**Università degli Studi di Genova**

Scuola Politecnica

Corso di Formal Languages and Compilers – A.A. 2013/2014

Prof. Giovanni Adorni

# Progettazione di un software per la rilevazione automatica di plagi

Palermo Roberto

Pignedoli Andrea

Valenza Andrea

# Indice dei contenuti


**1. Introduzione** **3**

**2. Studio di fattibilità** **5**

**3. Progettazione** **8**

3.1 Upload del testo 9

3.2 Ricerca pool testi correlati 10

3.3 Comparazione dei testi 11

3.4 Produzione testo aumentato 13

**4. Realizzazione pratica** **14**

4.1 Abstract 14

4.2 Installazione 14

4.3 Caso d’uso 15

4.4 Descrizione del codice sorgente 15

**5. Conclusioni** **19**

# Introduzione

L'obiettivo di questa relazione è quello di rendere un po' più chiaro, anche ad un profano della materia, in cosa consiste questo progetto sul Plagio. Questo documento è strutturato in più parti che ci guideranno attraverso i diversi aspetti dell'argomento trattato, dai più teorici fino all'implementazione più tecnica e dettagliata.

Questo progetto è nato al termine del corso di "Formal languages and compilers" che ci ha permesso di imparare diverse tecniche formali che stanno dietro agli attuali compilatori ed è proprio con queste conoscenze che ci siamo proposti di applicarle per realizzare un rilevatore di plagi. L'idea che sta alla base del nostro progetto è utilizzare delle grammatiche formali per poter estrarre da un testo il suo contenuto informativo e quindi, sulla base di quel contenuto, verificare se si tratta di un plagio di qualcosa già presente in rete per esempio. Al completamento di questo lavoro sarà possibile ottenere un responso, dato un testo in input, sul grado di plagio del testo stesso, potendo quindi valutare l'originalità del contenuto.

Al giorno d'oggi, avendo a disposizione l'intera rete internet come fonte, è facile imbattersi in testi che non sono altro che il frutto di un fantasioso copia-incolla oppure di una semplice rielaborazione. Con l'utilizzo di grammatiche formali sarà possibile analizzare nel dettaglio la struttura semantica della frase e pertanto riuscire a rilevare un tentativo di plagio.

Verranno in seguito spiegate nel dettaglio le grammatiche di cui si fa uso ed i vari algoritmi e strategie utilizzati per fare lo scanning dei testi in input. Il compito che ci siamo fissati è quello di mettere delle solide basi ed impostare la giusta

direzione a quello che è un progetto molto ambizioso che potrà essere ampliato a piacimento ed utilizzato in futuro in ambito reale.

# Studio di fattibilità

Per la risoluzione del problema oggetto del nostro studio, abbiamo analizzato differenti soluzioni con crescente accuratezza e grado di difficoltà nella realizzazione. Il sistema che deve essere realizzato deve permettere, come già descritto nell'introduzione, il riconoscimento di varie forme di plagio presenti nei documenti, prevalentemente accademici. In particolare è necessario, inserendo un documento nel sistema, riconoscere se e in che misura esso contiene dei plagi di altri documenti.

Uno dei problemi, quindi, incontrati inizialmente è stato la scelta di un accurato insieme di documenti da confrontare con quello su cui vogliamo effettuare l'analisi. Basandoci sia su progetti precedenti che affrontavano lo stesso problema sia su conoscenze personali, siamo giunti alla conclusione che la fonte da cui attingere i documenti per il confronto fosse la Rete Internet, mediante l'utilizzo adeguato dei motori di ricerca.

Per questo stretto legame tra l'analisi da effettuare ed Internet, oltre che per la sua flessibilità, potenza, disponibilità di librerie, vasto supporto e compatibilità, abbiamo deciso di realizzare un'applicazione web mediante il linguaggio di scripting PHP. Esso permette di realizzare facilmente e senza particolari limitazioni un portale per la ricerca di plagi all'interno di documenti, eventualmente estensibile ad un vero e proprio servizio web in futuro, disponibile su tutti i sistemi operativi e tutte le piattaforme. Inoltre si tratta di un sistema scalabile, che può essere installato su macchine di potenze adeguatamente dimensionate rispetto alla quantità di richieste da servire.

Il problema della ricerca dei plagi di per sé risulta senza dubbio risolubile, è stato infatti oggetto di numerosi Tesi di Laurea triennali. La differenza rispetto ad esse è

il livello di accuratezza con cui viene affrontato, cercando un metodo flessibile e facilmente adattabile a differenti formati di file o linguaggi. Inoltre altro scopo della nostra ricerca era ottenere un procedimento basato il più possibile su grammatiche formali, seppur coi limiti che esse intrinsecamente incontrano nella loro applicazione a linguaggi naturali, che permettesse il riconoscimento non solo dei plagi più "palesi", ma anche a forme più sottili ed elaborate di copiatura, come ad esempio quella a cui segue una parziale rielaborazione.

Le Tesi Triennali che abbiamo preso in oggetto per un'analisi preliminare effettuavano una scomposizione del documento in periodi e ricercavano sulla Rete se ciascuna frase comparisse all'interno di altri documenti. Tale metodo risulta efficace per plagi molto forti, dove interi pezzi di documenti vengono inseriti senza alcuna modifica all'interno del proprio elaborato, non è però in grado di riconoscere il plagio dopo una parziale rielaborazione e risulta poco efficiente a causa del numero di query da lanciare sui motori di ricerca.

Per trovare una soluzione più efficiente, perciò, è necessario salire ad un livello di astrazione superiore. Lo scopo che ci siamo preposti è quello di ricavare quindi un metodo il più possibile modulare, applicabile in modo generale. La soluzione che abbiamo progettato e che verrà descritta all'interno della presente relazione si basa in modo preponderante sull'uso dei metodi formali per l'applicazione di grammatiche sul testo. Analizzando ciascuna frase che compone il documento è possibile applicare una grammatica generica che consente l'estrapolazione delle componenti principali. In questo modo è possibile quindi effettuare un confronto tra le singole frasi dei documenti a livelli di astrazione via via più elevati, ricavando in questo modo un "grado di plagio" di ciascuna frase. Per farlo, però, una componente fondamentale del processo risulta essere la disponibilità di un "dizionario" che contenga per ciascuna parola la sua tipologia e le varie "declinazioni". Attraverso questo strumento di analisi e confronto, appoggiandosi

su una struttura dati appositamente progettata basata, per esempio, su un database MySQL, è possibile confrontare un documento di input con un pool di documenti presenti nel sistema, determinando in modo fine un livello di plagio, dettagliato per ciascun periodo, e la fonte dello stesso.

Non avendo a disposizione in locale però tutti i possibili documenti fonte di plagio e, anche nell'ipotesi di averli, risultando inefficiente effettuare un confronto di questo tipo su una vasta quantità di materiale, nasce il problema della creazione del "pool di testi".

Una componente fondamentale del sistema è quindi l'algoritmo che, dato il documento di input, è in grado di creare una base di testi con cui effettuare il confronto. Nella soluzione da noi pensata, tale algoritmo è in grado di estrapolare una serie di "keywords" dal testo ed effettuare delle ricerche sulla Rete, scaricando i documenti che si pensano essere maggiormente correlati a quelli in input. L'accuratezza di tale algoritmo è senza dubbio importante per la qualità della soluzione data al problema.

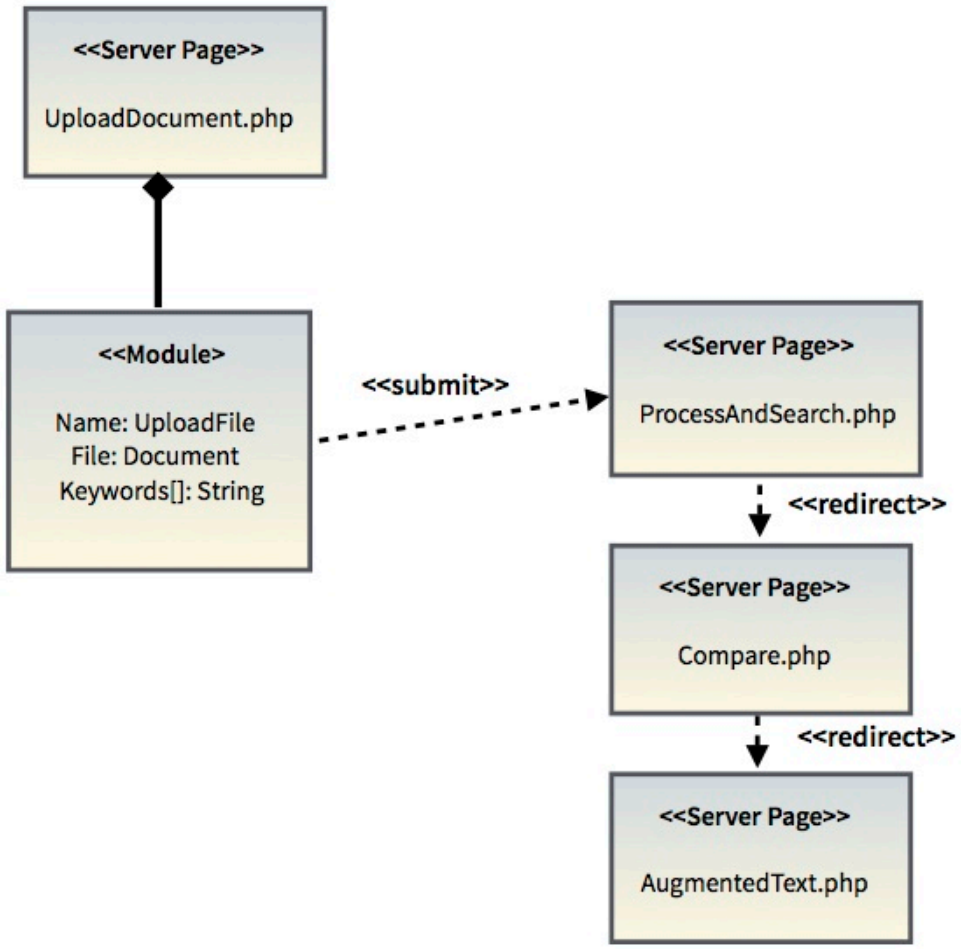
Un altro problema da affrontare è quello della tipologia di file da analizzare. Abbiamo pensato di risolverlo in modo modulare: attualmente la nostra soluzione è in grado di analizzare documenti di tipo HTML ma è possibile inserire "plugin" per l'analisi di altri formati.

Lo stesso vale per il problema della lingua: è possibile, attraverso l'inserimento di dizionari e di un plugin che rappresenti la grammatica, inserire nuovi linguaggi nel sistema.

Disponendo di un tempo limitato per la realizzazione di un sistema completo, abbiamo realizzato una "proof of concept" che potrà essere estesa ed approfondita in progetti futuri mediante il supporto della presente relazione, che spiega dettagliatamente i concetti e le soluzioni necessari per risolvere il problema della rilevazione dei plagi.

# Progettazione

Il funzionamento del rilevatore di Plagio può essere facilmente scomposto in quattro fasi: l'upload del testo da verificare, la ricerca di altri testi correlati a quello principale, il confronto vero e proprio del contenuto informativo di tali testi e come ultima fase viene fornito un responso sul livello di plagio del documento in esame. Vediamo brevemente ognuna di queste fasi. Per prima cosa l'utente deve fare l'upload tramite un apposito form web del testo per il quale desidera eseguire l'analisi. Una volta completata la prima fase il testo viene analizzato da un algoritmo (nei paragrafi successivi sarà descritto il tutto nel dettaglio), il quale estrae alcune keywords che caratterizzano l'argomento trattato e sulla base di queste vengono ricercati (su un db locale o sul web) altri documenti correlati che saranno successivamente oggetto del confronto. Arrivati al punto in cui si possiede un'adeguata quantità di testi adatti ad essere confrontati, viene eseguito un altro algoritmo che ci consente di valutare frase per frase il livello di plagio di tale porzione di testo. Al termine di questa procedura abbiamo raccolto i dati relativi all'intero testo e possiamo mostrare all'utente lo stesso testo che aveva uploadato sul nostro server ma aumentato con le informazioni estrapolate dall'analisi sul plagio. Questo testo aumentato rappresenta il risultato complessivo dell'analisi e ci permette di capire se il testo in ingresso al sistema è un plagio o meno, ed in caso affermativo verremo messi anche al corrente della profondità ed estensione di tale plagio.




**ProcessAndSearch.php**: analisi del testo, ricerca e salvataggio pool di test correlati
**Compare.php**: comparazione tra testo caricato e pool di testi
**AugmentedText.php**: visualizzazione del testo aumentato con percentuale di matching e riferimenti ai testi plagiati.

**Figura 1 - Diagramma Web-UML semplificato**

## Upload del testo

All'inizio del procedimento è necessario fornire al sistema di rilevazione dei plagi il testo del quale si vuole compiere l'analisi. Per questo motivo la pagina sarà dotata di un form web adatto all'upload del documento da analizzare. Tale documento potrà essere di tipo testuale, un PDF o una pagina web. In ogni caso il sistema sarà estensibile a piacimento aggiungendo i moduli software contenenti le grammatiche relative al tipo di file del quale si vuole aggiungere il supporto.

Per l’analisi dell’HTML abbiamo elaborato una grammatica ad hoc che permette l’estrazione dei contenuti significativi:

```
<DOCUMENTO> ::= “<HTML>”<HTML>”</HTML>”
<HTML> ::= <HEAD><BODY>
<HEAD> ::= “<HEAD>”<HEAD_EL><ALTRO_HEAD_EL>”</HEAD>”
<HEAD_EL> ::= <TITLE> | <KEYWORDS> | <DESCRIPTION> | <ALTRO>
<ALTRO_HEAD_EL> ::= <HEAD_EL><ALTRO_HEAD_EL> | ε
<TITLE> ::= “<TITLE>”[lettera_’ ’numero]*”</TITLE>”
<KEYWORDS> ::= “<META NAME = “KEYWORDS”>” parola[‘ ‘ parola]* ”</META>”
<DESCRIPTION> ::= “<META NAME = “DESCRIPTION”>” [lettera_num]* “</META>”
<ALTRO> ::= [lettera _ / ‘ ‘ < > = numero]*
<BODY> ::= “<BODY>”<BODY_EL><ALTRO_BODY_EL>”</BODY>”
<BODY_EL> ::= <H1> | <H2> | <H3> | <P> | <SPAN> | <ALTRO>
<H1> ::= “<H1>”<CONTENT>”</H1>”
<H2> ::= “<H2>”<CONTENT>”</H2>”
<H3> ::= “<H3>”<CONTENT>”</H3>”
<P> ::= “<P>”<CONTENT>”</P>”
<SPAN> ::= “<SPAN>”<CONTENT>”</SPAN>”
<CONTENT> ::= <H1><CONTENT> | <H2><CONTENT> | <H2><CONTENT> | <H3><CONTENT> |
<P><CONTENT> | <SPAN><CONTENT> | <ALTRO>
```

## Ricerca pool testi correlati

Al termine del caricamento del file da analizzare il sistema provvederà a cercare un pool di testi correlati sul web o in un'altra posizione che sarà possibile specificare al momento. Tale ricerca si basa sull’analisi del documento per capire l’argomento trattato e di conseguenza cercare un pool di testi significativo ed adatto ad un confronto. Per esempio, nelle pagine html spesso è presente una porzione di codice (tag content con attributo keywords) che ci permette già di comprendere velocemente le parole chiave attorno alle quali si articola il documento. Nel caso non fosse una pagina web entrerebbero in gioco alcuni meccanismi come l’analisi di frequenza delle parole per poter individuare manualmente delle keywords rappresentative del testo.

## Comparazione dei testi

Lo script che si occupa della procedura di comparazione dei testi è il cuore di tutto il sistema. Questa parte si divide in fasi:

- ogni testo nel sistema viene segmentato in frasi, utilizzando una semplice grammatica basata sui segni di interpunzione
- ciascuna frase viene analizzata e al suo interno vengono individuate, attraverso una grammatica basata su dizionario, le parti influenti e quelle ininfluenti; in particolare vengono scartati gli articoli e le congiunzioni
- le parti considerate influenti vengono classificate e ne viene individuata, attraverso una ricerca nel dizionario, una forma "generica"; ovvero, per esempio, per i verbi la loro forma infinita e per i sostantivi la forma maschile singolare
- la frase ripulita e portata in forma generica viene salvata in una struttura dati, associata alla frase completa e al numero e al tipo di elementi che la compongono.

Una volta che tale procedura è stata completata per ogni testo, inizia il confronto vero e proprio. Ciascuna frase del documento caricato dall'utente viene confrontata con ciascuna frase di ogni testo presente nel pool di confronto e, al termine di tale fase, viene restituito un "grado di plagio della frase". Questo viene determinato in base ad un serie di confronti successivi, durante i quali il grado decresce progressivamente:

- corrispondenza esatta delle frasi a livello di stringa
- corrispondenza del numero di elementi, del tipo degli elementi, della posizione in cui si incontrano nella frase e della loro forma generica
- corrispondenza del numero degli elementi, del tipo degli elementi e della posizione

- corrispondenza del numero degli elementi e del tipo degli elementi
- corrispondenza del numero degli elementi

Naturalmente altri criteri di confronto possono essere individuati e la scala del livello di plagio può essere caratterizzata in modo differente, mantenendo però l'idea di fondo del sistema.

Al termine di questa fase verrà salvato nella struttura dati, per ciascuna frase del documento caricato dall'utente, il suo livello di plagio e un riferimento ai documenti sorgenti dell'eventuale plagio. Queste informazioni verranno utilizzate nella fase successiva per mostrare un "documento aumentato" che renda evidente il livello di plagio e l'accesso ai documenti da cui proviene.

Per effettuare le analisi descritte in precedenza, risulta fondamentale definire una grammatica formale da applicare sui testi. La premessa da apporre necessariamente è quella di aver effettuato una "pulizia" dei documenti in modo da riportarli a puro *plaintext* prima dell'applicazione delle grammatiche.

La ricerca delle frasi all'interno del documento e delle parti che compongono ciascuna frase avviene attraverso l'applicazione del seguente pattern, modificabile a seconda del linguaggio utilizzato nel testo:

```
<CONT> ::= <FRASE><ALTRE_FRASI>
<FRASE> ::= <PAROLA><ALTRE_PAROLE><DELIMITATORE>
<PAROLA> ::= sostantivo | predicato | aggettivo | avverbio | articolo | altro
altro ::= [. , _ " ' -]* //caratteri da ignorare
sostantivo ::= DA INDIVIDUARE TRAMITE DIZIONARIO
predicato ::= DA INDIVIDUARE TRAMITE DIZIONARIO
aggettivo ::= DA INDIVIDUARE TRAMITE DIZIONARIO
avverbio ::= DA INDIVIDUARE TRAMITE DIZIONARIO
articolo ::= DA INDIVIDUARE TRAMITE DIZIONARIO
<ALTRE_PAROLE> ::= <PAROLA><ALTRE_PAROLE> | ε
<DELIMITATORE> ::= ['\n' . ! ?]*
```

## Produzione testo aumentato

Una volta completata la procedura di analisi e confronto del testo, il rilevatore di plagi produrrà una pagina in output che contiene lo stesso testo in input aumentato in relazione al grado di plagio che è stato rilevato dal sistema. Per esempio se un segmento di frase è stato trovato tale quale nello stesso contesto avremo il massimo grado di plagio, altrimenti si scende fino al livello zero. Tali gradi sono evidenziati nel testo anche attraverso l'uso di colori per rendere più evidente, anche a colpo d'occhio, il risultato dell'analisi. Inoltre, per ciascuna frase, saranno inseriti dei collegamenti ipertestuali alla fonte del plagio per effettuare un confronto diretto.

# Realizzazione pratica

## Abstract

Il codice incluso è una dimostrazione del funzionamento del progetto, non la sua completa realizzazione.

Molti punti sono lasciati intenzionalmente aperti per poter inserire nuovi “plugin” e ampliare il progetto.

Qui di seguito sono inserite istruzioni sulla struttura del codice e sui suoi utilizzi, ma si possono trovare molte delle informazioni qui contenute esaminando la documentazione contenuta nel codice stesso e nella pagina doc/index.html, creata con APIGen (generatore automatico di documentazione per codice PHP).

## Installazione

**Requisiti**:

1. Database MySQL con nome **plagio**
2. *Utente* **plagio** con *password* **Plagio** e privilegi di lettura e scrittura sul Database
3. Tabella chiamata **frasi** con queste colonne:
   + **id** *int*, PK e AI
   + **frase** *TEXT*
   + **source** *TEXT*

## Caso d’uso

### Primo avvio

(Si suppone che il progetto si trovi all'indirizzo *http://localhost/Plagio/*)

Oltre ai form e agli script, sono presenti 4 file nel progetto:

1. *http://localhost/Plagio/tempPool/pool1.html*
2. *http://localhost/Plagio/tempPool/pool2.html*
3. *http://localhost/Plagio/tempPool/pool3.html*
4. *http://localhost/Plagio/prova.html*

I primi 3 sono file che verrebbero caricati in automatico nel *Pool* di testi, il quarto è l'articolo che dobbiamo controllare.

1. Entrare nel *http://localhost/Plagio/admin.html* e inserire l'url completo di *http://localhost/Plagio/tempPool/pool1.html*, attendere il caricamento.
2. Ripetere per gli altri testi nella pool (**NON** per prova.html, altrimenti sarà sempre positivo)
3. Entrare nel *http://localhost/Plagio/index.html* e inserire l'url completo di *http://localhost/Plagio/prova.html*.
4. Ogni frase apparirà stampata a schermo come *frase semplice* nel caso non venga trovata corrispondenza, oppure come *link ipertestuale* nel caso venga trovata. Notare che cliccando sul link si viene reindirizzati alla pagina dalla quale è stata copiata la frase.

## Descrizione del codice sorgente

### index.html

Questa è la “home” del sito; da qui l’utente inserisce il testo che vuole controllare e riceve la risposta.

Dal momento che il formato di verifica è solo HTML, non sono permessi upload di file, ma solo link (che possono essere assoluti o anche relativi). Nel momento in cui fosse necessario fare upload di file, si può escludere il form dal flow, creando un form di Upload invece che di inserimento stringa e passando poi l'url temporaneo (ottenuto dopo l'upload ad esempio tramite $_FILES['tmp_name'] o altri metodi) e mantenere comunque l'architettura già creata.

**admin.html**

Da questa pagina è possibile caricare testi che vengono sottoposti al parser (funzione tokenize) e vengono quindi inseriti nel database.

Questa operazione è molto pesante da fare per grandi quantità di files da controllare. Bisogna ricordare però che questo è il comportamento della demo, il comportamento successivo ideale sarebbe automatizzare questo comportamento inserendo un motore di ricerca dedicato. La pool di testi verrà quindi riempita in modo automatico per ogni controllo di documento; per riempire la pool è già suggerito un metodo all'interno di questa relazione, ma è possibile usare anche metodi più raffinati o euristiche secondo necessità; per il comportamento suggerito (utilizzo di meta tags, titolo e keywords di html5) si usano valori che vengono stampati dal parser durante l'inserimento in index.html, una volta che il testo è formattato da ProcessAndSearch.php.

**ProcessAndSearch.php**

Questo è il cuore del progetto, in quanto svolge tutte le funzioni necessarie al parsing del testo fornito dall'utente e fornisce anche le keyword per effettuare la ricerca dei testi da inserire nella pool (non implementato nella demo). Fornisce poi i token "aumentati" con le informazioni sul plagio che stampa poi a video.

In questa fase la stampa è stata fatta direttamente con un *echo* dalla pagina. Sarebbe meglio, quando la pagina diventa più complicata, creare un template con una template engine (si consiglia Twig, già integrata) e riempire quello.

**upload.php**

Questo script serve ad inserire i token creati con tokenize nel database, tenendo traccia della sorgente da cui sono stati presi (*source*).

**includes/util.php**

Contiene la funzione **tokenize** e altre funzioni utili al resto della demo.

La funzione tokenize è il vero e proprio parser del documento, crea i token che vengono poi usati per fare i confronti.

Questa funzione può essere modificata a piacimento per seguire le regole della grammatica; in questo caso si fanno semplicemente delle divisioni delle stringhe in “frasi” usando delle espressioni regolari e delle funzioni php, ma bisogna ampliare questa funzionalità e aggiungere .

Tokenize si appoggia alla funzione *plaintext* fornita dalla libreria *simplehtmldom* che in questo caso serve a mantenere il testo della pagina eliminando i tag html presenti nel documento.

**vendor/simplehtmldom/**

Questa è la libreria di terze parti che viene usata per rimuovere i tag e analizzare il documento html. Può essere sostituita con una migliore se dovesse servire o può essere completamente eliminata e sostituita con una scritta appositamente.

**vendor/composer/ + composer.json + composer.lock**

Composer.phar è un gestore di librerie che semplifica l'aggiunta e la modifica dei vendor all'interno del progetto.

# Conclusioni

Alla luce di quanto visto durante lo sviluppo di questo progetto possiamo dire che tramite l'utilizzo di linguaggi formali è stato possibile trovare un processo adatto ad individuare dei pattern comuni all'interno dei testi sottoposti ad analisi. La difficoltà principale del progetto era proprio quella di trovare un approccio alternativo che non si basasse su tecniche euristiche. Così facendo, è stato quindi possibile costruire uno strumento che sia in grado di restituire il grado di plagio del documento in ingresso rispetto ad un insieme di documenti ad esso attinenti. Il processo è suscettibile ovviamente di miglioramenti ed affinamenti ma è stata fissata una solida base di partenza dalla quale si potrà costruire qualcosa di più ampio e strutturato, come ad esempio aggiungere altri moduli per supportare più tipi di file in input. Al di là della specifica tecnica implementativa, riteniamo che il lavoro svolto ci sia stato molto utile per comprendere bene l'utilità pratica delle grammatiche e più in generale dei linguaggi formali, creando al contempo qualcosa che, seppur in veste di esperimento, va a ricoprire un campo nel quale ancora oggi l'informatica non ha trovato una soluzione viabile e definitiva per risolvere il problema dei plagi.