mutua I_{AB} è:

$$I_{AB} = \frac{\mu}{2}\log_2(1 + SNR) \tag{2.4}$$

con $\mu = 1$ se la misura è omodina e $\mu = 2$ per la misura eterodina. Qui la SNR è la signal-to-noise ratio, data da::

$$SNR = \frac{\frac{1}{\mu}TV_{mod}}{1 + \frac{1}{\mu}\xi} \tag{2.5}$$

Per la stima di rumore e trasmittanza del canale abbiamo bisogno dei parametri a, b e c che rappresentano le entry della matrice di covarianza Σ_{AB}. Tale matrice rappresenta la covarianza fra le quadrature $\hat{q}$ e $\hat{p}$ per lo stato quantistico e deve essere stimata a partire dai risultati delle misure di Bob.

$$\Sigma_{AB} = \begin{pmatrix} a\mathbb{1}_2 & c\mathbb{1}_2 \\ c\mathbb{1}_2 & b\mathbb{1}_2 \end{pmatrix} \tag{2.6}$$

Le equazioni per il calcolo dei parametri vengono ancora una volta prese dai risultati dell'articolo [LPF+18]:

$$\begin{aligned} a &= V_{mod} + 1 \\ b &= \frac{\langle q_B^2 \rangle + \langle p_B^2 \rangle}{2} \\ c &= \sqrt{\frac{V_{mod} + 2}{V_{mod}}} \cdot \left(\frac{\langle q_A q_B \rangle + \langle p_A p_B \rangle}{2} \right) \end{aligned} \tag{2.7}$$

Calcolati i parametri a, b e c possiamo andare a stimare il rumore e la trasmittanza:

$$\begin{aligned} T &= \frac{c^2}{V_{mod}^2 + 2V_{mod}} \\ \xi &= b - TV_{mod} - 1 \end{aligned} \tag{2.8}$$

Ora non ci resta che calcolare χ_{EB} in questo modo:

$$\chi_{EB} = g(\nu_1) + g(\nu_2) - g(\nu_3) \tag{2.9}$$

Per fare questo avremo bisogno dei valori delle quantità ν_1, ν_2 e ν_3 che possiamo calcolare come segue:

$$\begin{aligned} \nu_1 &= \frac{1}{2}[z + (b - a)] \\ \nu_2 &= \frac{1}{2}[z - (b - a)] \\ \nu_3 &= \sqrt{a(a - \frac{c^2}{b})} \end{aligned} \tag{2.10}$$

con $z = \sqrt{(a+b)^2 - 4c^2}$.

La funzione $g(\nu)$ per il calcolo di della mutua informazione fra Eve e Bob è definita in questo modo:

$$g(\nu) = \left(\frac{\nu + 1}{2} \right) \log_2 \left(\frac{\nu + 1}{2} \right) - \left(\frac{\nu - 1}{2} \right) \log_2 \left(\frac{\nu - 1}{2} \right) \tag{2.11}$$

Infine, dopo aver calcolato l'informazione mutua tra Alice e Bob e quella tra Eve e Bob vengono confrontati i risultati e nel caso in cui χ_{EB} risulti essere maggiore di βI_{AB} [2] il protocollo viene abortito.

[2] β rappresenta l'accuratezza nel processare i dati dopo la trasmissione e viene calcolata come il rapporto tra il code-rate del codice classico scelto e la mutua informazione tra Alice e Bob

2.1.3 Riconciliazione delle informazioni

Se la stima dei parametri ha avuto successo Alice e Bob sono in possesso di due stringhe di numeri reali diversi ma correlati, a causa degli errori introdotti durante la trasmissione sul canale e la misurazione di Bob. Devono pertanto effettuare una riconciliazione fra queste due stringhe, per assicurarsi di ottenere una sola stringa condivisa da entrambi. La riconciliazione può avvenire in due forme:

- **diretta**: Alice produce un messaggio fittizio codificando i suoi dati (correlati a quelli di Bob) con una chiave. Bob basandosi sulle misure effettuate in fase di trasmissione cerca di estrarre la chiave dal messaggio fittizio;
- **inversa**: la procedura è la stessa della riconciliazione diretta solamente che vengono scambiate le parti, Bob produce il messaggio fittizio ed Alice cerca di estrarre la chiave.

La riconciliazione diretta presenta un problema: non può essere utilizzata per valori di trasmittanza troppo bassi, questo perché Eve avrebbe piú informazione sui dati di Alice rispetto a Bob e quindi risulterebbe impossibile creare una chiave crittografica sicura. D'altro canto la riconciliazione inversa non presenta questo problema quindi è possibile utilizzarla anche con valori bassi di trasmittanza, però è da tenere in considerazione che più bassa è la trasmittanza più sarà distruttivo l'effetto del rumore del canale [LPF+18].

A questo punto del protocollo Alice e Bob sono in possesso del subset di dati non utilizzati per la stima dei parametri che andiamo a chiamare $\mathbf{X}_0$ e $\mathbf{Y}_0$; prima di effettuare la riconciliazione vengono prodotte, a partire dai dati in loro possesso, altre due sequenze di dati correlati.

Quello che ci aspettiamo è che le varianze dei dati in possesso di Alice e Bob siano rispettivamente:

$$\begin{aligned} V_{X_0} &= V_{mod} + 1 \\ V_{Y_0} &= TV_{mod} + 1 + \xi \end{aligned} \tag{2.12}$$

Detto questo possiamo normalizzare $\mathbf{X}_0$ e $\mathbf{Y}_0$ in questo modo:

$$\begin{aligned} \mathbf{X} &= \frac{\mathbf{X}_0}{\sqrt{V_{mod}}} \\ \mathbf{Y} &= \frac{\mathbf{Y}_0}{\sqrt{TV_{mod} + 1 + \xi}} \end{aligned} \tag{2.13}$$

Così facendo abbiamo ottenuto altre due sequenze di dati $\mathbf{X}$ e $\mathbf{Y}$, correlate tra loro dall'equazione $\mathbf{X} = \mathbf{Y}+\mathbf{Z}$, dove $\mathbf{Z}$ è una variabile aleatoria che risponde ad una distribuzione normale centrata in zero e con varianza $V_z = \frac{1}{V_{mod}}$. Quindi normalizzando in questo modo abbiamo ottenuto lo stesso effetto che avremmo avuto se Bob avesso inviato ad Alice i proprio dati normalizzati $\mathbf{Y}$ su un canale classico e pubblico che aggiunge solamente rumore gaussiano[MFZG18]. Questo è il prototipo di canale di comunicazione classico utilizzato sin dai tempi di Shannon, detto canale di rumore bianco gaussiano additivo - Additive White Gaussian Noise Channel (AWGN). Per la trasmissione dati sull'AWGN è quindi possibile sfruttare una vasta gamma di codici studiati a partire dagli anni '60 [Gal72]-[RU08].

Il passo successivo per Bob sarà quello di generare una stringa di bit random $\mathbf{s} \in \{0,1\}^k$. Attraverso algoritmi di codifica LDPC (Low-Density Parity-Check) [BS] viene calcolata una matrice $\mathbf{H}$ necessaria per aggiungere alla stringa $\mathbf{s}$ dei bit ridondanti di parità ottenendo la stringa $\mathbf{c} \in \{0,1\}^n$ dove n corrisponde alla lunghezza della sottosequenza di dati non utilizzati per la stima dei parametri.

Dopodiché vengono utilizzati dei codici LDPC [BS] che rappresentano una classe di codici correttori di errori utilizzari nell'ambito delle telecomunicazioni. Una delle ragioni per cui questi codici sono largamente utilizzati è la loro abilità di raggiungere prestazione vicine alla capacità teorica del canale di comunicazione, nota come il limite di Shannon. I codici LDPC possono essere rappresentati in due modi:

- come una matrice di dimensioni $M \times N$ dove N rappresenta la dimensione del messaggio che si vuole codificare ed $M = N - k$, con k ottenuto dal prodotto tra N e il code-rate, corrisponde al numero di bit ridondanti nel messaggio codificato, che vengono utilizzati per

la correzione degli errori. Questa matrice è prevalentemente composta da zeri e contiene un numero limitato di uno.

- come una grafo bipartito rappresentato da nodi check e nodi variabile; facendo riferimento alla rappresentazione matriciale il numero di nodi check è M (numero di righe della matrice) e quello di nodi variabile è N (numero di colonne della matrice). Gli uno contenuti nella matrice, nel grafo, sono denotati dagli archi che collegano un nodo variabile ad un nodo check.

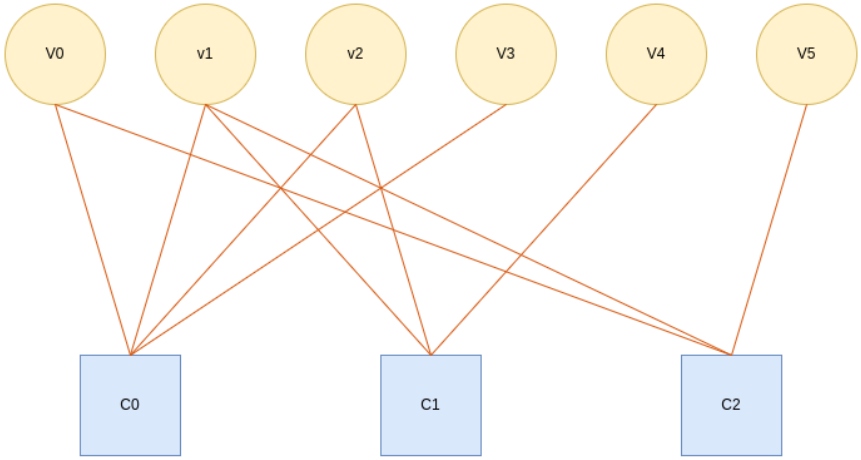


Figura 2.2: Grafo bipartito che rappresenta la matrice di parità, in questo particolara caso fa riferimento alla matrice H 2.14 sottostante.

Attraverso gli algoritmi LDPC viene calcolata una matrice $\mathbf{H}$ che sarà della forma $\left[-P^{T}_{(M\times k)}|I_{N-k}\right]$:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.14}$$

Esempio di una matrice con $M = 3$ ed $N = 6$.

Dalla matrice **H** possiamo ottenere la matrice generatore $\mathbf{G} = \left[I_k | P_{(k\times M)}\right]$ da una matrice identità di dimensione k e la trasposta della matrice $P^T_{(M\times k)}$:

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \tag{2.15}$$

Successivamente tramite il prodotto matriciale $\mathbf{s} \times \mathbf{G}$ tra la stringa **s** e la matrice generatore **G** si ottiene la stringa **c** che verrà utilizzata per modulare il segno della sequenza **Y** ottenedo così un nuovo messaggio **m** tale che $m_i = Y_i(-1)^{c_i}$ per $i = 1, 2, \ldots, n$.

Il messaggio **m** viene trasmesso ad Alice su canale classico publico che assumiamo non introduca errori. Alice estrae da **m** un messaggio fittizio **r** che le permettarà, attraverso l'utilizzo dell'algoritmo *sum-product* 3, di ottenere una stima della stringa di bit **c** prodotta originariamente da Bob.

2.2 Discussione della sicurezza contro attacchi di lettura del segnale

Questo protocollo è particolarmente sicuro perché consente il rilevamento di un'eventuale spia (Eve). Nel protocollo si assume che Eve sia in possesso di un computer quantistico e che abbia accesso completo al canale di comunicazione ed anche con queste assunzioni si è in grado di garantire la sicurezza per diversi tipi di attacchi.

Gli attacchi presi in considerazione tipicamente sono tre e differiscono per la loro efficacia:

- **attacchi individuali**: Eve effettua operazioni singolarmente su ogni segnale che viene scambiato tra Alice e Bob.

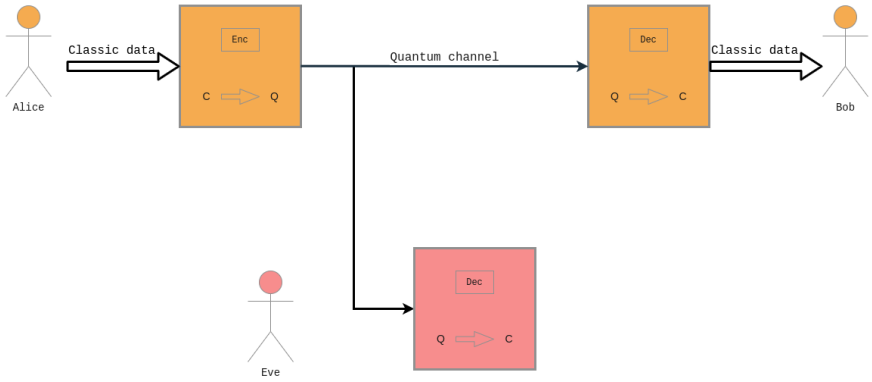


Figura 2.3: Schema che descrive un attacco individuale ponendo attenzione nel mostrare che le operazioni da parte di Eve vengono svolte singolarmente su ogni segnale.

- **attacchi collettivi**: Eve salva nella memoria del suo computer quantistico un certo numero di stati sui quali effettua una misura collettiva. Questo attacco è più potente del precedente perchè secondo i pricipi della fisica quantistica effettuando una misura collettiva si può potenzialmente ricavare più informazione che dalla misura del singolo stato.

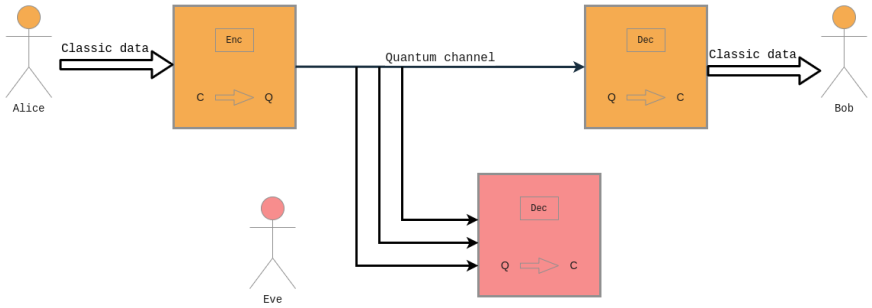


Figura 2.4: Schema che descrive un attatto collettivo mostrando che Eve effettua operazione su più segnali contemporaneamente.

- **attacchi coerenti**: sono attacchi più potenti dei collettivi perché Eve, oltre alla lettura collettiva, può utilizzare delle correlazioni quantistiche, cioé entanglement, per mettere in relazione gli stati di più run.

Tornando al motivo per cui effettivamente si riesce a rilevare Eve possiamo, anche questa volta, dare il merito ai principi della fisica quantistica. Questo perchè in caso di misura da parte di Eve lo stato coerente trasmesso verrà alterato. Due esempi di attacchi che Eve può mettere in atto e le loro conseguenze sono:

- sottrarre una parte del segnale per effettuare una misura sullo stesso, questo comporta una diminuzione di fotoni del segnale ricevuto da Bob
- misura il segnale attraverso una misura eterodina per stimare entrambe le componenti di quadratura. Utilizzando le componenti misurate come valori medi, realizza un nuovo stato coerente che trasmette a Bob. In questo modo poiché vengono misurati dei valori randomici e la misura eterodina stessa introduce ulteriore rumore, la varianza degli stati coerenti ricevuti da Bob sarà superiore a quella originaria.

In entrambi i casi, un attacco da parte di Eve non va a far altro che aumentare gli effetti distruttivi del canale di trasmissione.

Tutte queste alterazioni, in fase di stima dei parametri, produrranno dei valori indesiderati come $\chi_{EB} > I_{AB}$. Questa incongruenza tra le stime e ciò che ci si aspettava dalla scelta dei paramettri iniziali portano a concludere che una spia si sia intromessa nella comunicazione.

Capitolo 3

Simulazione

La realizzazione di questo simulatore per la distribuzione quantistica di chiavi con modulazione gaussiana si prefigge lo scopo di realizzare una chiave crittografica sicura condivisa tra due parti, la quale successivamente potrà essere utilizzate per la codifica e decodifica di un messaggio trasmesso su un canale pubblico classico.

Per lo sviluppo del simularore è stato utilizzato il linguaggio di programmazione C++. I motivi principali di questa scelta sono tre: la velocità del linguaggio di programmazione, la disponidilità di librerie che implementano le distribuzioni di probabilità necessarie alla simulazione e la presenza di librerie esterne che implementano due algoritmi indispensabili per la correzione degli errori.

Anche se il progetto è stato realizzato utilizzando C++ è stato deciso di illustrare quanto è stato fatto attraverso dello psdeudo-codice in modo tale da non legare la simulazione ad una specifico linguaggio di programmazione ed agevolarne la re-mplementazione in qualunque linguaggio di programmazione si voglia utilizzare.

Prima di addentranci nella descrizione delle fasi della simulazione, è bene andare a definire delle costanti necessarie per il corretto funzionamente del simulatore. Le costanti in questione sono il rumore e la trasmittanza del canale, rispettivamente $\xi = 0.005$ e $T = 0.1$, la varianza $V_{mod} = 5.226$ della

modulazione ed il code-rate $R = 0.3$. I valori per queste costanti sono stati scelti in base ai risultati ottenuti negli articoli [LPF+18] e [FNZG22].

3.1 Preparazione, trasmissione e misura

3.1.1 Preparazione dello stato coerente

Per la realizzazione degli stati coerenti da trasmettere sul canale vengono scelti due valori che rappresentano i valori medi delle due gaussiane con varianza unitaria che caratterizzano lo stato. I valori vengono estratti a loro volta da una distribuzione di probabilità normale centratata in zero e con varianza V_{mod}. Queste informazioni sono salvate in una struttura dati che, oltre a contenere tutte le informazioni necessarie per caratterizzare lo stato coerente, presenta dei tag che verranno utilizzati nella sottosezione **??** per distinguere le componenti di quadratura p e q.

3.1.2 Trasmissione su canale rumoroso

Per ogni stato coerente prodotto si simula la trasmissione in un canale rumoroso. Le operazioni che vengono effettuate in pratica sono: il prodotto dei valori medi della gaussiane per la radice quadrata della trasmittanza e la somma di ξ alla varianza. Quindi le le componenti di quadratura saranno descritte come nell'espressione 2.1.

3.1.3 Misura omodina del segnale

Per la misurazione del segnale verrà effettuata una misura omodina che consiste in due fasi. Per prima cosa si decide quale delle componenti di quadratura misurare: la decisione viene presa attraverso un distribuzione di probabilità uniforme. Questo significa che le componenti q e p hanno la stessa probabilità di essere misurate.

Successivamente avviene l'effetiva misura che consiste nell'estrarre un valore da una delle due distribuzione normali:

$$\begin{aligned} Q_B &\sim \mathcal{N}(\sqrt{T}q_B, 1+\xi) \\ P_B &\sim \mathcal{N}(\sqrt{T}p_B, 1+\xi) \end{aligned} \tag{3.1}$$

3.2 Sifting

Prima di procede con l'effettiva stima dei parametri è necessario effettua il sitfing (vaglio delle informazioni). A questo punto delle simulazione si hanno gli stati coerenti inviati e l'insieme delle misure delle componenti. Dal momento che viene effettuata la misura di una sola componente per ogni round di trasmissione, i dati trasmessi sono il doppio di quelli misurati. L'operazione che viene effettuata è molto semplice: per ogni misura effettuata viene determinata quale componente è stata misurata, questo grazie al tag di cui abbiamo parlato nella sottosezione 3.1.1; per il round di trasmissione corrispondente vengono mantenute solamente le informazione di valore medio e tag relative alla componente misurata scartando di resto.

Questa operazione viene effettata per ogni coppia di dati trasmessi e misurati.

3.3 Stima dei parametri

La stima dei parametri è una fase indispensabile del protocollo perché permette di determinare se la trasmissione è stata sicura oppure è avvenuta l'intromissione di una spia. Per la stima dei parametri Alice e Bob dovrebbero scambiarsi un subset random dei dati trasmissi ma per semplicità nella simulazione utilizziamo i primi n dati trasmessi. In un'applicazione reale del protocollo questa è una pessima scelta, perché se Eve venisse in possesso di questa informazione potrebbe non intercettare i primi n segnali andando a misurare solo la restante parte. Così facendo Eve non verrebbe rilevata pur

avendo intercettato la parte piú importante della comunicazione, cioè i dati che verranno utilizzati per distillare la chiave.

Chiamiamo i subset di dati dopo il sifting di Alice e Bob utilizzati per la stima dei parametri Q_A e Q_B per la componente di quadrature q e P_A e P_B per la componente di quadratura p.

Svolte tutte le operazioni illustrate dall'algoritmo 1 per ottenere la stima dell'informazione mutua tra Alice e Bob I_{AB} e di quella tra Eve e Bob χ_{EB} le andiamo a confrontare. Nel caso in cui I_{AB} risulti minore di χ_{EB} la simulazione si interrompe e deve ricominciare da capo.

Algorithm 1 : Stima dei parametri

1: **Step 1:** Stima dei parametri a, b e c
   $a \leftarrow V_{mod} + 1$
   $b \leftarrow \frac{\langle Q_B^2 \rangle + \langle P_B^2 \rangle}{2}$
   $c \leftarrow \sqrt{\frac{V_{mod}+2}{V_{mod}}} \cdot \frac{\langle Q_B Q_A \rangle + \langle P_B P_A \rangle}{2}$
2: **Step 2:** Stima del rumore e della trasmittanza del canale
   $T \leftarrow \frac{c^2}{V_{mod}^2 + 2V_{mod}}$
   $\xi \leftarrow b - TV_{mod} - 1$
3: **Step 3:** Calcolo dei ν
   $\nu \leftarrow nuCalc()$ ▷ vengono utilizzare le equazioni 2.10
4: **Step 4:** Calcolo I_{AB}
   $SNR \leftarrow signalNoiseRatio()$ ▷ viene utilizzata l'equazione 2.5
   $I_{AB} \leftarrow mutualInfoAliceBob(SNR)$ ▷ viene utilizzata l'equazione 2.4
5: **Step 5:** Calcolo χ_{EB}
   $\chi_{EB} \leftarrow g(\nu_1) + g(\nu_2) - g(\nu_3)$ ▷ la funzione $g(\nu)$ 2.11
6: **Step 6:** Confronto tra I_{AB} e χ_{EB}
   $\beta \leftarrow \frac{R}{I_{AB}}$ ▷ R rappresenta in code-rate
7: **if** $\beta I_{AB} < \chi_{EB}$ **then**
8: $abort()$ ▷ la simulazione si interrompe
9: **end if**

3.4 Riconciliazione

A questo punto della simulazione si è in possesso del subset di dati non utilizzati per la stima dei parametri che andiamo a chiamare $\mathbf{X}_0$ e $\mathbf{Y}_0$; prima di effettuare la reconciliazione vengono prodotte,

a partire da questi dati, altre due sequenze di dati correlati che chiameremo **X** e **Y** come illustrato nell'equazione 2.13.

Poi viene prodotta la stringa **s** la quale viene codificata attraverso l'algoritmo PEG (Progresive Edge Growth) per ottente la stringa **c**. Utilizziamo la stringa **c** per modulare il segno della sequenza di dati **Y** di Bob per produrre il messaggio **m** che verrà inviato ad Alice. Da messaggio **m** Alice estrarrà una messaggio fittizio **r** con $r_i = \frac{m_i}{X_i}$ che corrisponde alla trasmisione su un canale Gaussiano con rumore di varianza $V_i = \frac{V_z}{|X_i|}$.

Per tutti gli elementi del messaggio **r** vengono calcolati i valori di varianza che a loro volta vengono utilizzati per calcolare le LLRs (log-likelihood ratios) che sono definite come [GET+21]:

$$l_i = \log \frac{P(R_i = r_i | C_i = 0)}{P(R_i = r_i | C_i = 1)} = \frac{2r_i}{V_i} \tag{3.2}$$

Le LLRs sono il logaritmo del rapporto tra la probabilità condizionata che il messaggio r_i provenisse effettivamente da un bit 0 e la probabilitá che il messaggio r_i provenisse da una bit 1. I valori di LLRs permettono ad Alice, grazie all'utilizzo di un algoritmo LDPC di decodifica 3, di ottenere una stima **c'** della stringa **c** prodotta da Bob.

Tutte queste operazione vengono illustrate nell'algoritmo 2 facendo riferimento alla sottosezione 2.1.3.

3.4.1 Algoritmi Sum-Product e classic PEG

L'algoritmo LDPC utilizzato per la decodifica nella simulazione prende in nome di *sum-product* 3. Questo algoritmo utilizza la tecnica di *belief propagation* per creando per i nodi check e i nodi variabile dei messaggi. Questi messaggi vengono aggiornati ad ogni iterazione dell'algoritmo e permettono, a loro volta, di aggiornale i valori delle LLRs.

Per quanto riguarda la codifica abbiamo utilizzato l'algoritmo *classic-peg*, il quale realizza la matrice di parità rappresentata come un grafo partendo dalla dimensione della matrice che si vuole ottenere e da

Algorithm 2 : Riconciliazione

Step 1: Genera string di bit random
s $\leftarrow generaStringaBit()$ ▷ **s** $\in \{0,1\}^k$
Step 2: Crea la matrice **H** e la matrice **G**
H $\leftarrow peg(M, N, \lambda)$ ▷ l'algoritmo PEG prendo come parametri il grado dei
▷ nodi variabile, il numero di righe e di colonne
G $\leftarrow generaG(\mathbf{H})$
Step 3: Genera la code-word
c $\leftarrow$ **s** $\times$ **G**
Step 4: Viene modulato il segno dei dati di Bob
m $\leftarrow modulazione(\mathbf{Y})$
Step 5: Estrazione del messaggi fittizio da parte di Alice e calcolo LLR
r $\leftarrow messaggioFittizio(\mathbf{m})$
LLR $\leftarrow calcoloLLR(\mathbf{r}, \mathbf{X})$
Step 6: Stima della code-word
c' $\leftarrow sumProd(LLR, H)$

quantità Λ che rappresentano le frazioni dei nodi variabile che avranno un certo grado. È possibile calcolare le componenti del vettore Λ in questo modo:

$$\Lambda_i = \frac{r_i}{\sum_i r_i} \tag{3.3}$$

dove r_i vengono calcolati come segue:

$$r_i = \frac{\lambda_i}{i} \tag{3.4}$$

a partire da delle quantità λ che prendono in nome di node degree distribution e rappresentano la frazione di archi collegati ad una nodo di grado d.

Tutti i valori del vettore λ, i gradi dei nodi da utilizzare e l'espressione per il calcolo del vettore Λ fanno riferimento ai risultati ottenuti nell'articolo [FNZG22] in merito al code-rate ed al valore di SNR da noi scelto.

Come abbiamo detto l'algoritmo *classic-peg* prende input la dimensione della matrice di parità e quindi il numero di nodi check e nodi variabile del grafo oltre che la frazione di nodi di un certo grado.

L'algoritmo *classic-peg* genera il grafo, partendo dai parametri in input, iterando su tutti i nodi variabile ed aggiungendo archi che collegano il nodo variabile ai nodi check. Si passa al nodo variabile successivo solamente quando tutti gli archi richiesti per quel nodo sono stati creati. Per la creazione di un nuovo arco ci sono due scelte possibili:

- il nodo variabile corrente non presenta ancora nessun arco, in questo caso viene creato un arco che lo collega al nodo check con minor numero di archi tra tutti i nodi check disponibili;
- il nodo variabile corrente presenta già altri archi, in questo caso si hanno due scenari:
  - ci sono dei nodi che non appartengono al sottografo del nodo variabile corrente. Viene creato un arco tra il nodo variabile corrente ed il nodo check con meno archi che non appartiene al sottografo;
  - il sottografo del nodo corrente copre tutti i nodi del grafo. In questo caso si sceglie il nodo check più lontano dall'attuale nodo variabile.

Algorithm 3 : Sum-product decoding algorithm

Step 1: Inizializzazione dei messaggi dei nodi variabile $L(q_{i,j})^{\langle 0 \rangle}$
for $i \leftarrow 0; M$ **do**
  for $j \leftarrow 0; N$ **do**
    $L(q_{i,j})^{\langle 0 \rangle} \leftarrow LLRs_i$
  end for
end for
for $k \leftarrow 1; maxRound$ **do**
  Step 2: Calcola messaggi di check
  for $i \leftarrow 0; N$ **do**
    $tmp \leftarrow 1$
    for $j \leftarrow 0; M$ **do**
      for each $w \in C_i$ **do**
        if $w \neq i$ **then**
          $tmp \leftarrow tmp * \tanh\left(\frac{1}{2} L(q_{w,j})^{\langle k-1 \rangle}\right)$
        end if
      end for
      $L(r_{j,i})^{\langle k \rangle} \leftarrow 2 \tanh^{-1}(tmp)$
    end for
  end for
  Step 3: Aggiorna i messaggi dei nodi variabile
  for $i \leftarrow 0; N$ **do**
    for $j \leftarrow 0; M$ **do**
      $tmp \leftarrow 0$
      for each $w \in V_i$ **do**
        if $w \neq i$ **then**
          $tmp \leftarrow tmp + L(r_{w,i})^{\langle k \rangle}$
        end if
      end for
      $L(q_{i,j})^{\langle k \rangle} \leftarrow LLRs_i + tmp$
    end for
  end for
  Step 4: Aggiorna **LLRs** aggiornate
  for $i \leftarrow 0;$ **LLRs**$.size()$ **do**
    $tmp \leftarrow 0$
    for each $j \in V_i$ **do**
      $tmp \leftarrow tmp + L(r_{i,j})^{\langle k \rangle}$
    end for
    $L(Q_i) \leftarrow LLRs_i + tmp$
  end for
end for
Step 5: Stima **C**
for $i \leftarrow 1;$ **LLRs**$.size()$ **do**
  if $LLRs_i \geq 0$ **then**
    $C_i \leftarrow 0$
  else
    $C_i \leftarrow 1$
  end if
end for

Conclusioni e sviluppi futuri

Conclusioni

Sviluppi futuri

Bibliografia

[BB14] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7–11, dec 2014.

[BS] Saumya Borwankar and Dhruv Shah. Low Density Parity Check Code (LDPC Codes) Overview.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[FNZG22] Xiaodong Fan, Quanhao Niu, Tao Zhao, and Banghong Guo. Rate-compatible ldpc codes for continuous-variable quantum key distribution in wide range of snrs regime. *Entropy*, 24(10), 2022.

[Gal72] Robert Gallager. *Information Theory and Reliable Communication*. Springer Vienna, 1972.

[GET+21] Kadir Gümüş, Tobias A Eriksson, Masahiro Takeoka, Mikio Fujiwara, Masahide Sasaki, Laurent Schmalen, and Alex Alvarado. A novel error correction protocol for continuous variable quantum key distribution. *Scientific reports*, 11(1):10465, 2021.

[LPF+18] Fabian Laudenbach, Christoph Pacher, Chi-Hang Fred Fung, Andreas Poppe, Momtchil Peev, Bernhard Schrenk, Michael Hentschel, Philip Walther, and Hannes Hübel.

Continuous-variable quantum key distribution with gaussian modulation—the theory of practical implementations. *Advanced Quantum Technologies*, 1(1):1800011, 2018.

[MFZG18] Mario Milicevic, Chen Feng, Lei M. Zhang, and P. Glenn Gulak. Key Reconciliation with Low-Density Parity-Check Codes for Long-Distance Quantum Cryptography. *npj Quantum Inf*, 4(1):21, April 2018. arXiv:1702.07740 [quant-ph].

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[RU08] T. Richardson and R. Urbanke. *Modern Coding Theory*. Cambridge University Press, 2008.

[Sha49] C. E. Shannon. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28(4):656–715, 1949.

[Sho94] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[Zha18] Yusheng Zhao. Development of Quantum Key Distribution and Attacks against It. *J. Phys.: Conf. Ser.*, 1087:042028, September 2018.