

UNIVERSITÀ DEGLI STUDI DI MILANO
Facoltà di Scienze e Tecnologie
Corso di Laurea in Informatica

UN'APPLICAZIONE WEB PER
DIMOSTRARE LE PROPRIETÀ DEGLI
ALGORITMI DI LABEL
PROPAGATION

Relatore: Prof. Paolo CERAVOLO

Correlatore: Dr. Samira MAGHOOL

Tesi di:
Andrei Georgiani TALPALARU
Matricola: 942824

Anno Accademico 2019-2020

Ringraziamenti

Ringrazio molto il Professore Paolo Ceravolo per avermi permesso di lavorare su questo progetto, da cui ho imparato tante nuove tecnologie su cui non ho avuto modo di mettere mano fino ad ora.

Ringrazio anche la mia famiglia, che mi ha sostenuto per tutto questo tempo.

Indice

Ringraziamenti	ii
1 Introduzione	1
2 Social Network Analysis (da aggiungere piu informazioni a ogni sezione)	2
2.1 Reti	2
2.2 Label Propagation	3
2.3 Algoritmi di Label Propagation	3
2.4 Belonging Coefficient	3
2.5 AVPRA (Agent-based Vector-label PPropagation Algorithm)	3
2.5.1 Propagazione basata su agenti	4
2.5.2 Aggiornamento sincrono	4
2.5.3 Inizializzazione	4
2.5.4 Funzione di aggiornamento	4
2.5.5 Iterazione Finale	5
3 Il sviluppo dell'applicazione	6
3.1 Applicazioni di visualizzazione delle reti sociali	6
3.1.1 Visualizzatore Reti Sociali con AVPRA	6
3.1.2 Infomap Network Navigator	7
3.2 Sviluppo	8
3.2.1 Funzionalita previste	9
3.2.1.1 Backend	9
3.2.1.2 Frontend	9
3.2.2 Sviluppo dell'applicazione web	10
3.2.2.1 Frontend	10
3.2.2.2 Backend	12
3.2.3 Presentazione applicazione	13
3.2.3.1 Pagina Principale	13
3.2.3.2 Esempio Colori	14

3.2.3.3	Pagina Caricamento Reti Sociali Personalizzate . . .	21
3.2.3.4	Esempio Personalizzato	23
4	Sviluppi Futuri	26
5	Conclusioni	27

Capitolo 1

Introduzione

Capitolo 2

Social Network Analysis (da aggiungere piu informazioni a ogni sezione)

La Social Network Analysis e l'analisi delle strutture sociali attraverso le reti e la teoria dei grafi. [3] Grazie ad una rappresentazione astratta, dove le connessioni tra certi elementi di un sistema, chiamati anche nodi o agenti (quando sono capaci di fare scelte o azioni attive), la SNA puo essere applicata a una varieta di domini. Il comportamento del sistema, in termini di comunicazione, propagazione ed evoluzione delle connessioni e catturato da metriche di SNA. [4]

2.1 Reti

Una rete e composta da due componenti di base: i nodi e gli archi. Usando questi due componenti si possono descrivere una multitudine di strutture e relazioni (per esempio connessioni sociali, connessioni in una rete di computer, reti elettriche ...).

I nodi sono le entita presenti nella rete e che possono avere proprieta interne (per esempio un valore che rappresenta il grado del nodo, o il peso o una posizione nella rete).

Gli archi sono le connessioni tra i nodi che possono avere a loro volta delle proprieta (per esempio un peso o un'altro tipo di relazione tra i nodi collegati alle estremita dell'arco). [3]

2.2 Label Propagation

Label Propagation rappresenta una famiglia di algoritmi usati per predire le etichette dei nodi di una rete. Differisce da altri algoritmi di Social Network Analysis siccome, invece di adottare una visione globale della rete, le decisioni prese dall'algoritmo sono fatte rispetto a proprietà locali, quindi gli algoritmi di Label Propagation adottano una visione locale della rete. [5]

L'idea generale è che le etichette assegnate inizialmente ai nodi sono propagate attraverso la rete seguendo gli archi. La propagazione di un'etichetta a un nodo si basa sulle etichette dei suoi vicini, tipicamente a un salto di distanza. Molteplici iterazioni permettono il raggiungimento di un'approssimazione del ottimo globale partendo da un ottimo locale. La complessità temporale varia tra tempo lineare ed esponenziale, dipendendo dalla densità della rete. L'approccio è naturalmente decentralizzato, con passaggi singoli su ogni nodo ad ogni iterazione. [4]

2.3 Algoritmi di Label Propagation

La scelta principale che differenzia i diversi algoritmi di Label Propagation è legata alla Update Rule. Alcune varianti della Update Rule che sono state usate nel tempo sono: il grado dei nodi, il coefficiente di clustering e la similarità strutturale.

La limitazione principale degli algoritmi precedenti era l'impossibilità di scegliere più di un'etichetta per ogni nodo.

Gli algoritmi sono stati generalizzati per permettere l'identificazione di comunità sovrapposte. Ogni etichetta viene pesata con un belonging coefficient associato al nodo. Tutti i belonging coefficient di un nodo sommano a 1.

2.4 Belonging Coefficient

Da scrivere

2.5 AVPRA (Agent-based Vector-label PPropagation Algorithm)

AVPRA è un algoritmo di Label Propagation basato su un'organizzazione ad agenti che permette di implementare la regola di update rispetto a più fattori, come per esempio le proprietà di dominio.

AVPRA differisce dagli altri algoritmi tramite il fatto che l'output dell'algoritmo è un vettore compatibile col formato di input usato in maggior parte degli algoritmi

di Machine Learning. Il vettore ha una lunghezza fissata per tutti gli nodi e include tutte le label della rete assieme al loro belonging coefficient basato sulle proprieta strutturali della rete.

Il belonging coefficient nel caso di AVPRA rappresenta una misura di diffusione/accumulo delle etichete dei nodi, dipendente dalla distanza rispetto ad altri nodi e la loro frequenza. [4]

2.5.1 Propagazione basata su agenti

Consideriamo che le etichete possono scorrere nella rete e gli agenti possono tenere conto della loro incidenza. Gli agenti sono semi-intelligenti e agiscono rispetto alla situazione riscontrata. Possono adottare diverse strategie nella ricezione o rifiuto di un'eticheta che scorre da un nodo vicino, in concordanza con certe condizioni. In questo modo, l'algoritmo puo controllare gli effetti globali, come per esempio la condizione di terminazione.

2.5.2 Aggiornamento sincrono

Per agire i conflitti nella propagazione degli aggiornamenti, in AVPRA gli agenti sono aggiornati in modo asincrono. Il vettore delle etichete di un certo nodo a tempo t sara ottenuto dai vettori delle etichete dei vicini a tempo $t - 1$. In questo modo tutti gli agenti possono aggiornarsi simultaneamente.

2.5.3 Inizializzazione

Ogni agente viene etichettato con un vettore di d dimensioni dove d e la cardinalita dell'insieme L , dove L e l'insieme delle etichete uniche nella rete.

2.5.4 Funzione di aggiornamento

La regola di aggiornamento mostra il modo in cui i vettori di etichete sono aggiornati ad ogni iterazione. In AVPRA, la funzione di aggiornamento e:

$$VL_i[l](t) = w_1 VL_i[l](t - 1) + w_2 \sum_{j \in \Gamma(i)} VL_j[l](t - 1) \quad (1)$$

Dove $VL_i[l](t)$ rappresenta il valore del belonging coefficient dell'eticheta l nel nodo n_i al tempo t , w_1 rappresenta il peso delle etichete correntemente assegnate al nodo n_i , w_2 rappresenta il peso delle etichete correntemente assegnate ai nodi $\Gamma(i)$ e $\Gamma(i)$ sono i vicini del nodo n_i .

2.5.5 Iterazione Finale

L'algoritmo AVPRA finisce quando il sistema raggiunge un'iterazione s nella quale i vettori di etichete sono stazionari. Cio implica che le variazioni dei belonging coefficient devono essere minori di un parametro p che viene chiamato negligibly threshold.

Capitolo 3

Il sviluppo di un applicazione per dimostrare le proprietà dell'algoritmo AVPRA

3.1 Applicazioni di visualizzazione delle reti sociali

3.1.1 Visualizzatore Reti Sociali con AVPRA

AVPRA è un'algoritmo di Label Propagation basato su Agenti. Il visualizzatore delle Reti Sociali per AVPRA è un'applicazione web che permette all'utente di eseguire e visualizzare i risultati dell'algoritmo in un modo interattivo nel browser web. [4]

I Pro

- Ha tanti parametri che permettono all'utente di controllare il comportamento dell'algoritmo e dell'input.
- Permette l'esportazione della rete e delle informazioni visibili sullo schermo in formato PNG.
- Permette di esportare le informazioni riguardanti il grafo e il output in formato JSON.
- Singoli nodi possono essere selezionati in modo da poter vedere più dettagli e misure riguardanti quel specifico nodo.
- Offre la possibilità di selezionare, comparare ed esportare la comparazione tra due nodi diversi della rete.

- Sopporta la visualizzazione di due viste diverse: Vector Label View (mostra in modo grafico la distribuzione delle label su ogni nodo) e Community View (mostra in modo grafico la divisione della rete in comunità basate sulla struttura.).
- Mostra graficamente la divisione in comunità raggruppando i nodi della stessa comunità più vicini uno all'altro.
- Permette all'utente di eseguire l'algoritmo per più iterazioni e di vedere graficamente l'evoluzione dei Vector Label nel tempo (feature specifica di AVPRA).
- La palette di colori utilizzata per i label nei Vector Label può essere modificata usando un'ulteriore file nella fase di caricamento della propria rete.
- Permette all'utente di selezionare un nodo specifico tramite una barra di ricerca dove si può inserire l'id del nodo.

I Contro

- Sopporta un solo formato di input (tre file CSV: Lista Archi, Vector Label Iniziali, Nomi Label) e due formati di output (un file pickled contenente i risultati di tutte le iterazioni dell'algoritmo e un file JSON contenente i dati della rete o la comparazione di due nodi diversi).
- I dataset dell'utente e i risultati delle esecuzioni dell'algoritmo su quei dataset sono tenute sul server dell'applicazione, anche se per poco tempo.
- Performance più scarse nel caso venga usato con reti molto grandi.

3.1.2 Infomap Network Navigator

Infomap è un'algoritmo di clustering su reti basato su Map Equation. Infomap Online è un'applicazione web che permette agli utenti di eseguire l'algoritmo Infomap nel browser web. [1]

Infomap Network Navigator è un'applicazione che permette di visualizzare i risultati dell'algoritmo in un modo interattivo. [2]

I Pro

- I dataset dell'utente e i risultati dell'esecuzione dell'algoritmo su quei dataset non sono mantenute sul server di Infomap Online.
- Ha tanti parametri per controllare il comportamento dell'algoritmo, dell'input, del output e dell'accuratezza.

- Soppoporta tanti formati di input (Link-List, Pajek, Bipartite, Multilayer, With inter-layer links, Without inter-layer links, States) e tanti formati di output (Physical, State-level output, Tree, FTree, Clu, Newick, JSON). Infomap riconosce automaticamente il formato dei file dal header del file.
- Permette di esportare la visualizzazione in formati SVG e PNG.
- Permette l'esportazione del file di output.
- Singoli nodi possono essere selezionati in modo da vedere piu dettagli e misure riguardati il specifico nodo.
- Permette all'utente di selezionare un nodo specifico tramite una barra di ricerca.
- Visually shows communities by grouping nodes or other sub-communities in a module (Map Equation algorithm specific visualization). Mostra graficamente il raggruppamento dei nodi in community e sotto-community attraverso un modulo (feature specifica Map Equation).

I Contro

- L'esportazione della visualizzazione non include nient'altro oltre la rete stessa.
- Non offre la possibilita di vedere il modo in cui due nodi diversi della rete siano correlati.
- Esiste un limite sulla dimensione dei file di input che si possono utilizzare.
- Non esiste un modo per personalizzare la paletta di colori che viene usata nella visualizzazione.

3.2 Sviluppo

Il progetto e un'applicazione web che e divisa in 2 parti:

- Frontend - la componente dell'applicazione con cui l'utente interagisce direttamente.
- Backend - la componente dell'applicazione con cui interagisce il frontend e che si occupa di esporre un'API che esegue le funzionalita richieste dal frontend.

3.2.1 Funzionalità previste

3.2.1.1 Backend

- Dev'essere capace di eseguire l'algoritmo AVPRA dato in Python.
- Deve esporre un'API che:
 - permette all'utilizzatore di eseguire l'algoritmo
 - permette all'utilizzatore di controllare i parametri dell'algoritmo come numero di iterazioni, negligibility threshold, interpretazione della rete (directed, undirected), modo di propagazione delle label (da predecessori, da successori) e formula dei pesi della regola di Update.
 - permette di ottenere i dettagli riguardanti un solo nodo ad una certa iterazione.
 - permette di ottenere i dettagli riguardanti la relazione tra due nodi ad una certa iterazione.
 - permette di esportare i risultati e i dettagli dell'esecuzione in un formato definito.
 - Deve permettere all'utente di caricare ed eseguire l'algoritmo sulla propria rete usando un formato definito.

3.2.1.2 Frontend

- Deve permettere all'utente di visualizzare in modo interattivo la situazione, alla iterazione corrente, della rete e dei suoi vector label.
- Deve implementare un modo in cui l'utente possa caricare la sua propria rete.
- Deve dare la possibilità all'utente di avere più informazioni riguardo lo stato corrente di un vector label di un nodo specifico.
- Deve permettere di avere informazioni riguardo l'iterazione corrente e l'iterazione finale (da definire) dell'algoritmo.
- Deve implementare un modo in cui si possa andare avanti e indietro tra le iterazioni in modo da vedere l'evoluzione dei vector label.
- Deve fare possibile la visualizzazione delle comunità associate alla rete.
- Deve permettere di selezionare 2 nodi e di vedere la situazione corrente di entrambi.

- Deve implementare un modo di esportare i file che si possono esportare attraverso l'API esposta dal backend.

3.2.2 Sviluppo dell'applicazione web

3.2.2.1 Frontend

Per lo sviluppo della parte Frontend dell'applicazione, contenente l'interfaccia utente e la comunicazione col backend, sono state utilizzate diverse tecnologie e librerie.

Le librerie e tecnologie che sono state utilizzate sono presentate di seguito.

React.js è una libreria di JavaScript che viene usata per creare in modo facile delle interfacce utente interattive. I componenti più importanti di questa libreria che sono utilizzati all'interno dell'applicazione sono:

- Componenti

I componenti sono le parti che permettono di organizzare l'interfaccia utente in React. Questi possono essere definiti in due modi: Classi Componente e Funzioni Componente. [6]

Le Funzioni Componente sono funzioni JavaScript che prendono in input dei dati (chiamati props) e ritornano degli elementi React che descrivono, tramite JSX, il modo in cui dovrebbe essere visualizzato sullo schermo. [7]

Le Classi Componente sono classi che estendono `React.Component` e che implementano un metodo `render` che restituisce un elemento React.

- Stato

Gli aggiornamenti della UI in JavaScript base dovevano accadere trovando l'elemento nel DOM e aggiornandolo manualmente. React, invece, aggiorna automaticamente la UI in base al stato interno del componente.

Prima dell'aggiunta dei Hooks, per modificare il stato interno di un componente era necessario che esso veniva scritto come Classe Componente. Dopo l'aggiunta del hook **useState**, il stato poteva essere utilizzato anche all'interno di una Funzione Componente. E necessario usare **useState** (Funzione Componente) o **setState** (Classe Componente) in modo che React capisca che la UI che usa il stato interno debba essere renderizzato per mostrare il valore aggiornato dello stato. [8]

- Hooks

Gli Hooks sono stati aggiunti per permettere di utilizzare stato e altre funzioni che esistevano nelle Classi Componente anche nelle Funzioni Componente [10]. Alcuni esempi di hooks maggiormente utilizzati sono:

- **useState** - hook che permette di aggiungere stato a funzioni componente.
- **useEffect** - hook che permette di aggiungere side effects in funzioni componente, il che significa poter far succedere qualcosa in base al cambiamento delle dipendenze dichiarate. Può essere usato anche per simulare i comportamenti `componentDidMount` e `componentDidUpdate` delle classi componenti.

All'interno dell'applicazione sono stati creati alcuni Hook Custom che hanno permesso di facilitare il processo di sviluppo.

I hook che sono stati implementati sono **useGUIControls** (che mette a disposizione le funzioni e i dati da utilizzare per i controlli delle pagine riguardanti la visualizzazione dei grafi), **usePolling** (che permette di chiamare un endpoint dell'API in modo ripetuto a intervalli regolari nel caso esso rispondeva con un codice di Not Ready) e **useSS** (che permette di fotografare il schermo e viene utilizzato nel caso in cui si preme il tasto di Screenshot sull'applicazione).

- **JSX**

JSX rappresenta un'estensione di JavaScript che permette di scrivere HTML direttamente all'interno di JavaScript. Usando Babel, JSX viene poi compilato in chiamate `React.createElement()`. Ciò significa che JSX può essere usato anche in strutture condizionali, per poter produrre risultati diversi rispetto a una decisione. [9]

Typescript è un'estensione di JavaScript che aggiunge una sintassi per i tipi e una migliore integrazione coi editor, aiutando in questo modo il sviluppatore a trovare velocemente gli errori dovuti ai tipi. [11]

I tipi di dati che TypeScript introduce sono:

- **number** - numero floating point con doppia precisione
- **string**
- **bigint**
- **boolean**
- **symbol**

- null
- undefined
- object
- void - tipo inteso come tipo di ritorno per le funzioni che non ritornano un valore
- $T[]$ - array mutabili
- $[T, T]$ - tuple di dimensione fissata, ma mutabili
- $(t: T) \rightarrow U$ - funzioni che prendono in input un tipo T e restituiscono un tipo U

Next.js è un framework di React che risolve maggior parte dei problemi che si incontrano quando si cerca di lavorare solo con la libreria React (da aggiungere più dettagli). Per framework si intende che Next.js gestisce in modo autonomo gli strumenti e le configurazioni che React necessita, offrendo però anche funzionalità aggiuntive. [13]

D3.js è una libreria per manipolazione di documenti in base a dati. D3.js permette di associare dati al DOM e costruire visualizzazioni per quei dati sull'applicazione che si sta creando. Nel caso presente D3.js viene utilizzato per costruire la visualizzazione del grafo e per aggiungere interattività ad esso. [14]

Katex è una libreria JavaScript usata per visualizzare formule matematiche sulla pagina. Nel caso presente, questa libreria è stata usata per poter mostrare all'utente la formula coi pesi che si stanno inserendo quando si usa la visualizzazione delle reti personalizzate. [15]

Tailwind CSS è un framework di CSS che offre classi di utilità che aiutano lo sviluppatore a evitare di scrivere CSS direttamente. Usando solo Tailwind CSS si può evitare di usare valori arbitrari per maggior parte delle misure, e quindi si arriva ad essere più consistenti col design dell'applicazione. Nell'applicazione presentata, questo framework è stato usato per fare tutto il layout del sito. [16]

3.2.2.2 Backend

Da descrivere l'API.

Per lo sviluppo della parte Backend dell'applicazione, contenente l'API che ha la capacità di rispondere alle richieste del frontend, sono state utilizzate diverse tecnologie e librerie.

Le librerie e tecnologie che sono state utilizzate sono presentate di seguito.

Flask è un microframework web basato su Python che è stato usato, nel caso presente, per il sviluppo del Backend come un REST API. [17]

Flask permette di creare un'API nel modo più semplice possibile, ed è stato scelto come tecnologia di Backend poichè l'algoritmo che viene usato per la processazione delle reti è scritto in Python, e quindi la scelta più facile per un linguaggio di Backend era Python.

Nel caso dell'esempio dei colori, il server di backend risponde puramente da file statici con la rete già processata. Nel caso degli esempi personali, i file necessari vengono caricati sul server, vengono processati, e fino alla scadenza della sessione, si può visualizzare la rete personalizzata.

Sympy è una libreria di Python per l'interpretazione di matematica simbolica. È stato usato per poter interpretare le formule per i pesi che vengono mandate dal Frontend nel caso di esempi personalizzati. [19]

CDlib è una libreria che aggrega tanti algoritmi di community detection. È stato utilizzato solamente per l'implementazione dell'algoritmo di community detection di Leiden. [20]

3.2.3 Presentazione applicazione

3.2.3.1 Pagina Principale

La pagina principale serve per permettere all'utente di scegliere tra due modi di diversi di provare e visualizzare i risultati dell'algoritmo. Ogni modo ha una piccola descrizione del come funziona e un tasto per andare alla pagina specifica per quella modalita. Ogni modalita sara descritta in dettaglio nelle sezioni successive.

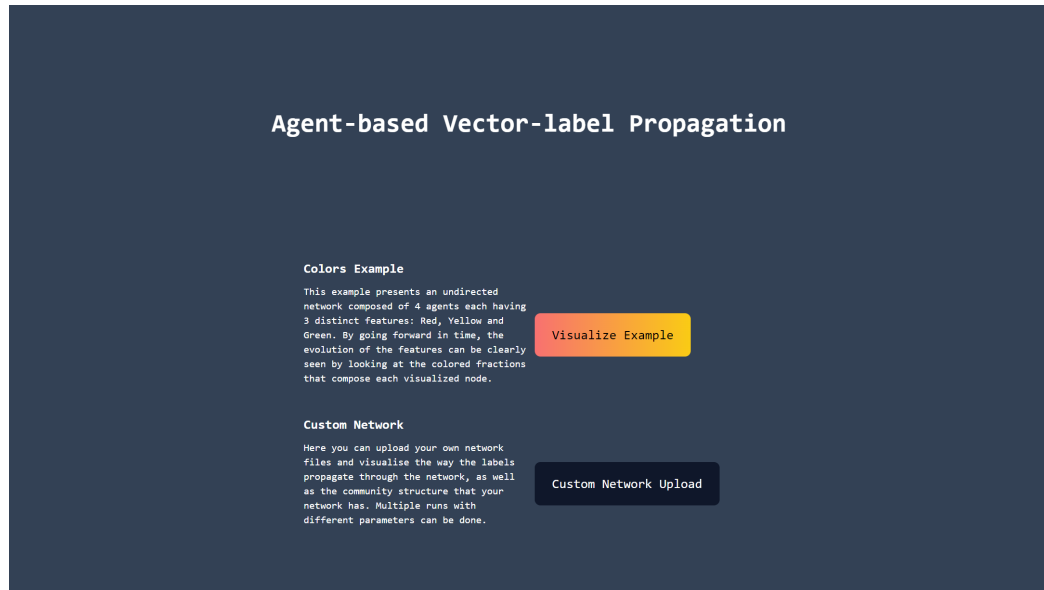


Figura 1: Pagina Principale

3.2.3.2 Esempio Colori

La pagina dell'esempio colori, che si può raggiungere premendo il bottone "Visualize Example" vicino a "Colors Example" sulla pagina principale, ha tutte le funzionalità dell'applicazione, dimostrate su un semplice esempio di grafo non diretto con 4 nodi e 3 archi. In questo esempio si può vedere il modo in cui le etichette si propagano ad ogni iterazione attraverso i colori.



Figura 2: Esempio Colori

I componenti che compongono l'interfaccia di visualizzazione della rete sono:

- **La bara di controllo**

Questo componente contiene tutte le modalita che si possono usare per controllare la visualizzazione della rete.

Iniziando da sinistra e andando verso destra, la bara di controllo contiene per prima cosa un rettangolo con dei tasti per controllare l'iterazione corrente che viene visualizzata, assieme alla iterazione che viene considerata finale dall'algoritmo (nel caso dell'esempio colori, il parametro p viene fissato a 0.1).

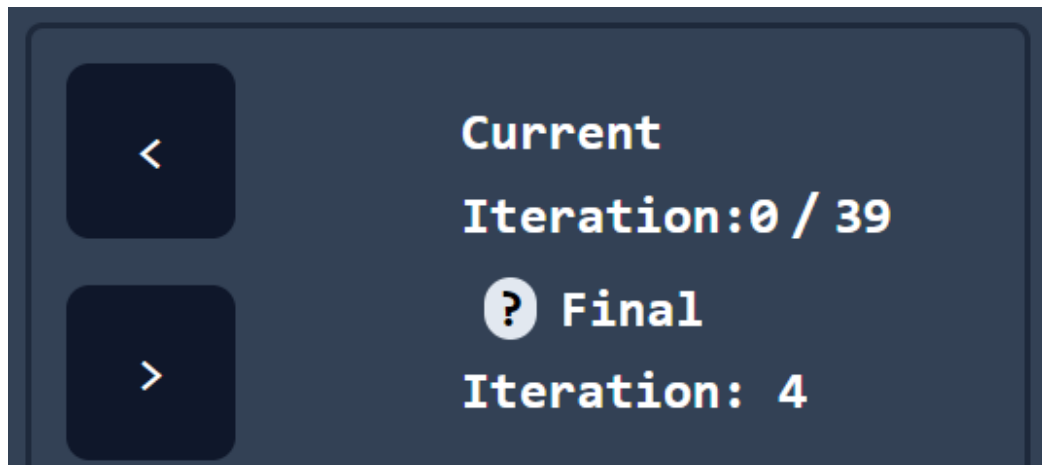


Figura 3: Controlli Iterazione

La seconda parte della barra di controllo viene utilizzata per cambiare il modo di selezione nodo e per cercare un nodo specifico nella rete. Se si preme il tasto per cambiare il modo di selezione nodo, si può selezionare un secondo nodo, in modo che, ulteriormente, si possono comparare i vettori di etichette dei due nodi specifici. La barra di ricerca di un nodo specifico diventa molto utile nel caso in cui la rete utilizzata ha molti nodi e quindi dove la selezione del nodo dalla rete diventa difficile.

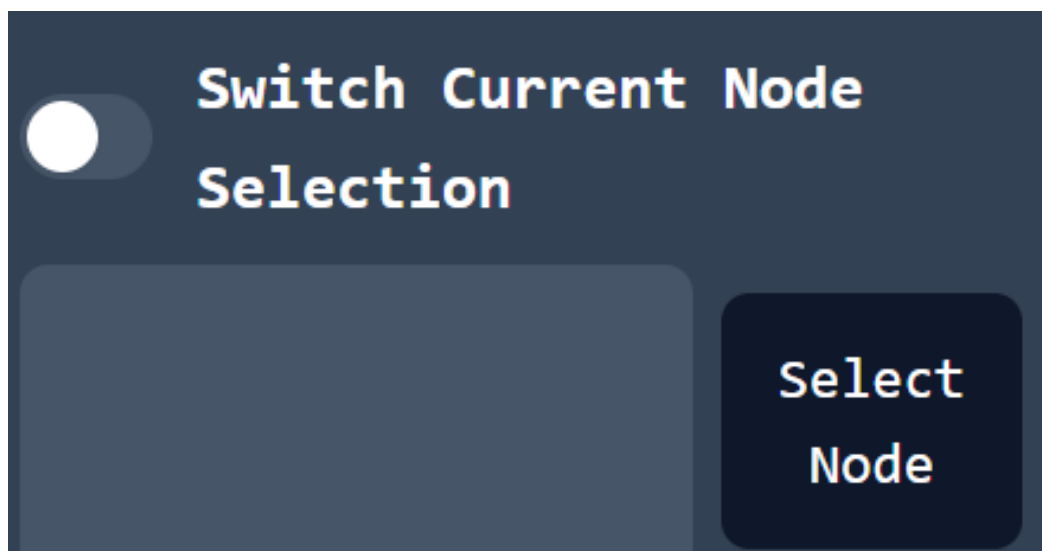


Figura 4: Controlli Selezione Nodo

L'ultima parte della barra di controllo contiene alcuni controlli per cambiare il layout di visualizzazione e alcuni tasti per: catturare la configurazione corrente dell'applicazione, scaricare il file dei risultati, scaricare il file contenente i dettagli del grafo e scaricare un file contenente la comparazione tra due nodi selezionati. Le opzioni per cambiare la configurazione dell'applicazione sono: Show VL View (che viene utilizzato per nascondere o per visualizzare la View che presenta graficamente la configurazione dei vettori d'etichete), Show Community View (che viene utilizzato per nascondere o per visualizzare la View che presenta graficamente la divisione in comunità della rete), Show Info Bar (che viene utilizzato per nascondere e far comparire la barra laterale che presenta le informazioni sulla rete, o sui nodi selezionati) e Show Ids (che viene utilizzato per nascondere o per visualizzare gli ID all'interno di ogni nodo in una View).

Formato Graph Details

Formato Comparazione nodi

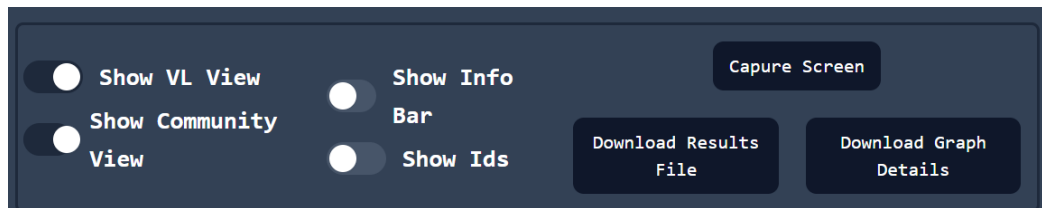


Figura 5: Controlli Vista

- **VL View**

La VL View rappresenta la vista che visualizza la rete dal punto di vista dei vettori delle etichete. Tutti i belonging coefficient sommano a 1 e quindi ogni eticheta viene rappresentata da un colore che occupa la percentuale specifica sul nodo (per esempio se un'eticheta su un nodo ha il belonging coefficient uguale a 0.5, allora il colore di quell'eticheta occuperà 50% del nodo nella VL View).

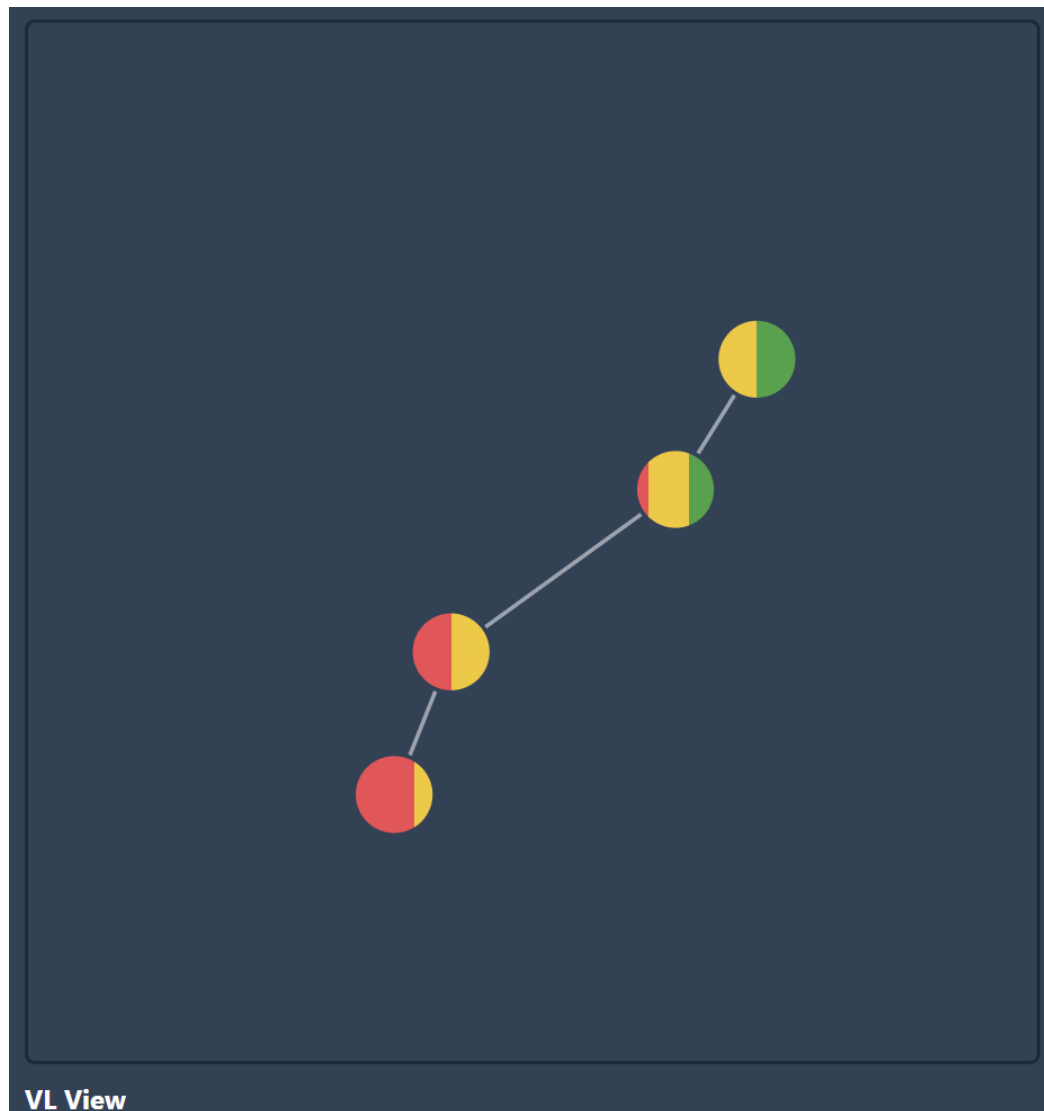


Figura 6: Vista Vettori di etichete

- **Community View**

La Community View rappresenta la vista che visualizza la rete dal punto di vista delle comunità che compongono la rete. I nodi che si trovano nella stessa comunità avranno lo stesso colore, in modo da identificare facilmente i nodi appartenenti alla stessa comunità dal punto di vista strutturale. La comunità di ogni nodo può essere conosciuta dalle informazioni specifiche di ogni nodo, quando esso viene selezionato.

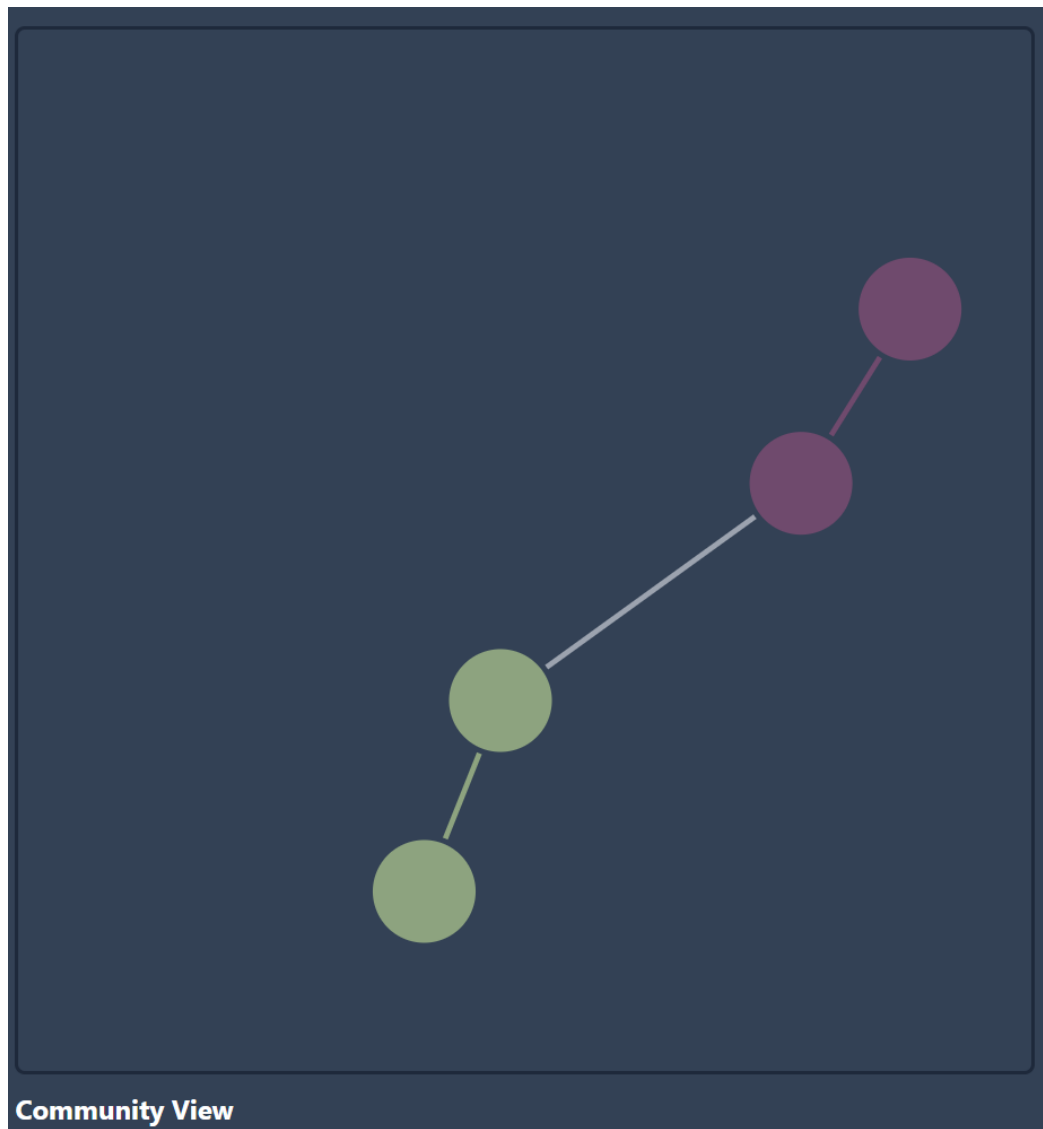


Figura 7: Vista Comunità

- **Info Bar**

La Info Bar visualizza, nel caso nessun nodo viene selezionato, le informazioni generali riguardanti la rete, come tipo di rete, numero di nodi, numero di archi e diametro. Nel caso in cui un nodo viene selezionato, la barra delle informazioni mostra il grado di quel nodo, la comunità appartenente, assieme al suo colore e il vettore delle etichette coi belonging coefficient di ogni etichetta, assieme al

colore dell'etichetta. Nel caso in cui due nodi vengono selezionati, ci saranno due bare di informazioni che mostrano le stesse informazioni di prima per entrambi i nodi.

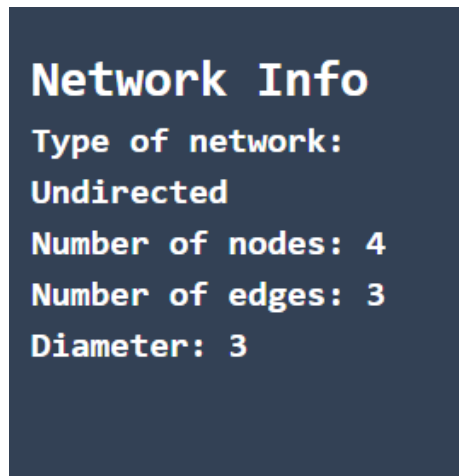


Figura 8: Barra Informazioni senza un nodo selezionato



Figura 9: Barra Informazioni con un solo nodo selezionato

Selected Node:	Selected
1	Node: 4
Properties	Properties
Degree: 2	Degree: 1
Community: ■ 0	Community: ■ 1
Vector Label	Vector Label
Red ■:	Red ■: 0.75
0.16666666666666666	Yellow ■:
Yellow ■: 0.5	0.25
Green ■:	Green ■: 0
0.3333333333333333	

Figura 10: Barra Informazioni con due nodi selezionati

3.2.3.3 Pagina Caricamento Reti Sociali Personalizzate

Nel caso della pagina di upload delle reti sociali personalizzate, l'utente può caricare i suoi dataset personali, formattati in un formato descritto nella pagina stessa, e visualizzare i risultati di AVPRA sulla sua rete.

L'utente ha la possibilità di personalizzare diverse opzioni riguardanti l'interpretazione dei dati e i parametri con cui l'algoritmo sarà eseguito.

I file riguardanti la rete che l'utente può caricare sull'applicazione sono i seguenti:

- Il file degli archi che dev'essere un file .csv col seguente formato:

$$\dots \text{nodo1} \text{ nodo2} \{ \text{peso} \} \dots \quad (2)$$

Questo formato mostra che il nodo1 è collegato al nodo2 e che l'arco ha il peso specificato. Il peso è opzionale e quindi il file può essere definito anche nel seguente modo:

$$\begin{array}{c}
\dots \\
nodo1 \ nodo2 \\
nodo2 \ nodo4 \\
\dots
\end{array} \tag{3}$$

- Il file dei vettori delle etichete con cui i nodi vengono inizializzati.

Questo file dev'essere un file .csv col seguente formato:

$$\begin{array}{c}
b_1(l_1); \ b_1(l_2); \dots; \ b_1(l_k) \\
b_2(l_1); \ b_2(l_2); \dots; \ b_2(l_k) \\
b_3(l_1); \ b_3(l_2); \dots; \ b_3(l_k) \\
\dots
\end{array} \tag{4}$$

In questo caso b_1, b_2, b_3, \dots sono le funzioni del belonging coefficient dei nodi $1, 2, 3, \dots$ e l_1, l_2, \dots, l_k sono le etichete della rete. Tutti i valori di ogni riga devono sommare a 1.

- Il file dei nomi delle etichete che dev'essere un file .csv col seguente formato:

$$l_1; \ l_2; \ l_3; \ l_4; \ \dots \tag{5}$$

- Il file dei colori con cui vengono rappresentate le etichete.

Questo file dev'essere un file .csv che ha il seguente formato:

$$hex_1; \ hex_2; \ \dots \tag{6}$$

In questo formato hex_1, hex_2, \dots sono i codici esadecimale dei colori con cui saranno rappresentate le etichete l_1, l_2, \dots nella visualizzazione.

Un'esempio del formato di questo file e:

$$\#433A3F; \ \#404A56; \ \#3D5A6C; \ \#58827E; \ \#659687; \ \dots \tag{7}$$

Le opzioni che l'utente puo scegliere sono le seguenti:

- Grafo diretto o non diretto

Quest'opzione permette all'utente di scegliere se il grafo costruito dal file degli archi viene interpretato come un grafo diretto o non diretto.

- Propagazione etichete dai predecessori o successori

Quest'opzione permette all'utente di scegliere se la propagazione delle etichete viene influenzata dagli nodi entranti o dagli nodi uscenti. Nel caso in cui il grafo viene interpretato come un grafo non diretto, quest'opzione non è disponibile.

- Negligibly Threshold

Quest'opzione permette all'utente di scegliere il valore del parametro negligibly threshold associato all'algoritmo, da cui dipende l'iterazione finale dell'algoritmo.

- Numero di iterazioni

Quest'opzione permette all'utente di scegliere il numero di iterazioni per cui l'algoritmo sarà eseguito.

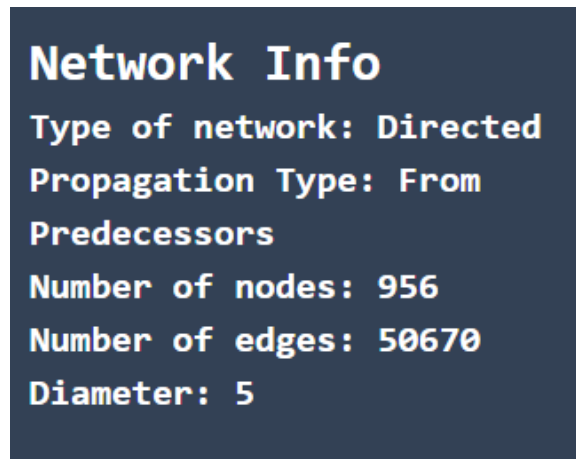
- Formula per il peso nella funzione di aggiornamento

Quest'opzione permette all'utente di scegliere una formula (dipendente dal numero di vicini del nodo) per i pesi nella funzione di aggiornamento dell'algoritmo AVPRA.

3.2.3.4 Esempio Personalizzato

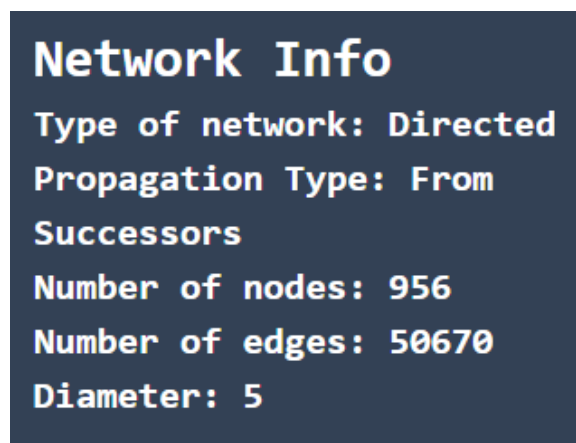
Nel caso dell'esempio personalizzato, l'interfaccia utente è praticamente uguale all'interfaccia presente nell'Esempio Colori, con qualche piccola differenza data dalle opzioni che l'utente ha scelto nella fase di caricamento della rete.

La bara di informazioni si comporta nello stesso modo della bara di informazioni dell'Esempio Colori, ma offre più informazioni generali riguardo la rete caricata sull'applicazione.



Network Info
Type of network: Directed
Propagation Type: From
Predecessors
Number of nodes: 956
Number of edges: 50670
Diameter: 5

Figura 11: Barra Informazioni Generale per una rete diretta e con propagazione delle etichete da predecessori



Network Info
Type of network: Directed
Propagation Type: From
Successors
Number of nodes: 956
Number of edges: 50670
Diameter: 5

Figura 12: Barra Informazioni Generale per una rete diretta e con propagazione delle etichete da successori

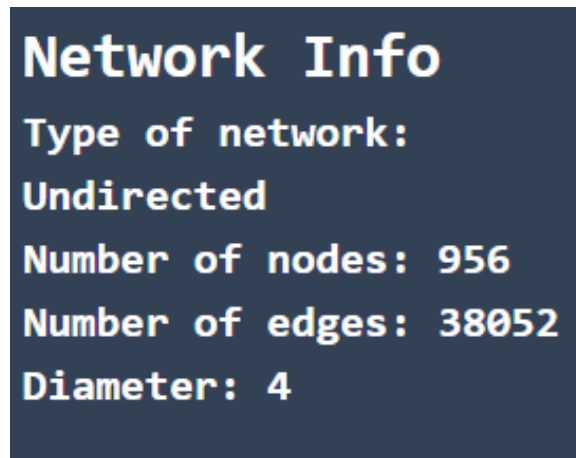


Figura 13: Barra Informazioni Generale per una rete non diretta

Figura 14: Barra Informazioni Senza Nodo Selezionato

La bara di controllo subisce modifiche solo nella sezione di Controlli Iterazione. In questo caso, il numero massimo delle iterazioni viene sostituito col numero di iterazioni scelto dall'utente durante il caricamento della rete, e l'iterazione finale cambia in modo che il numero d'ordine mostrato sia l'iterazione dove avviene la condizione di terminazione.

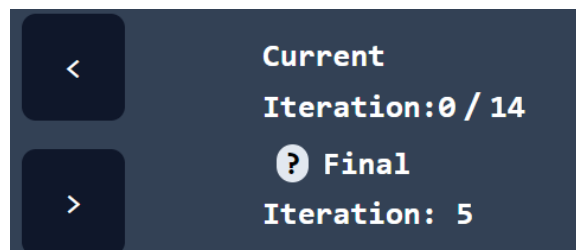


Figura 15: Controlli Iterazione per reti personalizzate

Capitolo 4

Sviluppi Futuri

Discussione riguardo il deployment dell'applicazione.

Capitolo 5

Conclusioni

Bibliografia

- [1] <https://www.mapequation.org/infomap/>
- [2] <https://www.mapequation.org/navigator/>
- [3] <https://towardsdatascience.com/social-network-analysis-from-theory-to-applications-with-python-d12e9a34c2c7>
- [4] Valerio Bellandi, Paolo Ceravolo, Ernesto Damiani, and Samira Maghool: Agent-based Vector-label Propagation for Explaining Social Network Structures. In: Springer Verlag (Lecture Notes in Communications in Computer and Information Science (CCIS)) (2022)
- [5] Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical review E 76(3), 036106 (2007)
- [6] <https://it.reactjs.org/docs/react-component.html>
- [7] <https://it.reactjs.org/docs/components-and-props.html>
- [8] <https://it.reactjs.org/docs/state-and-lifecycle.html>
- [9] <https://it.reactjs.org/docs/introducing-jsx.html>
- [10] <https://it.reactjs.org/docs/hooks-intro.html>
- [11] <https://www.typescriptlang.org/>
- [12] <https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes-func.html>
- [13] <https://nextjs.org/learn/foundations/about-nextjs/what-is-nextjs>
- [14] <https://d3js.org/>
- [15] <https://www.npmjs.com/package/katex>

- [16] <https://tailwindcss.com/>
- [17] <https://flask.palletsprojects.com/en/2.1.x/foreword/>
- [18] <https://flask.palletsprojects.com/en/2.1.x/>
- [19] <https://www.sympy.org/en/index.html>
- [20] <https://cdlib.readthedocs.io/en/latest/>