Università degli Studi di Salerno

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

Sviluppo di un tool per la comparazione qualitativa di algoritmi di confronto fra stringhe basati sulle Minimal Absent Words

Relatrice
Prof.ssa Rosalba Zizza

Candidato
Antonio Gravino
Matricola: 05121 07161

Anno Accademico 2021/2022

Alla mia famiglia,
faro in un oceano di incertezze.

Ai miei colleghi universitari,
esempi di vigore e costanza.

A me,
per aver perseverato nonostante le avversità.

E infine al popolo ucraino,
che davanti ad un nemico apparentemente invincibile,
ha dato dimostrazione di grande forza e risolutezza,
rinnovando profondamente il mio spirito europeista.

Indice

Abstract

I confronti fra stringhe sono un campo applicativo critico nell'informatica e, in particolare, nella bioinformatica. In letteratura, nel corso dei decenni sono state proposte numerose soluzioni nella forma di algoritmi più o meno efficienti in grado di misurare in maniera precisa il grado di similitudine fra un insieme di stringhe. Tuttavia, ci si pone il problema di elaborare un algoritmo di confronto fra stringhe che, invece di computare in funzione dell'informazione positiva - ottenuta analizzando la composizione propria e reale delle stringhe in input - sfrutti piuttosto l'informazione negativa intrinseca delle stesse nella forma delle loro *minimal absent words*. Successivamente, si analizza lo stato dell'arte dei tool visuali adibiti alla generazione di grafici illustrativi riguardanti i medesimi confronti, e si propone un'alternativa adatta alle speciali esigenze in materia di studio, sviluppando un tool web in grado di accogliere, integrare e computare algoritmi basati sull'informazione negativa in virtù di un'opportuna metrica, e dunque capace di elaborare uno spettro di rappresentazioni grafiche che consentano di effettuare valutazioni qualitative degli algoritmi impiegati.

Capitolo 1

Introduzione

Capitolo 2

Cenni teorici e nozionistici

Capitolo 3

Analisi dell'algoritmo scMAW

Capitolo 4

Analisi del tool SMART

Capitolo 5

Realizzazione del tool LW Index

In virtù delle mancanze del tool SMART delineate nel Capitolo 5, si è deciso di procedere alla realizzazione di un'applicativo web ad hoc per sopperire alle lacune del suddetto tool.

Il tool in sviluppo, denominato LW Index, in riferimento all'omonima metrica presentata nel Capitolo 2, è definito come SMART-like, o simil-SMART. Il tool SMART permetteva di confrontare una serie di algoritmi di confronto fra stringhe, a prescindere dalla loro natura, e generare conseguentemente dei grafici predisposti a valutazioni di natura qualitativa da parte dell'utente.

Benché sia disponibile in una versione parallela adoperante un'interfaccia grafica, SMART nasce come programma da terminale; nella fattispecie, come si è osservati nel Capitolo 4, è in grado di produrre - a partire da una collezione di stringhe (o più banalmente un testo) e uno spettro di algoritmi selezionati - una pagina web costruita tramite elementi di HTML, CSS e JavaScript.

D'altro canto, l'interazione dell'utente nasce e muore nel contesto del terminale: difatto, la pagina web autogenerata da SMART è di mera consultazione e non prevede ulteriore interazione con l'utente.

L'obiettivo è dunque di creare un tool che riesca ad accogliere algoritmi basati sulla metrica LW, incompatibili coi metodi di elaborazione e computazione di SMART; si noti, in ogni caso, che la metrica LW e le *minimal*

absent words sono strettamente legate, motivo per cui il tool è stato pensato specificamente per algoritmi che, oltre a misurare la similitudine con la metrica LW, facciano uso di *minimal absent words* e derivati nella loro composizione funzionale.

5.1 Progettazione del tool

Si osservi che, d'ora in poi, l'espressione "tool in sviluppo" sarà utilizzata in maniera intercambiabile col suo proprio nome, quale LW Index.

5.1.1 Caratteristiche e funzionalità ad alto livello

In generale, il tool in sviluppo permetterà di creare grafici (al più 4) in maniera del tutto simile al tool SMART; svuotare la tavola di lavoro (cioè distruggere i grafici–tutti o individualmente) e analizzare i grafici. Creare un grafico significa, ad alto livello, generare un'istanza di lavoro vuota, popolabile con dati e stringhe durante la fase di analisi.

Analizzare i grafici significa porre in evidenza uno dei grafici fra quelli creati sulla tavola di lavoro ed effettuare una serie di azioni, fra cui visionare, in maniera grafica o testuale, il processo computazionale svolto per arrivare a quei dati, e modificare i grafici aggiungendo, rimuovendo o alterando stringhe.

E' importante sottolineare che modificare il grafico significa necessariamente creare una nuova richiesta computativa.

Sarà possibile cambiare algoritmo. Il tool sarà sufficientemente versatile da accogliere un insieme diversificato di algoritmi che hanno in comune l'impiego della stessa metrica di similitudine fra stringhe, quale l'indice LW.

I grafici rappresenteranno sulle ascisse le stringhe (su alfabeti ben definiti) e sulle ordinate il corrispettivo indice LW. Questo significa che, per

ogni stringa, il grafico produrrà una curva distinta a cui assocerà un indice LW per ogni altra stringa di quel grafico.

Come accennato, il tool potrà ospitare al più 4 grafici per volta, ed ogni grafico sarà analizzabile e modificabile indipendentemente dagli altri.

5.1.2 Architettura dell'applicazione

Il tool LW Index adotterà un'architettura web di tipo client-server. L'utente si interfaccerà con l'applicazione attraverso un browser web come Google Chrome o Mozilla Firefox. Questa scelta risulta funzionale perché il tool dovrà offrire un maggior numero di opzioni d'interazione rispetto a SMART, e preservare i dati fra un'esecuzione e l'altra.

Gli algoritmi di confronto dovranno essere ospitati sul server in una specifica directory.

Nel dettaglio, l'utente potrà avviare l'applicativo tramite terminale. Una volta lanciato uno specifico comando, un'istanza del client (ossia una pagina web) si aprirà e contemporaneamente il webserver si avvierà.

Fra le caratteristiche proposte al punto 5.1.1, l'unica funzionalità che richiede l'intervento del server è la modifica di un grafico. Si noti che, pure se un grafico è vuoto (e cioè non è stato ancora popolato da stringhe), si sta affrontando un'operazione strettamente di modifica. Ogni volta che si richiede di aggiungere, eliminare o modificare stringhe, e si avvia il processo computativo alla pressione di un tasto, il webserver dovrà essere contattato e informato sulla natura della richiesta, per poi trattare i dati in put ed eventualmente fornire in output i dati necessari alla generazione del grafico.

5.1.3 Scelta delle tecnologie impiegate nell'implementazione

L'applicativo sarà implementato con tecnologie moderne e convenienti, al fine di semplificare la fase di implementazione e ridurre al minimo i casi di errore da gestire. Si è dunque deciso di implementare il webserver e il

backend dell'applicazione con Node, mentre il frontend del client con React. Il techstack costituito da React e Node è un'alternativa popolare all'utilizzo grezzo della tecnologia fondamentale dalla quale entrambi derivano, quale JavaScript.

Per ciò che concerne il frontend, le applicazioni web implementate in React possono sfruttare JavaScript o TypeScript; quest'ultimo, notoriamente, è un'alternativa al primo ed è caratterizzato da una forte tipizzazione, del tutto mancante invece in JavaScript.

E' stato deciso di implementare il tool in sviluppo con la variante TypeScript di React.

Per ciò che concerne il backend, Express, popolare framework di Node, è stato selezionato per l'implementazione del webserver. Express permette di avere immediatamente disponibile un ambiente di sviluppo facilmente configurabile, al quale è possibile agganciare nuove API attraverso sistemi di routing personalizzabili.

Questa scelta risulta immediata poiché il server dovrà ammettere un singolo processo elaborativo, quale la trasmissione dei dati fra il client e l'eseguibile dell'algoritmo selezionato per la computazione, sintetizzabile in un'unica route del webserver.

Ulteriori informazioni su React e Node sono reperibili in italiano rispettivamente ai seguenti indirizzi: `https://it.reactjs.org/` e `https://nodejs.org/it/`.

5.1.4 Struttura del filesystem dell'applicazione

A basso livello, la struttura del filesystem di LW Index si presenta come segue. La directory `/api` conterrà il sorgente del backend, mentre la directory `/client` il sorgente del frontend. Entrambe le directory sono contenute in un macrospazio di lavoro denominato `codebase`.

```
~/{root}
```

```
/api
/client
```

Il contenuto predefinito di ambo le cartelle è costituito dai file predefiniti di React (variante TypeScript) e Node (variante Express). Una volta installati rispettivamente React e Node, infatti, le due cartelle si popoleranno con un insieme di file critici per il funzionamento dell'applicazione. Dalla struttura base fornita dal processo di installazione, è possibile iniziare ad implementare le funzionalità dell'applicativo.

Gli algoritmi di confronto saranno collezionati in una sub-directory di `/api` che prenderà il nome di `/algo`. La cartella `/algo` presenterà, dunque, un insieme di sottocartelle, ognuna contenente il codice sorgente del proprio algoritmo.

```
/api
    /algo
        /Algoritmo_1
        /Algoritmo_2
        ...
        /Algoritmo_N
```

Il contenuto della i-esima cartella `/Algoritmo_i` è del tutto indipendente rispetto al resto dell'applicativo. Questo significa che gli algoritmi possono essere strutturati in qualsiasi modo (ad esempio utilizzare certe strutture dati piuttosto che altre), a patto che siano implementati in C o C++ e che risultino pre-compilati ed eseguibili una volta inseriti nella propria directory. Non è, inoltre, possibile avere casi di omonimia fra directory ospitanti i sorgenti degli algoritmi.

5.2 Implementazione del tool

5.3 Esempi di utilizzo pratico

Capitolo 6

Conclusioni e possibili sviluppi futuri