Crittografia mediante permutazione pseudo-random di campioni di FFT

Terreni Maurizio

Università degli studi di Firenze
maurizio.terreni@stud.unifi.it

Abstract

L'elaborato consiste in un'applicazione Android. Il programma deve essere in grado di criptare e decriptare un segnale audio attraverso la trasformata di Fourier.

I. Introduzione

La parola crittografia ha origini Greche e indica la scrittura segreta. Consiste nell'arte di scrivere messaggi in codice che possono essere letti e compresi solo dalle persone autorizzate. Al giorno d'oggi scambi di dati tra due computer rappresentano progressi informatici ormai alla portata di tutti, infatti attraverso internet viaggiano sempre più spesso informazioni riservate, anche se trascinano con se molteplici rischi, come per esempio la raccolta di informazioni personali da parte di individui esterni. Uno dei metodi più sicuri per inviare dati è quello di modificarli attraverso funzioni matematiche, che con l'ausilio di una chiave privata rendono il file leggibile soltanto agli utenti che dispongono di tale chiave, questo viene chiamato messaggio criptato. Quindi l'obbiettivo della crittografia è quello di rendere un messaggio inutilizzabile a utenti estranei alla conversazione che non possiedono la chiave per decifrare i dati. Per la programmazione di android sono stati utilizzati varie documentazioni trovate in internet come ad esempio per la registrazione dei file audio in formato .wav, mentre per un quadro generale di come poter programmare per android sono state utilizzate le apiguide fornite dal sito androidSDK.

II. Principio di funzionamento

L'applicazione è in grado di acquisire dati audio, campionarli e successivamente eseguire la trasformata di Fourier, in modo tale da ottenere un segnale nel dominio della frequenza anzichè nel dominio del tempo, criptando direttamente lo spettro del segnale. I campioni che costituiscono lo spettro vengono salvati su di un array e tramite una chiave vengono scambiate le posizioni dei singoli elementi. Una volta fatto ciò lo antritrasformiamo, in modo da riacquisirlo nel dominio del tempo. Abbiamo cosi ottenuto un file audio apparentemente incomprensibile, ma, avendo la chiave con cui è stato criptato, possiabmo in maniera inversa decriptare il file riottenendo il messaggio in chiaro.

III. Trasformata di Fourier

La trasformata di Fourier è uno degli strumenti matematici maggiormente utilizzati. Essa permette di scrivere una funzione dipendente dal tempo nel dominio delle frequenze, e per fare ciò decompone la funzione nella base delle funzioni esponenziali con un prodotto scalare. Questa rappresentazione viene chiamata spesso spettro del segnale. La trasformata di Fourier è invertibile, infatti a partire dalla trasformata di una funzione $\tilde{x}$ è possibile risalire alla funzione x tramite l'inversa di Fourier.Data una funzione $f(t)$ reale o complessa a quadrato sommabile, periodica di pe-

riodo 2π, $\forall n$ intero esistono coefficienti complessi $c_n = \frac{1}{2\pi}\int_0^{2\pi} e^{-int} f(t)dt$ e la funzione può essere rappresentata come $f(t) = \sum_{n=0}^{+\infty} c_n e^{-int}$. L'interpretazione dello sviluppo in serie di Fourier è che un segnale periodico di potenza finita si può sviluppare come combinazione lineare di funzioni periodiche semplici la cui frequenza è un numero intero. Il tutto si generalizza facilmente al caso in cui il periodo si T e quindi la frequenza fondamentale abbia il valore $\frac{1}{T}$ si definisce $\int_{\infty}^{-\infty} f(t)dt = \lim_{T\to\infty}\int_{-T}^{T} f(t)dt$ ovvero si considera il valore principale nel senso di Cauchy; allora data una funzione f(t) reale o complessa a quadrato sommabile, sotto opportune condizioni analitiche, vale la coppia di trasformate:

$$F(\omega) = \int_{-\infty}^{\infty} e^{-i\omega t} f(t)dt$$
$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\omega t} F(\omega)d\omega$$

la funzione $F(\omega)$ viene detta integrale di Fourie. L'integrale di Fourie sviluppa una funzione aperiodica nelle sue componenti di frequenza.

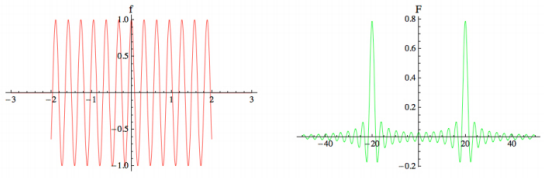

Figure 1: Esempio di un segnale nel dominio del tempo e di un segnale nel domionio della frequenza.

Il calcolo della trasformata discreta di Fourier richiede l'esecuzione di N moltiplicazioni ed N somme(commplesse) $\forall\ V_h, h = 0, 1, .., N-1$. Poichè si devono calcolare N valori, in totale sono richieste N^2 somme ed N^2 moltiplicazioni. Questo si può abbreviare dicendo che il calcolo del DFT ha complessità $O(N^2)$, cioè richiede un numero di operazioni proporzionali ad N^2. Un algoritmo molto utilizzato nel campo informatico ci aiuta a semplificare la complessità dei calcoli da eseguire diminuendone i tempi, questo algoritmo è chiamato Fast Fourier Transform, infatti permette di calcolare la trasformata $V_h, h = 0, 1, .., N-1$ con una complessità $O(N \log_2 N)$ cioè con un numero di calcoli proporzionale ad $N \log_2 N$.

IV. L'applicazione

L'elaborato è stato sviluppato per la piattaforma android, utilizzando le API fornite da google per l'acquisizione dei campioni audio. Il linguaggio di programmazione utilizzato è il Java per la parte di programmazione e xml per la parte grafica. L'applicazione è compatibile su tutti i dispositivi android sul qualve vi è installato una versione uguale o superiore ad android 4.0 ("Android Ice Cream Sandwich"). I file audio registrati vengono salvati su di una memoria esterna.

IV.1 Interfaccia

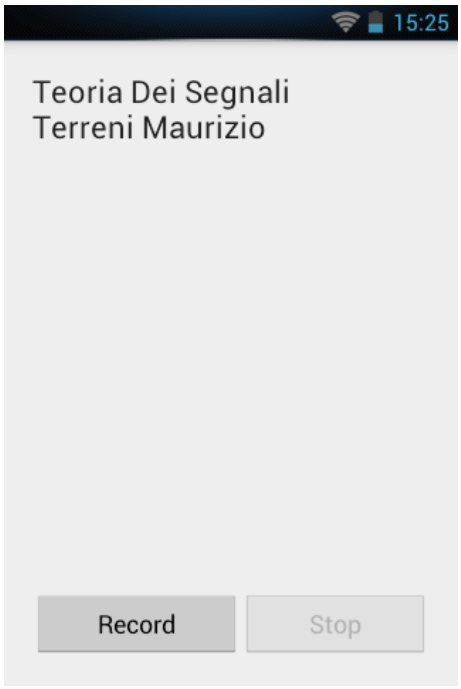

Figure 2: Schermata principale

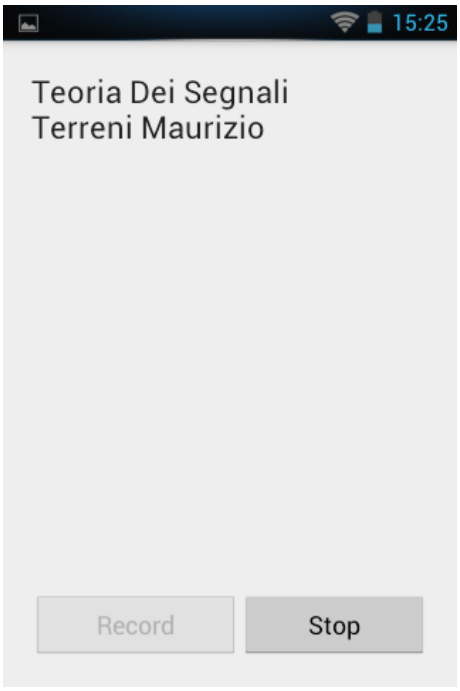

Figure 3: Fase di registrazione

Figure 5: Criptaggio terminato

Figure 4: Selezione Frequenza FFT

Figure 6: Cartelle contenenti i file elaborati

IV.2 Funzionamento

Appena viene avviata l'applicazione (fig. 2) è possibile iniziare a registrare (fig. 3) il priprio file audio premendo il pulsante "Start". Appena l'utente decide di interrpompere il messaggio premendo il tasto "Stop", l'applicazione in automatico salva il file in formato .wav appena registrato nella cartella "AudioRecorder". Successivamente viene richiesta la frequenza di campionamento da applicare alla trasformata di Fourier, l'utente infatti può selezionare una frequenza di campionamento che va da 8 campioni al secondo fino ad un massimo di 1024, l'aumento della frequenza permette una migliore riuscita del processo di criptazione. Una volta premuto il tasto "Esegui FFT" inizia il processo di criptaggio sul documento(fig. 4). L'algoritmo appena termina la lettura del file originale estrae i campioni ed esegue la trasformata di Fourier andando a salvare i risultati su di un array bidimensionale il quale contiene sia la parte reale sia la parte immaginaria dei singoli campioni. Successivamente attraverso una chiave privata calcolata attraverso una permutazione pseudo-random delle posizioni del vettore, l'algoritmo esegue lo swap dei campioni tenendo conto delle simmetrie. Per generare l'array contenete la chiave di permutazione è stato utilizzata la funzione di matlab "randperm(x,y)" dove $x = \frac{freq}{2}$ e $y = freq$ in questa maniera otteniamo un array permutato che rispetta la simmetria dei dello spettro del segnale. Nel caso in cui non venisse rispettata la simmetria una volta antitrasformato il segnale non riusciremmo piu a riottenere il messaggio originale. Terminato il processo di swap sui singoli elementi dell'array, rispettando i critedi imposti dalla F-trasformata, l'algoritmo ha introdotto un disturbo sul segnale, ma senza andare a compromettere il file originale, eseguirà quindi l'antitrasformata per riottenere dei campioni da salvare sempre nel formato .wav all'interno della cartella "AudioRecorder-Out"(fig. 5), ma al termine del processo,se un qualsiasi utente prova ad ascoltare il messaggio appena salvato sentirà solamente un file distorto e incomprensibile, solo attraverso la chiave sarà possibile riottenere tramite un processo inverso il file orginale. Oltre al file criptato l'algoritmo esegue anche la decriptazione salvando il file nella cartella "AudioRecorder-Out"(fig. 6) in maniera da garantire la perfetta esecuzione del processo.

V. Risultati Sperimentali

Al fine di testare l'applicazione, sono state eseguite criptazioni sullo stesso file per tutte le possibili frequenze della FFT. Successivamente è stato calcolato SNR_{db} di tutti i test eseguiti.

Per calolare SNR_{db} è stata utilizzata la seguente formula:

$$SNR_{db} = 10log_{10}\frac{\sum_{n=1}^{N} X_0[n]^2}{\sum_{n=1}^{N}(X_0[n]^2 - X_r[n]^2)}$$

Freq.	SNR_{db}
8	-4.1334
16	-5.5347
32	-4.7785
64	-4.1739
128	-4.9479
256	-4.8138
512	-4.8535
1024	-4.3877

Table 1: Tabella SNR_{db} dei file audio

La Tabella1 mostra il rapporto segnale/rumore tra il file audio originale e quello ricostruito per tutte le varie frequenze di decriptazione. Anche se riusciamo a ricostruire perfettamente il segnale di partenza notiamo che rimane una leggara distorsione che comunque non influenza la comprensione del messaggio registrato.

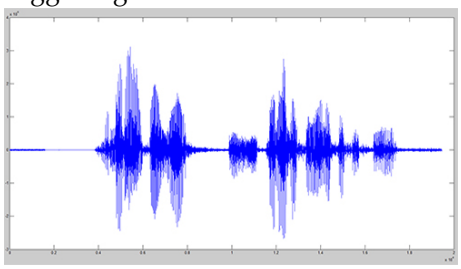

Figure 7: Segnale Registrato

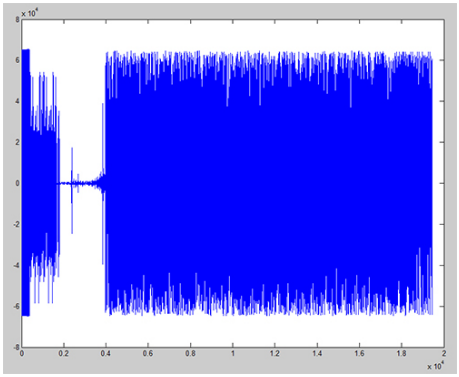

Figure 8: Segnale Criptato

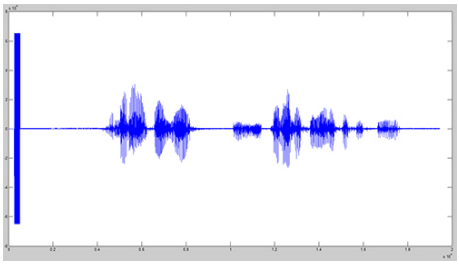

Figure 9: Segnale Ricostruito

Le figure riportate mostrano le modifiche apportate al file audio una volta criptato il messaggio e successivamente la ricostruzione del file originale.

VI. Problemi Riscontrati

1. Leggera distorsione nel file decriptato, che però non compromette il messaggio registrato
2. Elevato carico di CPU per messaggi molto lunghi.
3. Possibilità di registrare soltano audio di tipo MONO.

VII. Possibili Soluzioni

1. Aumentare la frequenza dei compioni nella trasformata di Fourier, e introdurre un filtro in maniera tale da eliminare i disturbi esterni che potrebbero compromettere il file originale.
2. Utilizzo algoritmi più efficienti per eseguire la trasformata di Fourier.

VIII. Conclusioni

Attraverso questo elaborato ho appreso come trasformare un segnale appartenente al dominio del tempo nel dominio delle fequenze. In oltre ho anche appreso come poter manipolare i vari campioni dello spettro del sengale riuscendo comunque ad riottenere un segnale nel dominio del tempo.

IX. Bibliografia

References

[1] Gianfranco Cariolaro, Gianfranco Pierobon *Segnali e sistemi*, McGraw-Hill, 2005.

[2] Fabrizio Argenti,Lorenzo Mucchi, Enrico del Re *Elaborazione numerica dei segnali* , McGraw-Hill, 2011.

[3] Massimo Carli *Android 4 : [Guida per lo sviluppatore]*, Apogeo, 2013.

[4] Android wav tutorial. *(www.edumobile.org/android/audio-recording-in-wav-format-in -android-programming/*

[5] Android sdk website *www.developer.android.com/sdk/index.html)*