Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

Evoluzione dei contenuti e disallineamento su piattaforme di editing collaborativo

Relatore:
Dott. Angelo Di Iorio

Presentata da:
Damiano Bellucci

Sessione IV
Anno Accademico 2017/2018

Indice

Elenco delle figure

Capitolo 1

Introduzione

L'obiettivo di questa tesi è studiare l'evoluzione dei contenuti nelle piattaforme di editing collaborativo e proporre diversi meccanismi per analizzarla.

Le piattaforme di editing collaborativo sono piattaforme per la gestione di contenuti in cui lettori e scrittori sono figure sovrapposte cooperanti, ad esempio Wikipedia, che è la piattaforma presa in esame. Per studiare l'evoluzione dei contenuti si propone un modello in cui si analizzano le situazioni di disallineamento tra conoscenza del mondo reale e conoscenza espressa dai contenuti della piattaforma. Con disallineamento si intende una situazione in cui in una finestra temporale (timespan) la conoscenza del mondo reale riguardo un tema presente nella piattaforma è superiore a quella espressa nel contenuto della piattaforma riguardante quel tema. Le situazioni di disallineamento sono interessanti in quanto sono quelle che precedono l'evoluzione dei contenuti nella piattaforma, che porta ad un ri-allineamento con la conoscenza del mondo reale.

E' stata scelta Wikipedia come piattaforma di riferimento in quanto è un'enciclopedia online a contenuto libero, collaborativa, multilingue e gratuita, quindi una piattaforma di editing collaborativo su larga scala in cui chiunque può contribuire utilizzando il suo sistema di modifica e pubblicazione.

I contenuti in Wikipedia sono organizzati in pagine, per questo motivo per studiare l'evoluzione dei contenuti su Wikipedia bisogna studiare l'evoluzione delle sue pagine. La traccia dell'evoluzione di una pagina Wikipedia è il suo storico revisioni, un tassello fondamentale per il modello proposto. Infatti la vita di una pagina in Wikipedia consiste in una o più revisioni in successione in cui la prima revisione corrisponde alla creazione della pagina e l'ultima allo stato attuale della pagina. Ogni revisione corrisponde ad un edit da parte di un utente della piattaforma Wikipedia e quindi ad uno stato in cui è transitata la pagina.

In diversi lavori precedenti si ricavano indicatori dallo storico delle revisioni per studiare la loro relazione con la qualità dell'informazione delle pagine. Sebbene gli scopi di questo modello siano diversi, questi indicatori sono rappresentativi dell'evoluzione di una pagina e quindi efficaci anche in questa tesi. Di conseguenza, dallo storico revisioni si

ricavano indicatori quali edits e comments (revisioni del talk, pagine dedicate agli utenti per discutere sulle modifiche della pagina) che assieme all'indicatore views costituiscono gli indicatori di potenziale disallineamento che verranno usati come filtri per individuare tra una collezione di pagine quelle in evoluzione in un periodo di tempo (timespan). Oltre a edits e comments, dallo storico revisioni si ricavano indicatori ulteriori quali età della pagina, minor edits, authors (autori distinti) che assieme ai primi due saranno utili per studiare l'evoluzione di una pagina.

L'implementazione software del modello proposto è Monitor Wiki (MoW), uno script parametrico command line. MoW permette di selezionare un host (installazione di MediaWiki) su cui lavorare (es. en.wikipedia.org, it.wikipedia.org, ecc...), selezionare le pagine per categoria o per titolo, filtrare le pagine in base al timespan e in base ai parametri sui filtri degli indicatori di potenziale disallineamento, raccogliere dati sugli indicatori in un determinato timespan.

L'analisi svolta riguarda tre casi rappresentativi di situazioni comuni, il cui obiettivo è stato mostrare esperimenti preliminari per testare e mostrare esempi di utilizzo del modello e del relativo software, discutendone i risultati principali. In particolare si è presentato un caso di pagine in alta evoluzione confrontando i risultati in due diversi anni, dove si è potuto vedere come l'evoluzione cresca nel tempo e come l'indicatore comments influisca sugli indicatori di edits, minor edits e authors. Il secondo caso riguarda una situazione di allineamento dove si può vedere come i valori degli indicatori di evoluzione edits, minor edits e comments risultano bassi e stabili nel tempo, così come per le views. Infine, si presenta un caso in cui si analizza l'impatto di un evento nel mondo reale sull'evoluzione dei contenuti in Wikipedia.

La tesi è strutturata nel seguente modo:

- il capitolo 2 descrive l'idea di editing collaborativo, la piattaforma Wikipedia e si parla dei lavori correlati che usano l'evoluzione dei contenuti di Wikipedia
- il capitolo 3 descrive il modello proposto per studiare l'evoluzione dei contenuti entrando nei dettagli dei concetti fondamentali quali finestre temporali, indicatori e filtri;
- il capitolo 4 descrive il software MoW che implementa il modello proposto nel capitolo 3, focalizzandosi sull'input/output di ogni operazione;
- descrive l'implementazione del software MoW, entrando nei dettagli delle scelte implementative, dell'architettura e della codifica di alcune delle componenti;
- il capitolo 6 descrive esempi di utilizzo del modello e del relativo software, discutendo i risultati ottenuti;
- il capitolo 7 descrive i possibili sviluppi futuri.

Capitolo 2

Background

L'obiettivo di questa tesi è studiare l'evoluzione dei contenuti nelle piattaforme di editing collaborativo e proporre diversi meccanismi per analizzarla. Questi meccanismi si basano su criteri per cui è necessario capire quali sono stati tutti gli altri lavori che cercano di estrarre informazioni dall'evoluzione dei contenuti nelle piattaforme di editing collaborativo. In questo capitolo si da una visione generale di questi lavori, cercando di collegarli. Si introdurrà l'idea di editing collaborativo e di conseguenza si parlerà di Wikipedia, che è la piattaforma di editing collaborativo in esame in questo lavoro.

2.1 Editing collaborativo

L'editing collaborativo è l'opera di un gruppo di persone che produce testi grazie ai singoli contributi dei membri. Si adotta questa pratica soprattutto nella scrittura di documenti di testo o di codice sorgente del software. Una delle caratteristiche dell'editing collaborativo è la sua asincronicità, cioè il fatto che i membri del gruppo non devono per forza riunirsi per lavorare insieme. La figura 2.1 mostra uno schema di editing collaborativo: 3 utenti visionano e modificano la stessa pagina. Le modifiche apportate vengono combinate per ottenere una nuova versione della pagina, che sarà il risultato del contributo dei 3 utenti. Le modifiche possono avvenire in tempi diversi e verranno combinate ogni volta che un utente esegue una modifica.

Oggi per gestire l'editing collaborativo si usano dei software: i più usati per la scrittura di documenti sono le Wiki, mentre per il codice sorgente del software il controllo versione. Anche la maggior parte degli editor di testo permettono di aggiungere annotazioni o commenti a margine, permettendo così a molte persone di lavorare sullo stesso documento, su cui ciascuno può apporre le proprie modifiche.

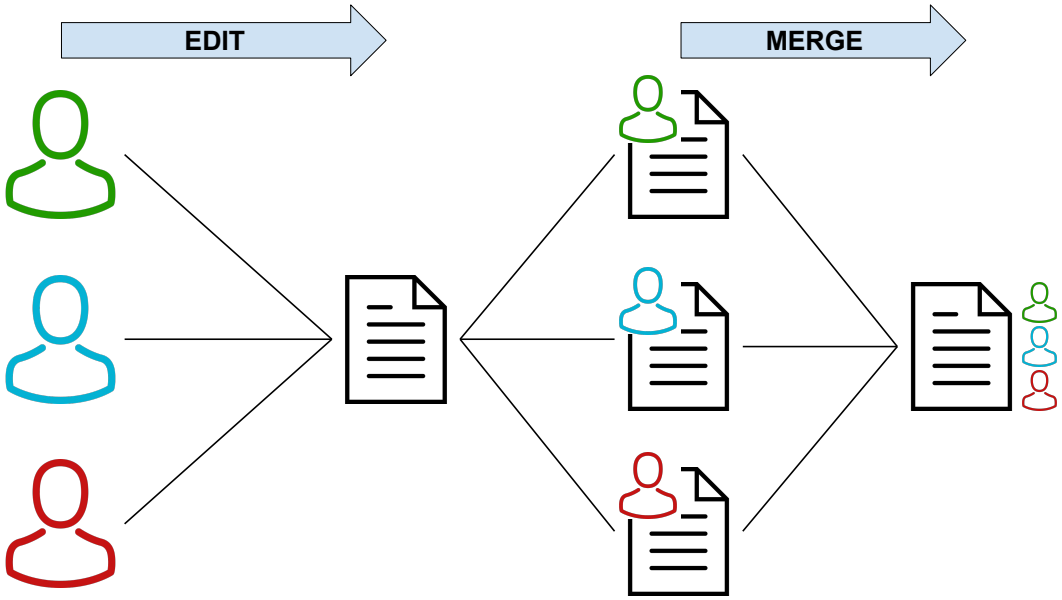

Figura 2.1: Schema editing collaborativo

Software Wiki

Wiki è un'applicazione web che permette la creazione, la modifica e l'illustrazione collaborativa di pagine all'interno di un sito web. Wiki è dunque un software collaborativo che in genere utilizza un linguaggio di markup semplificato o un editor di testo online. Il risultato è una raccolta di documenti ipertestuali che viene aggiornata dai suoi stessi utilizzatori e i cui contenuti sono sviluppati in collaborazione da tutti coloro che vi hanno accesso (contenuto generato dagli utenti), memorizzati normalmente su una base di dati o un repository. La modifica dei contenuti è aperta. Questo vuol dire che il testo può essere modificato da tutti gli utenti (a volte soltanto se registrati, altre volte anche anonimi) contribuendo nell'accrescimento dei contenuti, ma anche modificando ciò che hanno scritto gli autori precedenti. Ogni modifica è registrata in una cronologia che permette in caso di necessità di riportare il testo alla versione precedente (rollback); lo scopo è quello di condividere, scambiare, immagazzinare e ottimizzare le informazioni in modo collaborativo.

Il **motore Wiki** è il sistema software di tipo collaborativo su cui gira un sistema Wiki. La sua implementazione consiste generalmente in un programma installato su diversi server che gestiscono un contenuto solitamente memorizzato in un database relazionale. Un esempio è MediaWiki, il motore di molte Wiki tra cui le diverse istanze di Wikipedia (capitolo 5.1 per i dettagli).

Un wiki permette di scrivere collettivamente dei documenti in un semplice linguaggio di marcatura (markup) usando un web browser. Una singola pagina in un wiki è chiamata **pagina wiki**, mentre l'insieme delle pagine è chiamato "**il wiki**". Una caratteristica distintiva della tecnologia wiki è la facilità con cui le pagine possono essere create e aggiornate. Generalmente, non esiste una verifica preventiva sulle modifiche, e la maggior parte dei wiki è aperta a tutti gli utenti e la registrazione non è sempre richiesta.

Per quanto riguarda il **controllo delle modifiche**, molti wiki pubblici evitano le procedure di una registrazione obbligatoria, tuttavia molti dei maggiori motori wiki (incluso MediaWiki, usato anche in Wikipedia) forniscono metodi per limitare l'accesso in scrittura. Alcuni motori wiki permettono che utenti singoli siano interdetti dalla scrittura mediante il blocco del loro particolare indirizzo IP o, se disponibile, del loro username. Una comune difesa contro i vandali è permettere di cancellare e modificare quante pagine desiderano, in quanto possono essere facilmente tracciati e annullati nei loro atti. Inoltre, alcuni wiki hanno una base di dati che può essere impostato in modalità di sola sola lettura, mentre alcuni impongono la regola per cui solo utenti che si siano registrati prima di una certa data possano continuare a scrivere. In casi estremi molti wiki forniscono pagine che possono essere protette dalla modifica. Le pagine protette in Wikipedia, ad esempio, possono essere solo modificate dagli amministratori, che possono anche revocare la protezione. La protezione è considerato come violazione alla filosofia di base del Wiki e, quindi, è spesso evitato.

2.2 Wikipedia

Wikipedia è un esempio di piattaforma di editing collaborativo su larga scala basata sul software Wiki MediaWiki (capitolo 5.1 per i dettagli). E' un'enciclopedia online a contenuto libero, collaborativa, multilingue e gratuita, nata nel 2001. Essendo un'enciclopedia collaborativa, chiunque può contribuire utilizzando il suo sistema di modifica e pubblicazione e non c'è nessun controllo preventivo sul contenuto inviato. L'attendibilità dei contenuti si basa sul diritto di citazione delle fonti di origine delle informazioni. Da queste caratteristiche su cui si fonda Wikipedia nascono interrogativi. In particolare ci si chiede perché le persone dovrebbero preoccuparsi a contribuire al suo accrescimento e mantenimento o fidarsi di usare il servizio e perché quest'ultimo non sfocia nell'anarchia. I suddetti dubbi riguardano la qualità dell'informazione di Wikipedia, che da questo momento verrà chiamata IQ. Ci sono diversi modi in cui Wikipedia tutela la IQ : i ruoli dei contribuitori, le pagine talk e gli articoli in primo piano (FA).

Contribuitori

I contribuitori si possono identificare in diverse figure con diversi ruoli, che possono essere ricoperti anche da bot che svolgono compiti in automatico. Sia persone che programmi

vengono identificati sotto il nome di agenti.

- agenti editori : aggiungono nuovi contenuti a Wikipedia;
- agenti IQA : i garanti della IQ che si occupano di sistemare le situazioni causate da vandalismo, mantenere l'ordine nella community e gestire le norme IQ;
- agenti ambientali: il cambiamento del mondo reale e della conoscenza umana può essere considerato come un agente che comporta obsolescenza e quindi invalidare contenuti.

Pagina Talk

Ogni pagina in Wikipedia ha la sua "pagina talk" corrispondente (figura 2.2).

Figura 2.2: Pagina talk dell'articolo Computer Science

E' uno spazio dedicato alla comunicazione tra i contribuitori per la gestione e il mantenimento della pagina. Sono utilizzate spesso dagli agenti IQA per comunicare con gli autori della pagina e fornire riscontri sulla sua qualità. La pagina talk è a tutti gli effetti una pagina Wikipedia. Ogni sezione della pagina talk è relativa ad una discussione sulla pagina (esempio figura 2.3).

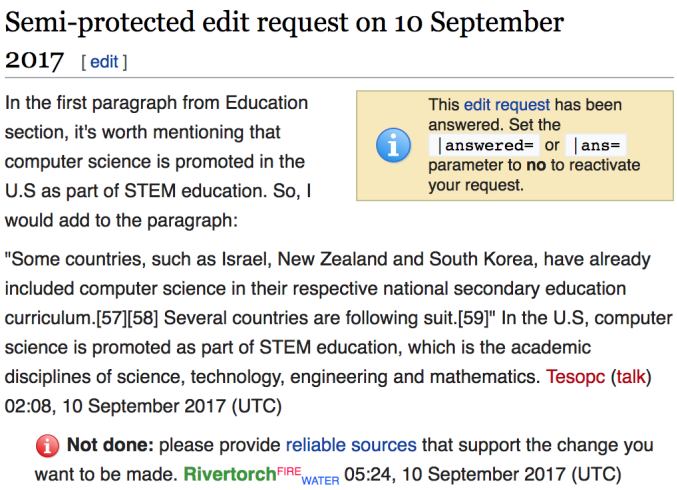

Figura 2.3: esempio di sezione del talk della pagina Computer Science

Articoli in primo piano

Gli articoli in primo piano (FA) sono gli articoli che si ritengono i migliori dalla comunità Wikipedia per IQ (un articolo è associato ad una pagina). Essi vengono prima candidati FA per poi essere sottoposti a processi di revisione che ne attestino la IQ secondo i criteri di qualità di un FA, cioè la buona scrittura, comprensibilità, accuratezza e verificabilità, neutralità, stabilità, stile appropriato, comprensività, il focus sul tema dell'articolo.

IQ Standard	Description
Well-written	Its prose is engaging, even brilliant, and of a professional standard
Comprehensive	It neglects no major facts or details and places the subject in context
Well-researched (Accurate& Verifiable)	It is a thorough and representative survey of the relevant literature on the topic. Claims are verifiable against high-quality reliable sources and are supported by inline citations where appropriate
Neutral	It presents views fairly and without bias
Stable	It is not subject to ongoing edit wars and its content does not change significantly from day to day, except in response to the featured article process
Appropriate Style	It follows the style guidelines, including the provision of a lead, appropriate structure and consistent citations
Comprehensive	It has images that follow the image use policy and other media where appropriate, with succinct captions, and acceptable copyright status
Focus	It stays focused on the main topic without going into unnecessary detail

Figura 2.4: Criteri IQ pagina FA da [9]

Uno studio del 2007 [6], dove si fa un'analisi della qualità e del processo di creazione dei contenuti usato in Wikipedia, afferma come la IQ è migliore di quanto ci si possa aspettare, a dispetto degli scetticismi derivanti dai principi alla base di Wikipedia. Infatti, i FA sono usati come benchmark per la IQ delle pagine. Lo standard di qualità dei FA seppur non essendo l'ideale come meccanismo per assicurare la IQ risulta essere rigoroso. Inoltre le pagine discussione vengono spesso utilizzate attivamente come luogo di collaborazione e scambio di opinioni sulla IQ della pagina. Sebbene chiunque possa partecipare, i risultati dei contributi vengono esaminati e discussi.

2.2.1 Storico revisioni

La vita di una pagina in Wikipedia consiste in una o più revisioni in successione in cui la prima revisione corrisponde alla creazione della pagina e l'ultima al suo stato attuale. Di conseguenza la vita di una pagina può essere vista come una sequenza di stati in cui la prima revisione corrisponde alla creazione della pagina e l'ultima alla sua versione attuale. Ogni revisione corrisponde ad un edit da parte di un utente della piattaforma Wikipedia e quindi ad uno stato in cui è transitata la pagina (figura 2.5).

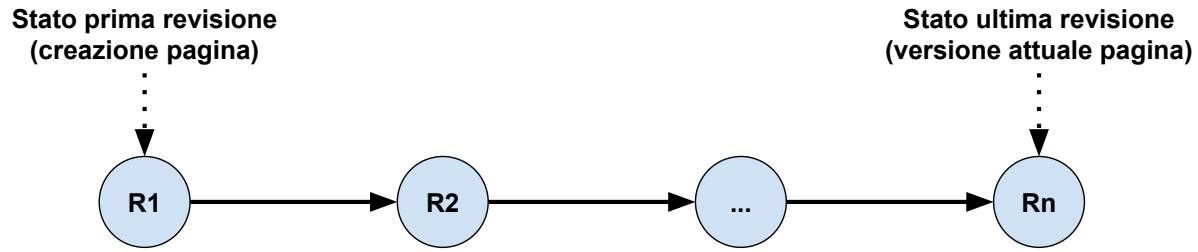

Figura 2.5: Storico revisioni

Ogni revisione è caratterizzata da:

- ID revisione
- Marcatura temporale (timestamp): denota la data in cui è avvenuta la revisione
- ID revisione precedente: ogni revisione ha un riferimento alla revisione precedente.
- Autore: autore che ha eseguito la revisione alla pagina
- Commento: commento (facoltativo) dell'autore sulla revisione apportata alla pagina.
- dimensione pagina: dimensione in byte della pagina.
- Importanza: le revisioni che non comportano grandi cambiamento nella pagina possono essere contrassegnate come "m" (m in grassetto nella figura 2.6) dall'autore.

E' di fondamentale importanza per questo studio lo storico revisioni di una pagina. Infatti lo storico revisioni è un log (traccia) dell'evoluzione dei contenuti per una pagina. Generalizzando sull'insieme di pagine in Wikipedia, è una traccia dell'evoluzione dei contenuti su Wikipedia. La marcatura temporale ci permette di distribuire le revisioni di una pagina nel tempo (figura 2.5).

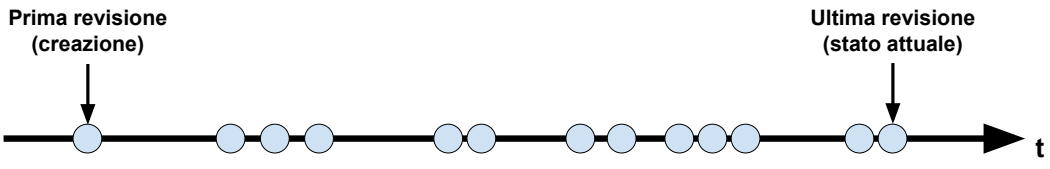

Figura 2.6: Storico revisioni e linea temporale

Per studiare l'evoluzione che ha avuto una pagina nel corso del tempo abbiamo bisogno del suo storico delle revisioni, in quanto fornisce tutte le informazioni e tutti i riferimenti degli stati in cui è transitata la pagina nel corso della sua vita. L'ID della revisione è importante perché è un riferimento diretto alla pagina nello stato in cui era per quella revisione. Tramite esso si possono studiare le componenti strutturali (links,sezioni, ecc...) della pagina per la revisione a cui fa riferimento l'ID.

$$\textbf{Evoluzione dei contenuti} = \textbf{Storico revisioni} \tag{2.1}$$

La figura 2.8 fa riferimento ad una porzione dello storico revisioni della pagina Computer Science in en.Wikipedia.org, che può essere visionata da un utente cliccando "View history" (figura 2.7) nella pagina stessa. Un'altra vista dello storico revisioni della pagina (formato JSON) è mostrato nel listing 2.1.

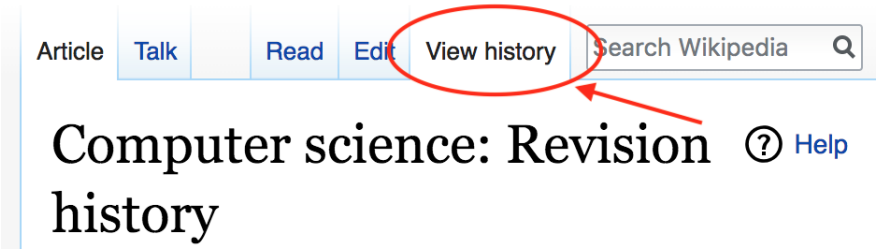

Figura 2.7: Pagina storia revisioni della pagina Computer Science

Figura 2.8: Storia revisioni della pagina Computer Science

```
{
    "pageid": 5323,
    "title": "Computer science",
    "revisions": [
        {
            "revid": 883569917,
            "parentid": 882625498,
            "minor": false,
            "user": "WilliamHamiltonijk",
            "timestamp": "2019-02-16T05:36:57Z",
            "comment": ""
        },
        {
            "revid": 882625498,
            "parentid": 882625421,
            "minor": true,
            "user": "Shadowssettle",
            "timestamp": "2019-02-10T10:49:48Z",
            "comment": "/* Challenges */ ...
        }
    ]
}
```

Listing 2.1: Esempio storico revisioni formato JSON

2.3 Lavori correlati

In questa tesi si vuole studiare l'evoluzione dei contenuti su piattaforme di editing collaborativo, in particolare su Wikipedia. Lo storico revisioni gioca un ruolo fondamentale in questo, in quanto come spiegato nel capitolo 2.2.1 tiene traccia della vita di una pagina e quindi della sua evoluzione nel tempo, proprio quello che ci interessa studiare.

In diversi lavori, sono stati ricavati indicatori dallo storico revisioni (quindi indicatori dell'evoluzione dei contenuti) per studiare la relazione tra questi e la IQ di una pagina.

Per quanto riguarda il concetto di IQ, Strong, Lee e Wang [4] [7] definiscono un'informazione di qualità come un'informazione "idonea all'uso" per chi ne usufruisce, invece Marshak [3] la definisce come "quanto accuratamente una informazione rappresenta un particolare evento" mentre per Eppler [1] significa "soddisfare le aspettative del fruitore e soddisfare i requisiti dell'attività". Dall'esigenza di valutare e misurare la IQ sono state formalizzate diverse metodologie al riguardo, tra cui AIMQ [2] che costituisce un

modello di base di carattere generale per la valutazione della IQ, utilizzabile in svariati contesti.

Gli indicatori derivanti dallo storico revisioni di cui si fa uso in questi lavori e su cui si focalizza l'attenzione di questa tesi sono:

- edits
- authors (autori unici)
- indicatori strutturali
  - sezioni
  - links interni
  - links esterni
- età della pagina
- comments (modifiche al talk della pagina)

In [8] si sono studiati i fattori che influiscono sulla IQ degli articoli di Wikipedia durante il processo di miglioramento della IQ dell'articolo. Si sono proposte metriche da quattro aspetti, comprese le caratteristiche linguistiche, strutturali (links interni ed esterni, sezioni, ecc...), storiche (edits, informazioni sugli editori, ecc...) e reputazionali. Si è trovato che le caratteristiche linguistiche sono più importanti negli stadi di qualità inferiore dell'articolo, e le caratteristiche strutturali (links interni ed esterni, sezioni, ecc...), insieme alle caratteristiche storiche (edits, autori, età della pagina), sono diventate più importanti mentre la qualità degli articoli è migliorata.

Un lavoro interessante è [9], dove si sono esaminate 50 milioni di modifiche apportate a 1,5 milioni di articoli di Wikipedia in lingua inglese e si è trovato che il gli articoli di alta qualità si distinguono per il notevole aumento del numero di edits, authors (autori unici) e l'intensità del comportamento cooperativo (comments) rispetto ad altri articoli di visibilità ed età simili (età della pagina).

In [10] si sono studiati i modi in cui i consumatori di informazioni valutano la IQ dei contenuti in un ambiente di editing collaborativo, in questo caso Wikipedia. L'analisi mostra che gli attributi che più spesso hanno aiutato gli utenti a decidere sulla IQ sono: attributi come quantità di informazioni, soddisfazione sul contenuto e link esterni. Attributi come gli edits e il numero di autori hanno ricevuto risultati contraddittori: pochi edits/autori e molti edits/autori sono stati entrambi designati come attributi di alta IQ.

Un modello per lo studio della IQ in Wikipedia viene proposto in [5]. Lo storico revisioni di una pagina in questo caso viene usato per ricavarne dimensioni IQ quali l'autorevolezza, la verificabilità, la completezza e consistenza delle pagine tramite le misure di indicatori quali edits, authors, links interni ed esterni, età della pagina.

Un indicatore importante di cui si fa uso in questa tesi e che non è ricavato dallo storico revisioni delle pagine è quello delle views delle pagine Wikipedia. Questo non è un indicatore di evoluzione dei contenuti, ma è interessante in quanto ci da informazioni sullo stato di attività di una pagina (cioè il livello di interazione degli utenti). In [11] si studia la relazione tra le views di una pagina e i trend di ricerca nel web. Si è constatato come la frequenza di una parola chiave di ricerca web generalmente riflette il grado di interesse pubblico in un particolare argomento e che le parole chiave di ricerca frequenti presentano correlazioni notevolmente elevate con le visualizzazioni di pagine di Wikipedia. Ciò suggerisce che le visualizzazioni di pagine di Wikipedia possono essere uno strumento efficace per determinare le tendenze della ricerca web globale più popolari. Questo sarà utile in questa tesi in quanto si può supporre che le views siano un indizio sull'evoluzione dei contenuti della pagina, in quanto riflettono la sua popolarità e di conseguenza è un'informazione sulle interazioni degli utenti.

Capitolo 3

Un modello per studiare l'evoluzione dei contenuti e il disallineamento della conosocenza

In questo capitolo si vuole proporre un modello per studiare l'evoluzione dei contenuti in una generica piattaforma di editing collaborativo, analizzando le situazioni di disallineamento tra conoscenza del mondo reale e conoscenza espressa nei contenuti della piattaforma. Si prenderà come piattaforma di riferimento Wikipedia, di cui si è parlato nel capitolo 2. Per farlo si introduce il concetto di finestra temporale, che assieme allo storico revisioni (discusso nel capitolo 2.2.1) e ai suoi indicatori (alcuni anticipati nel capitolo 2.3) e relativi filtri di cui si parlerà in questo capitolo, sono i mattoni fondamentali nella costruzione di questo modello.

3.1 Finestre temporali

Una finestra temporale (timespan) è una porzione della linea temporale, delimitata da un estremo destro ed un estremo sinistro. La figura 3.1 mostra una rappresentazione grafica di finestra temporale che sarà utilizzata nel resto del capitolo.

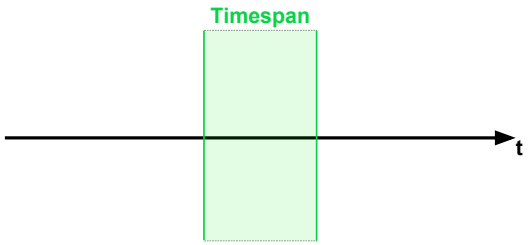

Figura 3.1: Timespan

Questo modello studia l'evoluzione dei contenuti di una piattaforma di editing collaborativo nel tempo e lo fa analizzando situazioni di disallineamento tra conoscenza del mondo reale e conoscenza espressa nei contenuti della piattaforma in un timespan. Un esempio di situazione che si vuole studiare con questo modello è quella di un contenuto della piattaforma molto editato in un timespan, a cui corrispondere un evento nel mondo reale che riguarda il tema del contenuto. Questo vuol dire che ci sarà stato un momento di disallineamento tra conoscenza nel mondo reale che riguarda il tema del contenuto e quella espressa dal contenuto. Un'altro esempio può essere di un contenuto a cui gli autori si disinteressano in un certo timespan e nel frattempo è successo un evento nel mondo reale, con conseguente disallineamento del contenuto.

Si scriverà **CR** in riferimento alla "conoscenza del mondo reale", mentre **CP** la "conoscenza espressa dai contenuti della piattaforma".

Si considera la linea temporale in figura 3.1 come un riferimento nel tempo della piattaforma, in cui può avvenire o meno evoluzione dei contenuti.

Prima di definire il ruolo dei timespan riguardo il modello, è bene definire il concetto di disallineamento formalmente, di conseguenza anche il concetto di allineamento.

Allineamento: situazione in cui in un timespan CR = CP. La formula 3.1 spiega come pensando di quantificare la conoscenza nei due casi, l'allineamento corrisponde ad un rapporto uguale ad 1.

$$\frac{CR}{CP} = 1 \tag{3.1}$$

Disallineamento: situazione in cui in un timespan CR > CP (formula 3.2).

$$\frac{CR}{CP} > 1 \tag{3.2}$$

In figura 3.2 viene schematizzata l'evoluzione dei contenuti. L'evoluzione che porterà all'allineamento (CR = CP) viene preceduta da un timespan in cui CR > CP (disallineamento).

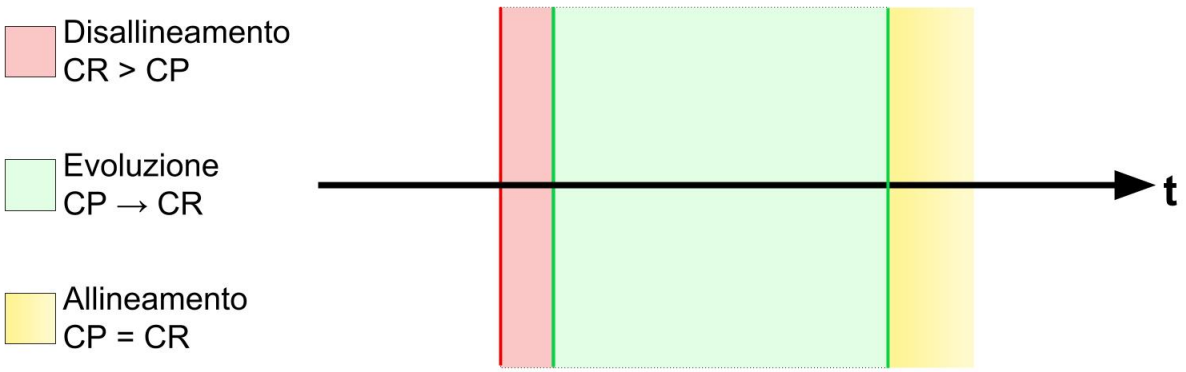

Figura 3.2: Eventi evoluzione contenuti

3.2 Indicatori

Per cercare un timespan in cui si ha evoluzione dei contenuti, è necessario una suo log (traccia) da cui ricavare indicatori che siano rappresentativi dell'evoluzione dei contenuti nel timespan, in modo che si possa riconoscere di conseguenza situazioni di disallineamento.

Il log in questione nel caso della piattaforma Wikipedia è lo storico revisioni come discusso nel capitolo 2.2.1. D'ora in poi ogni riferimento a "piattaforma" è da considerarsi come "piattaforma Wikipedia", in quanto è la piattaforma in esame in questo lavoro, anche se il modello proposto è generalizzabile a qualunque piattaforma di editing collaborativo, con i dovuti accorgimenti. Anche CP (conoscenza piattaforma) è da considerarsi d'ora in poi in riferimento alla piattaforma Wikipedia, così come evoluzione dei contenuti fa riferimento all'evoluzione dei contenuti in Wikipedia. Nei lavori correlati citati nel capitolo 2.3 si ricavano indicatori dallo storico revisioni per studiare la loro relazione con la IQ delle pagine. Sebbene gli scopi di questo modello siano diversi, questi indicatori sono rappresentativi dell'evoluzione dei contenuti e sono utili per studiarla, quindi efficaci anche in questo lavoro, dove sono stati usati come indicatori di evoluzione dei contenuti. Gli indicatori anticipati non sono la totalità degli indicatori scelti. Questi sono **potenziali indicatori di disallineamento**, in quanto danno informazioni riguardo l'evoluzione della pagina, ma non necessariamente indicano disallineamento. Di seguito si entrerà nel dettaglio degli indicatori scelti per questo modello, parlando di come sono stati ricavati e per ognuno si fornirà una spiegazione del perché sono utili alla causa.

3.2.1 Edits

Edits è l'indicatore che riguarda il numero di revisioni di una pagina in un certo timespan e può essere calcolato a partire dallo storico delle revisioni. Si assume che se in un timespan una pagina viene modificata significa che c'è stato un momento in cui c'è stato disallineamento, mentre se in un timespan il contenuto non ha subito modifiche posso assumere che non c'era bisogno di modificarlo perché era in una situazione di allineamento.

Minor Edits

L'indice Minor edit riguarda il numero di revisioni di importanza minore che non comporta cambiamenti sostanziali al contenuto della pagina come ad esempio una correzione ortografica. Può essere calcolato a partire dallo storico delle revisioni. Si assume che se le modifiche effettuate in un timespan sono edits minori, allora si è nel caso di allineamento. Viceversa se non si tratta di minor edits.

Editors

L'indice editors riguarda il numero degli editori diversi che contribuiscono alle modifiche di una pagina e può essere calcolato a partire dallo storico delle revisioni. Si assume che se in un timespan una pagina viene modificato da molti editori diversi, significa disallineamento che ha richiesto che molti editori diversi modificassero la pagina, mentre se in un timespan il contenuto ha pochi editori diversi posso assumere che il contenuto sia stabile e che quindi ci sia una situazione di allineamento.

3.2.2 Comments

Comments è l'indice che riguarda il numero di revisioni del talk (capitolo 2.2 per i dettagli del talk di una pagina) di una pagina in un certo timespan e può essere calcolato a partire dallo storico delle revisioni dell talk della pagina. Questo indice è chiamato comments in quanto la pagina talk di una pagina è usata dagli editori della pagina per collaborare alle modifiche della pagina e discutere sulle modifiche apportate. Si è assunto che ogni revisione della pagina talk sia un commento di un editore che appunto commenta per collaborare alla pagina. Si assume che se in un timespan ci sono molti commenti relativi ad una pagina e siccome gli editori usano i commenti per discutere sulle modifiche apportate, vuol dire che c'è stato un momento in cui il contenuto era disallineato e che ha necessitato di modifiche.

3.2.3 Views

L'indice views riguarda il numero di visualizzazioni di una pagina in un certo timespan. Si assume che se in un timespan ci sono molte visualizzazioni questo potrebbe essere conseguenza ad un evento relativo alla pagina nel mondo reale e che c'è stato un momento di disallineamento in cui il contenuto della pagina necessitava modifiche.

3.2.4 Componenti strutturali

Gli indicatori sui componenti strutturali riguardano gli indicatori del numero di link esterni ed interni ed il numero di sezioni di una pagina Wikipedia. Si assume che se in un timespan sono aumentati i valori di questi indicatori e quindi il contenuto di una pagina, allora vuol dire che c'è stato un periodo nel passato in cui il contenuto era disallineato.

3.3 Filtri

Gli indicatori descritti in precedenza possono essere usati per scoprire eventuali situazioni di disallineamento, questo è alla base di questa tesi. Infatti guardando questi indicatori

è possibile verificare alcune configurazioni e accorgersi se un contenuto è disallineato o meno.

Si consideri una sola pagina Wikipedia, quindi in questo caso CP = "conoscenza espressa *da una pagina* della piattaforma Wikipedia" ed evoluzione dei contenuti fa riferimento all'evoluzione dei contenuti di una sola pagina. Alla linea temporale si sovrappone l'evoluzione della pagina, cioè il suo storico revisioni. Il risultato è schematizzato nella figura 3.3, in cui ad ogni pallino nella linea temporale corrisponde ad una revisione (di norma non sono distribuite uniformemente nel tempo) della pagina, cioè un edit.

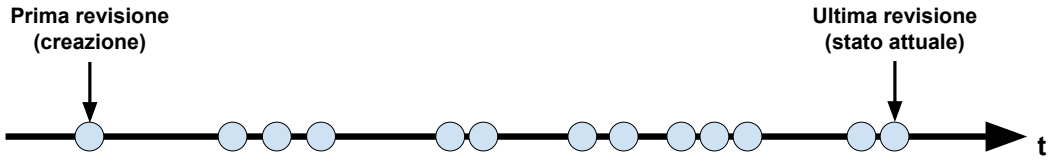

Figura 3.3: linea temporale e storico revisioni pagina

Si consideri ad esempio l'indicatore edits, che permette di misurare quante volte il contenuto è stato editato. Alla figura 3.3 applico un timespan che "cattura" un sottoinsieme dello storico delle revisioni della pagina (figura 3.4). Questo dirà in modo in cui la pagina è stata editata.

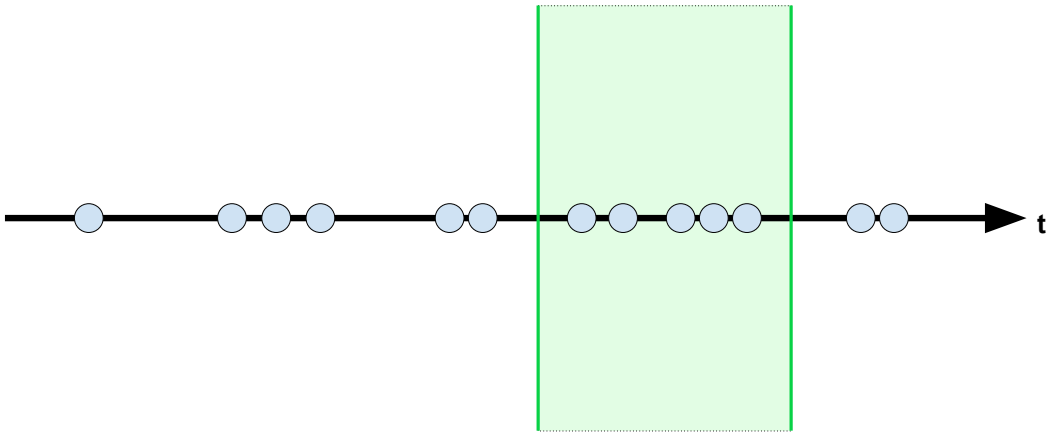

Figura 3.4: Sottoinsieme storico revisioni dato dal timespan

Ampliando il discorso su una collezione di pagine, questo permette di classificarle ad esempio come mostrato in figura 3.5.

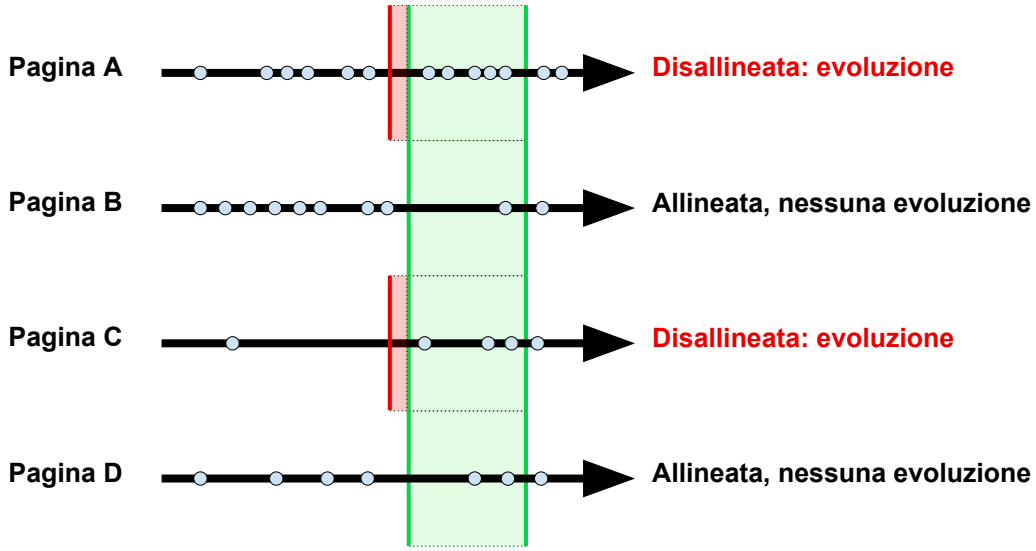

Figura 3.5: Tag pagine

La figura 3.9 mi dice che:

- pagina disallineata (A, C): c'è evoluzione e quindi disallineamento perché la pagina nel timespan ha subito edits con un'alta intensità
- pagina allineata (B, D): è allineata nel timespan e quindi non c'è evoluzione. Questo perché la pagina ha subito edits prima del timespan comportando l'allineamento e di conseguenza poco bisogno di edits.

Lo scopo della classificazione è applicare una etichetta (tag) ad ogni pagina, che aiuti a capire cosa è successo in quel timespan per ogni pagina. L'etichetta corrisponde ad una configurazione dei miei indicatori.

Non è detto che il mapping tra la configurazione degli indicatori ed etichetta sia necessariamente stabile, perché potrebbero esserci configurazioni diverse per scenari diversi: casi in cui si cercano molti edit e molte views, casi in cui si cercano pochi commenti, e così via. La configurazione si traduce in un insieme di filtri, che sono soglie rispetto alle quali vengono valutati i vari indicatori. Nella discussione è stato fatto per gli edits, ma lo stesso si può ripetere per tutti gli altri indicatori.

Assegnazione tag

Per spiegare la modalità con la quale vengono assegnate le etichette alle pagine in base alla configurazione degli indicatori per semplicità si torna all'esempio di una sola pagina (figura 3.4).

Si deve fare una valutazione sugli indicatori per il timespan di riferimento per stabilire la mappatura tra configurazione filtri ed etichetta. La mappatura va fatta a seconda dei casi e spetta all'utilizzatore del modello.

Si fa l'esempio di configurazione degli indicatori per apporre il tag allineata/disallineata, quindi per riconoscere una situazione di evoluzione dei contenuti. Si indicano con "valori indicatori sottoinsieme" i valori degli indicatori per il sottoinsieme dello storico revisioni in figura 3.4. Se la configurazione dei filtri corrisponde con i valori degli indicatori del sottoinsieme, cioè:

$$\textbf{configurazione filtri} = \textbf{Valori indicatori sottoinsieme} \tag{3.3}$$

Allora il tag sarà "disallineata", facendo riferimento al fatto che in un timespan antecedente l'evoluzione si è avuto disallineamento (come mostrato in figura 3.6).

Se invece:

$$\textbf{Valori indicatori filtro} \mathrel{!}= \textbf{Valori indicatori sottoinsieme} \tag{3.4}$$

Allora si ha una situazione di allineamento, quindi per quel timespan non si ha evoluzione.

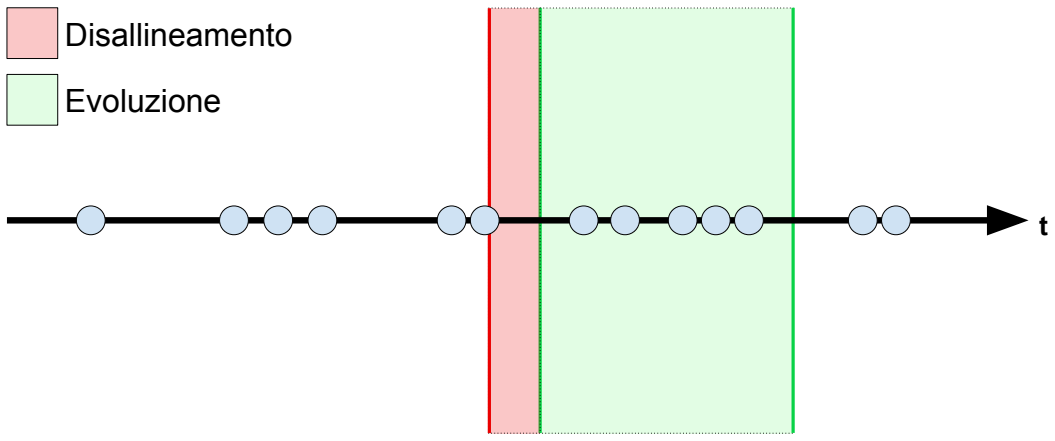

Figura 3.6: linea temporale e storico revisioni pagina

Nel caso di tag "disallineata" sarà utile andare a vedere gli indicatori per uno o più timespan antecedenti e/o sovrapposti al primo per studiare l'evoluzione della pagina (figura 3.7 a pagina seguente).

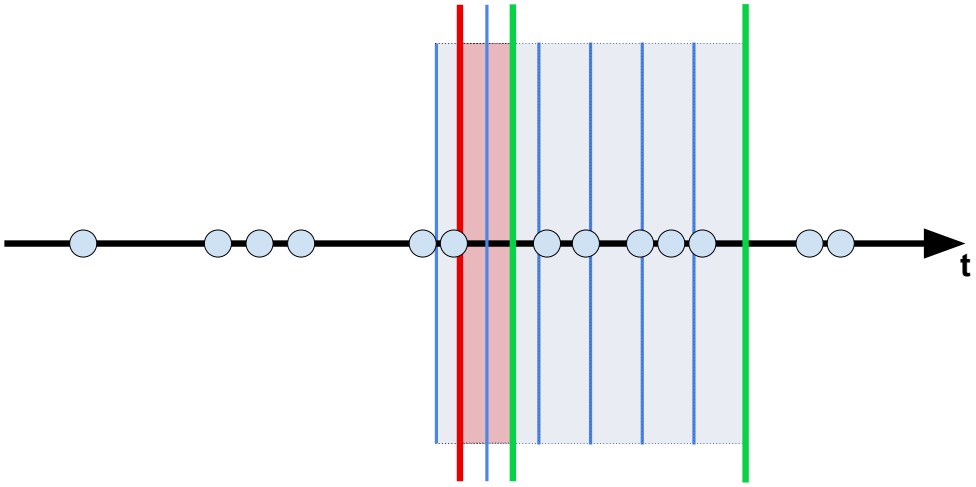

Figura 3.7: Studio evoluzione pagina tramite timespan

Capitolo 4

Monitor Wiki: un software per monitorare l'evoluzione dei contenuti

In questo capitolo parlerà di Monitor Wiki (MoW), uno script parametrico command line che implementa il modello proposto nel capitolo 3. Lo script è pensato per lavorare su diverse installazioni di MediaWiki. I dettagli implementativi verranno discussi nel capitolo 5, in questo si parlerà delle operazioni di MoW che sono Preview, List, Info, AggregateInfo e dei loro input e output.

MoW permette di selezionare un host (installazione di MediaWiki) su cui lavorare, selezionare le pagine per categoria o per titolo, filtrare le pagine in base al timespan e in base ai parametri sui filtri degli indicatori, raccogliere dati sugli indicatori in un determinato timespan.

Per tutte e 4 le operazioni, i parametri in input verranno presi sotto forma di "-flag valoreFlag".

4.1 Preview

L'operazione Preview prende in input una collezione di pagine, un timespan e filtri su uno o più indicatori. L'output sarà una lista di pagine in cui per ogni pagina si ha il tag misaligned, che è true se la pagina nel dato timespan rispetta i parametri di filtro degli indicatori, false altrimenti. Oltre al tag, per ogni pagina si avrà anche i valori degli indicatori per quel timespan.

Esempio

- Input: -h en.wikipedia.org -q category:computer science -l 1 -t 20180101,20190101 -n 100,1000 -v 50000,* -c 2,*

- Output: figura 4.1

```
Title: Artificial intelligence | misalignment: true | edits: 857  comments: 146 views: 3777297
Title: Computer science | misalignment: true | edits: 246  comments: 10 views: 3019136
Title: Data science | misalignment: true | edits: 190  comments: 29 views: 811655

3 misaligned pages / 1567 total pages
```

Figura 4.1: Esempio risultato operazione Preview

Input

Per l'operazione Preview, lo script prende in input i seguenti flag:

- -h : host (installazione MediaWiki)
- -q : collezione di pagine in input nella forma "titoloPagina1, titoloPagina2, ...". E' possibile scegliere una o più categorie dalle quali verranno prese le pagine, nella forma "Category:titoloCategoria1, Category:titoloCategoria2". Alternativamente una combinazione delle due "Category:titoloCategoria1, titoloPagina1, titoloPagina2, Category:titoloCategoria2, ...".
- -l : livello di profondità per cui prendere pagine nelle categorie. Con il valori "0" si prenderanno le pagine della categoria, con 1 le pagine della categoria e pagine della sottocategoria, ecc...
- -t : timespan nella forma "YYYYMMDD,YYYYMMDD"
- -a : se settato, si mostrano in output tutte le pagine (disallineati e non). Se non è settato, in output ci saranno solo le pagine misaligned true.
- filtri indicatori: intervallo di valori naturali (0 incluso) delimitato da due estremi: "valore1,valore2". Il valore2 può essere "*" che equivale a +infinito.
  - -n : filtro indice edits
  - -c : filtro indice comments
  - -v : filtro indice views
  - -f : filtro indice frequenza (edits/anno, si può usare in alternativa a indice edit).

Output

L'output consisterà in una lista di pagine in cui per ogni pagina si avrà il tag misaligned con valore true o false a seconda che per il timespan preso in riferimento si rispetti l'AND delle condizioni dei filtri sugli indicatori. Se è settato il flag -a si avranno solo le pagine misaligned true. Infine, sarà indicato il totale delle pagine misaligned true rispetto al totale delle pagine della ricerca.

Una precisazione doverosa da fare è che le pagine che sono stati creati dopo del timespan preso in input, non compariranno nell'output, in quanto la loro prima revisione non rientra nel timespan. Diversamente, se una pagina è stata creata in un periodo antecedente al timespan in input ed in questo timespan non ha subito modifiche, essa comparirà nella lista delle pagine in output, perché la prima revisione risale ad un periodo precedente al timespan.

4.2 List

La seconda operazione che il modello Monitor Wiki propone è List. Concettualmente è la stessa operazione di Preview, ma cambia l'output, che in questo caso sarà un file.

Input

Tutti i flag in input dell'operazione Preview valgono anche per l'operazione List. Si aggiunge il flag -e per indicare il nome del file out.

Output

L'output consisterà nella lista delle pagine in input in cui per ogni pagina si ha il tag di disallineamento corrispondente, calcolato in base al timespan e ai parametri degli indicatori in input come nel caso dell'operazione di Preview. Oltre al tag di disallineamento per ogni singola pagina, vi è il tag di disallineamento per ogni indicatore, che esplica se per quell'indice la pagina rispetta i criteri per i valori di filtro degli indicatori.

Esempio

Input: -h en.wikipedia.org -a -q machine learning, blockchain -t 20180101,20190101 -n 500,* -v 1000,* -c 2,*

Output: figura 4.2

```
{
    "query": {
        "h": "en.wikipedia.org",
        "a": "",
        "q": "machine learning, blockchain",
        "t": "20180101,20190101",
        "n": "500,*",
        "v": "1000,*",
        "c": "2,*",
        "e": "list.json"
    },
    "pages": {
        "233488": {
            "title": "Machine learning",
            "misalignment": {
                "isMisaligned": false,
                "misalignmentForFilter": {
                    "edits": false,
                    "comments": true,
                    "views": true
                }
            }
        },
        "44065971": {
            "title": "Blockchain",
            "misalignment": {
                "isMisaligned": true,
                "misalignmentForFilter": {
                    "edits": true,
                    "comments": true,
                    "views": true
                }
            }
        }
    }
}
```


Figura 4.2: Esempio risultato operazione List

4.3 Info

L'operazione Info ha lo scopo di raccogliere informazioni di una o più pagine in uno o più timespan. Le informazioni in questione sono:

- storico revisioni della pagina

  - per ogni revisione links, numero links, numero sezioni

- storico revisioni del talk della pagina
- views con granularità giornaliera della pagina in cui per ogni giorno si ha il numero delle views e il timestamp del giorno
- i giorni di vita di una pagina dalla sua creazione all'estremo destro del timespan
- data relativa alla prima revisione della pagina

Sarà possibile scegliere quali delle informazioni raccogliere, tranne che per la data relativa alla prima revisione della pagina e i giorni di vita di una pagina.

Le pagine in input dalla quale prelevare le informazioni saranno le pagine in output dell'operazione List. Per quanto riguarda il/i timespan in input, non per forza dovranno essere all'interno del timespan preso nell'operazione List.

E' doveroso precisare che se una pagina non è presente nel risultato dell'operazione List che si da in input all'operazione Info, questa pagina non sarà presente in nessun caso nel risultato di questa operazione.

Input

- -f : file con lista di pagine risultato di una operazione List
- -t : si possono indicare uno o più timespan ad esempio "timespan1,timespan2,...". Il formato del timespan sarà uguale a quello delle operazione Preview e List (YYYYMMDD,YYYYMMMMDD).
- -i: quali informazioni includere per ogni pagina. I valori possibili sono edits, views, comments, nlinks (numero di links), listlinks (lista dei links)
- -all : include tutte le informazioni citate nel flag -i. Questo flag è da usare al posto di -i se si vogliono includere tutte le informazioni.

Output

Per ogni timespan in input si avrà un risultato di questo tipo:

- per ogni pagina
  - titolo pagina
  - timestamp prima revisione
  - giorni di vita pagina

  - lista views, ogni elemento composto da:
    * views giornaliere
    * timestamp
  - storico revisioni, ogni elemento composto da:
    * info revisione
    * links
    * lista links
    * sezioni
  - lista delle revisioni del talk della pagina, ogni elemento composto da:
    * info revisione

I campi dell'output sono condizionati dagli indicatori in input, tranne per i campi titolo pagina, timestamp prima revisione e giorni di vita pagina. Di seguito la regola per la presenza dei campi:

- lista views: presente se negli indicatori è indicato "views"
- storico revisioni: presente se negli indicatori è indicato "edits" e/o "nlinks" e/o "listlinks" e/o "sections"
- lista delle revisioni del talk della pagina: presente se negli indicatori è indicato "comments"
- links : presente se negli indicatori è indicato "nlinks"
- lista links : presente se negli indicatori è indicato "listlinks"
- sezioni : presente se negli indicatori è indicato "sections"

Esempio

- Input List: -h en.wikipedia.org -q machine learning, blockchain -t 20180101,20190101 -n 100,* -v 1000,* -c 2,* -e list.json
- Input Info: -f list.json -t 20180101,20180102 -t 20190101,20190102 -d info.json -i edit,views,comments
- Output: figura 4.3 e figura 4.4

```
{
    "result": {
        "0": {
            "query": {
                "f": "list.json",
                "t": "20180101,20180102",
                "d": "info.json",
                "i": "edit,views,comments",
                "h": "en.wikipedia.org"
            },
            "pages": {
                "233488": {
                    "title": "Machine learning",
                    "revisions": {…
                    },
                    "views": [
                        {
                            "timestamp": "20180101",
                            "views": 5898
                        },
                        {
                            "timestamp": "20180102",
                            "views": 6943
                        }
                    ],
                    "talks": {
                        "history": [],
                        "count": 0
                    },
                    "daysOfAge": 5337,
                    "firstRevision": "2003-05-25T06:03:17Z"
                },
                "44065971": {
                    "title": "Blockchain",
                    "revisions": {
                        "history": […
                        ],
                        "count": 10
                    },
                    "views": [
                        {
                            "timestamp": "20180101",
                            "views": 30510
                        },
                        {
                            "timestamp": "20180102",
                            "views": 31677
                        }
                    ],
                    "talks": {
                        "history": […
                        ],
                        "count": 5
                    },
                    "daysOfAge": 1181,
                    "firstRevision": "2014-10-09T14:38:31Z"
                }
            }
        },
```


Figura 4.3: Esempio risultato operazione Info, prima parte

```
"1": {
    "query": {
        "f": "list.json",
        "t": "20190101,20190102",
        "d": "info.json",
        "i": "edit,views,comments",
        "h": "en.wikipedia.org"
    },
    "pages": {
        "233488": {
            "title": "Machine learning",
            "revisions": {
                "history": [⋯
                ],
                "count": 1
            },
            "views": [
                {
                    "timestamp": "20190101",
                    "views": 5783
                },
                {
                    "timestamp": "20190102",
                    "views": 5382
                }
            ],
            "talks": {
                "history": [],
                "count": 0
            },
            "daysOfAge": 5702,
            "firstRevision": "2003-05-25T06:03:17Z"
        },
        "44065971": {
            "title": "Blockchain",
            "revisions": {
                "history": [],
                "count": 0
            },
            "views": [
                {
                    "timestamp": "20190101",
                    "views": 8185
                },
                {
                    "timestamp": "20190102",
                    "views": 14822
                }
            ],
            "talks": {
                "history": [],
                "count": 0
            },
            "daysOfAge": 1546,
            "firstRevision": "2014-10-09T14:38:31Z"
        }
    }
}
}
```


Figura 4.4: Esempio risultato operazione Info, seconda parte

4.4 Aggregate Info

L'operazione aggregazione sulle informazioni degli indicatori è simile all'operazione Info, solo che invece di presentare in output le informazioni sulle pagine in forma estesa, li presenta in forma aggregata, cioè presenterà la loro quantità.

Gli indicatori delle pagine di cui sarà possibile avere i valori aggregati saranno:

- edits: numero totale delle revisioni
- minor edits: numero delle revisioni minori apportate ad una pagina
- authors: numero di autori distinti che hanno contribuito alle revisioni
- comments : numero di revisioni della pagina discussione della corrispondente pagina
- views : totale numero di visualizzazioni della pagina
- links : numero di links
- sezioni : numero delle sezioni

I valori numerici degli indicatori soprastanti riguarderanno il timespan di riferimento.

Come nel caso dell'operazione Info, per ogni pagina si avranno le informazioni sui giorni di vita di una pagina e la data di prima revisione.

Input

I flag in input sono gli stessi dell'operazione Aggregate Info. Questa volta il flag -i indicherà per quali informazioni si otterranno indicatori aggregati.

Sarà possibile scegliere uno o più indicatori di quelli possibili in input.

Output

Per ogni timespan in input si avrà un risultato di questo tipo:

- per ogni pagina
  - titolo pagina
  - timestamp prima revisione
  - giorni di vita pagina
  - views
  - edits

  - minor edits
  - authors
  - comments
  - storico revisioni, ogni elemento composto da:
    * info revisione
    * links
    * sezioni

I campi dell'output sono condizionati dagli indicatori in input, tranne per i campi titolo pagina, timestamp prima revisione e giorni di vita pagina. Di seguito la regola per la presenza dei campi:

- views: presente solo se negli indicatori è indicato "views"
- edits, minor edits , authors, comments: presente solo se negli indicatori è indicato "edits"
- storico revisioni: presente solo se negli indicatori è indicato "links" e/o "sections"
- links: presente solo se negli indicatori è indicato "links"

Esempio

- List: -h en.wikipedia.org -q blockchain,machine learning -t 20180101,20181010 -n 0,* -v 0,* -c 0,* -e list.json
- Aggregate Info: -f list.json -t 20180101,20180501 -t 20190101,20190201 -d aggregateInfo.json -i edit,views,comments
- output: figura 4.5 (a pagina seguente)

```
{
    "query": {
        "f": "list.json",
        "t": {
            "0": "20180101,20180501",
            "1": "20190101,20190201"
        },
        "d": "aggregateInfo.json",
        "i": "edit,views,comments"
    },
    "result": {
        "0": {
            "timespan": "20180101,20180501",
            "pages": {
                "233488": {
                    "title": "Machine learning",
                    "daysOfAge": 5456,
                    "firstRevision": "2003-05-25T06:03:17Z",
                    "edits": 83,
                    "minorEdits": 14,
                    "authors": 60,
                    "comments": 4,
                    "views": 784425
                },
                "44065971": {
                    "title": "Blockchain",
                    "daysOfAge": 1300,
                    "firstRevision": "2014-10-09T14:38:31Z",
                    "edits": 432,
                    "minorEdits": 79,
                    "authors": 188,
                    "comments": 39,
                    "views": 2568986
                }
            }
        },
        "1": {
            "timespan": "20190101,20190201",
            "pages": {
                "233488": {
                    "title": "Machine learning",
                    "daysOfAge": 5732,
                    "firstRevision": "2003-05-25T06:03:17Z",
                    "edits": 26,
                    "minorEdits": 8,
                    "authors": 23,
                    "comments": 1,
                    "views": 184646
                },
                "44065971": {
                    "title": "Blockchain",
                    "daysOfAge": 1576,
                    "firstRevision": "2014-10-09T14:38:31Z",
                    "edits": 12,
                    "minorEdits": 4,
                    "authors": 11,
                    "comments": 5,
                    "views": 266923
                }
            }
        }
    }
}
```


Figura 4.5: Esempio risultato operazione AggregateInfo

Capitolo 5

Implementazione

Nel capitolo 4 si è parlato di un modello concettuale per monitorare l'evoluzione dei contenuti su piattaforme di editing collaborativo. Si è preso come riferimento l'enciclopedia online Wikipedia per spiegare le idee alla base del modello e le operazioni per far emergere dati utili a capire l'evoluzioen dei contenuti in un ambiente di editing collaborativo. In questo capitolo si propone un software (MoW, che sta per Monitor Wiki) che implementa il modello formalizzato nel capitolo 4. Questo software è stato pensato per le varie versioni di Wikipedia, ma più in generale per istanze MediaWiki (di cui ogni Wikipedia ne è una). Sebbene questo prototipo sia fortemente dipendente dalle caratteristiche dell'infrastruttura MediaWiki, si può prendere come punto di partenza e di esempio per implementazioni software del modello per piattaforme di editing collaborativo differenti. Si parlerà in particolare di cos'è il software MediaWiki, le scelte implementative quali linguaggio di programmazione, librerie, architettura del software, codifica delle componenti principali ed infine dei problemi del software che possono interessare e influire su sviluppi futuri.

5.1 Il software MediaWiki

Prima di entrare nei dettagli dell'implementazione è necessario spiegare cos'è il software MediaWiki, in quanto influenzerà in gran parte MoW. La definizione di cosa è il software MediaWiki è riportata come segue nel suo manuale d'uso: " MediaWiki è un software libero lato server, rilasciato sotto licenza GNU General Public License (GPL). È progettato per essere eseguito su grandi server farm per siti che ricevono milioni di visite al giorno. MediaWiki è un software estremamente potente e scalabile, che permette di implementare wiki ricche di funzionalità. Utilizza PHP per processare e mostrare dati salvati in un database, come MySQL. Le pagine usano il formato wikitext, cosicché anche gli utenti che non conoscono l'HTML o il CSS possono modificarle facilmente. Quando un utente invia una modifica a una pagina, MediaWiki lo scrive nel database, ma senza

eliminare le versioni precedenti della pagina, consentendo quindi di tornare facilmente in caso di vandalismo o spam. MediaWiki può anche gestire file immagine e multimediali, che sono memorizzati nel filesystem. Per i wiki di grandi dimensioni con molti utenti, MediaWiki supporta il caching e può essere facilmente accoppiato con il software del server proxy. Il programma è stato sviluppato principalmente per essere eseguito in grandi server farm per Wikipedia e per i suoi progetti fratelli. Le sue caratteristiche, funzionamento, capacità di configurazione, facilità di uso, ecc. sono disegnati spesso con questa prospettiva".

MoW è funzionante su qualsiasi istanza di MediaWiki, in quanto utilizza gli strumenti del software MediaWiki (quali le API per reperire le informazioni) comuni a tutte le piattaforme che utilizzano questo software.

5.1.1 API

Nella definizione del software MediaWiki si legge "Utilizza PHP per processare e mostrare dati salvati in un database, come MySQL". Una parte del software MediaWiki assolve a questo compito e sono le API MediaWiki, strumenti tramite la quale è possibile reperire informazioni nel database dell'istanza MediaWiki ma non solo. Infatti le API MediaWiki permettono di accedere a varie funzionalità wiki come l'autenticazione, le operazioni sulle pagine e la ricerca. Può fornire meta informazioni sul wiki e sugli utenti che hanno effettuato l'accesso. In sostanza, le seguenti sono le operazioni principali:

- Controllare una installazione MediaWiki
- Creare bot per gestire una installazione MediaWiki
- Accedere ad una istallazione MediaWiki, accedere ai dati e postare cambiamenti tramite richieste HTTP.

Nel caso in esame la funzionalità delle API che più ci interessa è una parte della terza in lista, cioè "accedere ad una istallazione MediaWiki".

Di seguito un esempio di come si possono usare le API MediaWiki per reperire informazioni.

Ottenere storico revisioni pagina

Si vuole ottenere lo storico delle revisioni della pagina 'Computer Science' in Wikipedia per il giorno 2018-12-07. Per ogni revisione si vuole: id della revisione (identifica la pagina nello stato in cui era dopo la modifica relativa a quella revisione), il timestamp della revisione e l'utente che ha eseguito la revisione. Le componenti principali della richiesta sono:

- endpoint: https://en.wikipedia.org/w/api.php?

- parametri della richiesta:

```
{
    action: "query",
    prop: "revisions",
    rvprop: "ids | timestamp | user",
    titles: "Computer Science",
    rvstart: "2018-12-07T00:00:00Z",
    rvend: "2018-12-07T23:59:59Z",
    format: "json",
    formatversion: "2"
}
```

Il risultato avrà questa forma (esempio esplicativo e non completo nei suoi dettagli):

```
{
    "title": "Computer science",
    "revisions": [
        {
            "revid": 872526288,
            "parentid": 871505723,
            "user": "45.114.89.21",
            "anon": true,
            "timestamp": "2018-12-07T17:43:27Z",
        },
        {
            "revid": 872526636,
            "parentid": 872526288,
            "user": "Gilliam",
            "timestamp": "2018-12-07T17:46:00Z",
        }
    ]
}
```

5.2 Scelte implementative

MoW è uno script parametrico eseguibile da command line, sviluppato in Node.js .

Si è scelto di sviluppare MoW in Node.js principalmente perché esiste una libreria scritta in Node.js chiamata "nodemw"che semplifica l'uso delle API MediaWiki. Si fa uso della libreria ogni volta (o quasi) che c'è bisogno di recuperare informazioni.

5.2.1 La libreria nodemw

La libreria nodemw [1] è una libreria scritta in Node.js la quale permette di semplificare molte operazioni eseguibili in un'istanza MediaWiki tramite le apposite API, ad esempio:

- richieste HTTP
- creazione/modifica/spostamento/rimozione di pagine
- upload di file
- operazioni di richieste di informazioni, ad esempio:
  - lista di pagine in una data categoria
  - prendere tutte le revisioni di una pagina
  - lista di tutte le categorie presenti nel wiki

In MoW si userà principalmente nodemw per le richieste HTTP.

E' doveroso parlare della libreria nodemw in quanto gioca un ruolo di rilievo nello sviluppo di MoW, in quanto è coinvolta in quasi tutte le operazioni di recupero delle informazioni. I motivi per cui si è scelto di usare nodemw sono principalmente due:

- gestione delle richieste in parallelo: nodemw si occupa di gestire le richieste in parallelo così da evitare il sovraccaricamento del server e l'eventuale interruzione nella risposta delle richieste.
- gestione delle richieste non complete: per richieste che riguardano grandi moli di dati (quali potrebbe essere ad esempio la storico revisioni di una pagina in un timespan molto ampio) si devono fare più richieste in quanto MediaWiki "divide" i dati in più parti. In ogni parte ci sarà il riferimento per la succesiva parte dei dati da ottenere con una nuova richiesta. Infine bisogna ricombinare i dati. Nodemw gestisce le richieste e la ricombinazione così che nell'utilizzo ci si debba preoccupare solo dell'input e dell'output.

Adattamento della libreria

Nodemw ha a disposizione molte operazioni di base ma poco flessibili (non è possibile selezionare i parametri a piacimento) e quindi poco utili ai fini di MoW. Nella libreria è presente una primitiva di uso generale (call) che permette di fare richieste con parametri a piacimento, ma che non integra uno dei punti di forza di nodemw, cioè la gestione delle richieste non complete.

[1] https://github.com/macbre/nodemw

E' stato quindi necessario creare un nuovo componente nella libreria (getAllParametricData), quasi identico ad uno già esistente (getAll) che gestisce le richieste solo per operazioni "preconfezionate" e che integra il meccanismo di gestione delle richieste non compelete. Il nuovo componente sfrutta il vecchio e lo rende adatto a qualsiasi richiesta parametrica.

5.3 Implementazione operazioni

Le operazioni del MoW hanno molte componenti software in comune. Ad esempio Preview e List sono molto simili nell'architettura, così come Info e Aggregate Info.

Nella spiegazione dell'implementazione delle operazioni ci si soffermerà sulle compomenti più importanti, non ripetendo la spiegazione più volte se il componente è presente in più operazioni.

Con compomente si intende una funzione che assolve ad un compito specifico nel software.

5.3.1 Preview

Nella figura 5.1 è raffigurata l'architettura dell'operazione Preview (la componente WriteFile non va considerata in quanto è una componente dell'operazione Preview) .

Di seguito i componenti principali e rispettivi componenti interni dell'architettura di Preview.

- SanityCheckPreview : controllo validità dell'input command line
- ParseRequest : conversione stringa in input in oggetto JSON
- SearchPages : ricerca pagine in base alla stringa di richiesta
  - WrapperNameInference : richiesta delle pagine più probabili in base alla stringa in input
  - WrapperGetPagesInfo : richiesta informazioni sulle pagine (la pagina è o meno una categoria)
  - WrapperGetInfoCategory : richiesta titolo pagine appartenenti alla categoria
  - CategorySearchPages : ricerca pagine in sottocategorie
- SearchFirstRevision : ricerca prima revisione delle pagine e gestione degli eventuali errori di richiesta
  - WrapperFirstRevision : richiesta prima revisione delle pagine

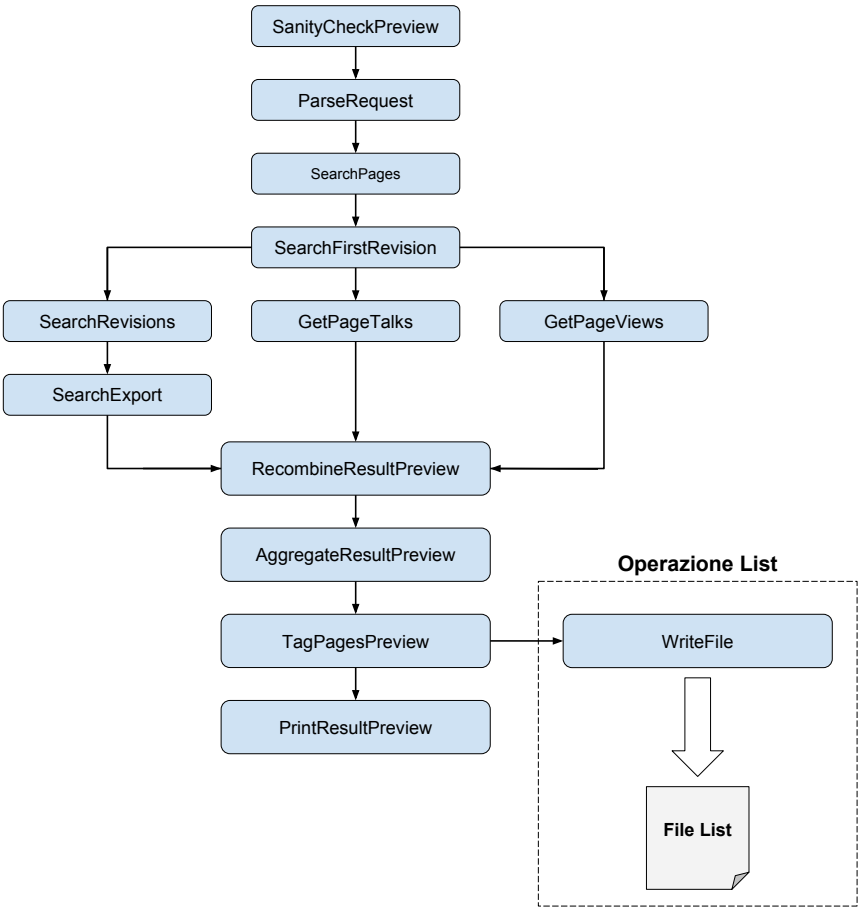

Figura 5.1: Architettura Preview e List

- SearchRevisions : ricerca storico revisioni delle pagine e gestione degli eventiali errori di richiesta
  - WrapperGetParametricRevisions : richiesta storico revisioni delle pagine
- GetPageTalks : ricerca storico revisioni del talk delle pagine e gestione degli eventuali errori di richiesta
  - WrapperTalks : richiesta storico revisioni dei talk delle pagine
- GetPageViews : ricerca views giornaliere delle pagine e gestione degli eventuali errori di richiesta
  - WrapperViews : richiesta views giornaliere delle pagine

- SearchExport : ricerca componenti strutturali delle revisioni (links esterni ed interni, sezioni) delle pagine gestione degli eventuali errori di richiesta
  - GetPageExport : richiesta componenti strutturali delle revisioni
- RecombineResultPreview : creazione oggetto in cui in ogni campo è dedicato ad una pagina e alle relative informazioni richieste
- AggregateResultPreview : aggregazione delle informazioni di ogni pagina sotto forma di indicatori
- TagArticlesPreview : tag delle pagine come misaligned true/false in base ai suoi indicatori delle informazioni
- PrintResultPreview : mostra nel terminale il risultato dell'operazione preview

Le componenti SearchRevisions, GetPageTalks e GetPageViews nella codifica vengono eseguite in questo ordine, anche se concettualmente possono essere considerate in parallelo.

Pattern per la ricerca di informazioni

Nel recupero di informazioni SearchFirstRevision, SearchRevisions, GetPageViews, SearchExport, GetPageViews viene utilizzato lo stesso pattern per le richieste e per la gestione degli eventuali errori. Per GetPageViews si ha qualche differenza rispetto alle altre, in quanto c'è bisogno di fare una richiesta diversa perché non non si possono usare le API standard di MediaWiki (in questo caso non è possibile utilizzare la libreria nodemw per la richiesta, si utilizzeranno le API RESTBase Wikimedia [2]). Di seguito verrà spiegato il pattern nel caso del recupero dello storico revisioni di una collezioni di pagine (il codice mostrato è esplicativo e non completo di tutti i suoi dettagli) e quello per il recupero delle views.

SearchRevisions prende in input il timespan che servirà per limitare la richieste per lo storico delle revisioni ad un timespan della vita delle pagine e la collezione delle pagine. Per ogni pagina viene fatta una richiesta (params è l'oggetto della richiesta per la singola pagina). Le richieste verrano "wrappate" tramite il wrapperGetParametricRevisions, che si occupera di effettuare la richiesta per una pagina e recuperare i dati. Le promesse verranno accodate e risolte tutte assieme. WrapperGetParametricRevisions ritorna una promessa "data" (lo storico revisione della pagina), mentre nel caso di errore della richiesta ritorna una promessa oggetto con titolo della pagina e campo error. Questo servirà per riconoscere nella coda di promesse risolta le richieste che non sono andate a buon fine. Le richieste non andate a buon fine verranno filtrate dalla coda dei risultati e verranno

[2] https://wikimedia.org/api/rest_v1/

reintrodotte nella coda "allPagesQuery" per essere di nuovo effettuate. SearchRevisions restituisce una promessa con la coda dei risultati delle richieste una volta che la coda "allPagesQuery" è vuota.

```
async function SearchRevisions(timespanArray, allPagesQuery) {
    return new Promise(async (resolve) => {
        let queue = [];
        let resultOfQuery = [];
        do {
            for (el of allPagesQuery) {
                let params = {
                    action: 'query',
                    prop: 'revisions',
                    rvprop: ['ids', 'timestamp', 'size', 'flags', '
                       comment', 'user'].join('|'),
                    rvdir: 'newer', // order by timestamp asc
                    rvlimit: 'max',
                    titles: el,
                    rvstart: timespanArray[0],
                    rvend: timespanArray[1]
                }
                queue.push(wrapper.wrapperGetParametricRevisions(params
                   ));
            }
            resultOfQuery = resultOfQuery.concat(await Promise.all(
               queue));
            queue = [];
            allPagesQuery = resultOfQuery.filter(el => { return el.
               hasOwnProperty('error') }).map(el => el.page);
            resultOfQuery = resultOfQuery.filter(el => { return !el.
               hasOwnProperty('error') });

        } while (allPagesQuery.length > 0)
        resolve(resultOfQuery);
    });
}
```

Listing 5.1: Recupero storico revisioni di una pagina

```
function WrapperGetParametricRevisions (params) {
    return new Promise((resolve) => {
        client.getAllParametricData(params, function (err, data) {
            if (err) {resolve({ page: params.titles, error: '' });}
            else {
            /*...parsing delle informazioni ricevute...*/
            resolve(data);} }); }); }
```

Listing 5.2: Richiesta storico revisioni di una pagina

Per la ricerca delle views giornaliere come preannunciato non si può sfruttare la libreria nodemw. Si dovrà gestire la coda delle richieste per evitare l'interruzione delle risposte. Si processa una porzione della lista di pagine alla volta, per accodamento delle richieste e gestione degli errori si segue il pattern senza cambiamenti.

```
async function getPageViews(pagesInfo, timespanArray, parsedRequest) {
    return new Promise(async (resolve) => {
        let queueViews = []; let resultViews = [];
        do {
            if (pagesInfo.length > 500) {
                while (pagesInfo.length > 0) {
                    queueViews = [];
                    chunkedArrayOfPages = pagesInfo.slice(0, 25);
                    for (let page of chunkedArrayOfPages) {
                        let params = {
                            pageTitle: page.title,
                            pageid: page.pageid,
                            start: timespanArray[0],
                            end: timespanArray[1],
                            server: parsedRequest.h
                        };
                        queueViews.push(wrapper.wrapperViews(params));
                    }
                    pagesInfo = pagesInfo.slice(25, pagesInfo.length);
                    resultViews = resultViews.concat(await Promise.all(
                        queueViews));
                }
            } else {
                for (let page of pagesInfo) {
                    queueViews.push(wrapper.wrapperViews({
                        pageTitle: page.title,
                        pageid: page.pageid,
                        start: timespanArray[0],
                        end: timespanArray[1],
                        server: parsedRequest.h
                    }));
                }
                resultViews = resultViews.concat(await Promise.all(
                    queueViews));
            }
            queueViews = [];
            pagesInfo = resultViews.filter(el => { return el.
                hasOwnProperty('error') });
            resultViews = resultViews.filter(el => { return !el.
                hasOwnProperty('error') });
        }
        while (pagesInfo.length > 0)
        resolve(resultViews);
```

```
40      }); }
```

Listing 5.3: Ricerca views di una pagina

Per quanto riguarda il WrapperViews, si devono fare due precisazioni. La prima è che l'API per la richiesta in un determinato giorno resistuisce le views della pagina del giorno prima. Occorre riscalare il timespan di un giorno indietro per fare la richiesta, quindi nel risultato bisognerà riscalare un giorno avanti il timestamp delle views giornaliere. La seconda è che l'API restiuisce informazione solo dalla data 2015/05/01 in poi, per date antecedenti restituirà "not found", quindi nell'oggetto relativo alle views giornaliere si applica 'Not Available'.

```
1  function WrapperViews (params) {
2      return new Promise((resolve) => {
3          params.start = functions.RescaleTimespanForViews(params.start,'
               right');
4          params.end = functions.RescaleTimespanForViews(params.end,'
               right');
5
6          let urlRequest = 'https://wikimedia.org/api/rest_v1/metrics/
               pageviews/per-article/' +
7              params.server + '/all-access/all-agents/' + encodeURI(
                   params.pageTitle) + '/daily/' +
8              params.start + '/' + params.end;
9
10         request(urlRequest, (err, res, body) => {
11             if ((err)) {
12                 params.error = '';
13                 params.title = params.pageTitle;
14                 resolve(params);
15             }
16             if (body.title === 'Not found.') {
17                 resolve({ title: params.pageTitle, pageid: params.
                       pageid, dailyViews: 'Not Available' });
18             }
19             else {
20                 for (el in body.items) {
21                     body.items[el].timestamp = functions.
                           RescaleTimespanForViews(body.items[el].timestamp
                           , 'left');
22                 }
23                 resolve({ title: params.pageTitle, pageid: params.
                       pageid, dailyViews: body.items });
24             }
25         });
26     });
27 };
```

Listing 5.4: Richiesta views giornaliere

Ricerca delle pagine in categorie e sottocategorie

L'utente può richiedere pagine in base alla categoria potendo scegliere il livello di profondità nella ricerca di pagine nelle sottocategorie. Si esegue una visita in ampiezza a partire dalla categoria (livello 0) fino alle sottocategorie del livello scelto. Questa ricerca è implementata dal componente CategorySearchPages.

CategorySearchPages prende in input la lista "pagesInfo", in cui per ogni elemento vi sono le informazioni di una pagina quali titolo, id, ns (ns==14 se la pagina è una categoria). "deepLevel" è il livello di profondità per la ricerca scelto dall'utente. "stack" è la lista delle categorie già visitate. Questo serve per evitare di "andare in loop". Infatti la struttura delle categorie di Wikipedia non è gerarchica ma è un grafo, quindi è possibile che una sottocategoria di una categoria abbia tra le sue pagine proprio la categoria. La ricerca delle pagine va avanti finché ci sono pagine che sono categorie nell'oggetto "pagesInfo" e finché non si è raggiunto il livello scelto dall'utente. Per ogni categoria si controlla che non sia già nella lista "stack". Se non è nella lista stack si ottengono le pagine della categoria tramite WrapperGetInfoCategory e si aggiungono alla lista da processare pagesInfo. Si aggiunge la categoria processata alla lista stack e la si elimina dalla lista pagesInfo. Se invece la categoria processata era già nella lista "stack", non si richiede le pagine della categoria e si elimina la categoria da "pagesInfo".

```
1  function CategorySearchPages(pagesInfo, deepLevel) {
2      let level = 0; let stack = [];
3      return new Promise(async (resolve) => {
4          while (ThereAreCategories(pagesInfo) && level <= deepLevel) {
5              for (let index in pagesInfo) {
6                  if (pagesInfo[index].ns === 14) {
7                      if (!stack.find(el => { return el === pagesInfo[
                           index].pageid })) {
8                          pagesInfo = pagesInfo.concat((await Promise.
                               resolve(wrapper.WrapperGetInfoCategory(
9                              {
10                                 "action": "query",
11                                 "format": "json",
12                                 "generator": "categorymembers",
13                                 "formatversion": "2",
14                                 "gcmpageid": pagesInfo[index].pageid,
15                                 "gcmtype": "page|subcat",
16                                 "gcmlimit": "max"
17                             }
18                         ))));
19                         stack.push(pagesInfo[index].pageid);
20                         pagesInfo.splice(index, 1);
21                     } else {
22                         pagesInfo.splice(index, 1);
23                     }
24                 }
```

```
25              }
26              level += 1;
27          }
28          resolve(pagesInfo);
29      });
30  }
```

Listing 5.5: CategorySearchPages

5.3.2 List

Come anticipato, l'operazione list consiste nell'operazione Preview in cui in output si ha un file, risultato dell'operazione Preview. Di conseguenza, tra i componenti di List ci sarà un'unico componente in più rispetto a Preview, WriteFile che è l'unico componente da aggiungere all'architettura (figura 5.1).

- WriteFile : creazione e salvataggio del file risultato dell'operazione Preview in formato JSON (in questo caso non corrisponde ad una funzione, ma può essere considerato un componente).

5.3.3 Info

La figura 5.2 mostra l'architettura dell'operazione Info (l'operazione ManageaggregateInfo non va considerata in quanto è un componente che fa parte della sola operazione AggregateInfo). Le componenti SearchRevisions, GetPageTalks e GetPageViews nella codifica vengono eseguite in questo ordine, anche se concettualmente possono essere considerate in parallelo. Come spiegato nel capitolo 4, l'operazione Info recupera le informazioni di una o più pagine per uno o più timespan. Questo vuol dire che il recupero delle informazioni viene eseguito per ogni timespan in input, come schematizzato in figura 5.2.

Le principali componenti dell'architettura dell'operazione Info che non sono già state citate per le operazioni Preview e List sono:

- SanityCheckInfo : controllo validità dell'input command line
- ReadFile : lettura del file di input
- RecombineResultInfo : ricombinazione dei risultati per i vari timespan in un unico oggetto
- InsertNotYetCreatedPagesInfo : le pagine che nel timespan in input non erano stati creati, vengono aggiunte al risultato finale con il campo "NotYetCreated"

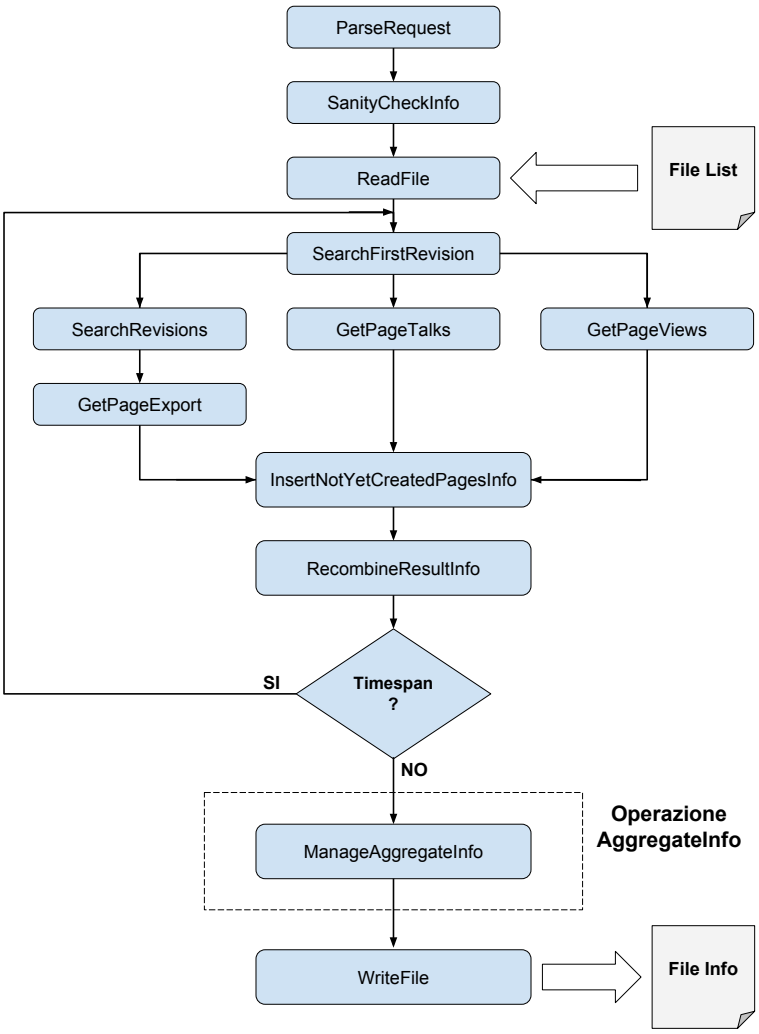

Figura 5.2: Architettura Info e AggreateInfo

5.3.4 Aggregate Info

Per l'operazione Aggregate Info si aggiunge all'architettura di info il componente di aggregazione degli indicatori (figura 5.2). Il file di output conterrà un oggetto in cui per ogni pagina si avranno le informazione aggregate sugli indicatori.

- ManageAggregateInfo : aggrega gli indicatori per le informazioni degli indicatori richieste

5.4 Problematiche principali

In questa sezione si analizzeranno i problemi principali lato utente quali i tempi di risposta e la memoria usata per l'esecuzione di MoW.

5.4.1 Tempi di riposta

Uno dei problemi principali di questo software sono gli inevitabili tempi di latenza dovuti alle richieste al server MediaWiki. Questo comporta lunghi tempi di esecuzione per richieste che coinvolgono molte pagine e molte revisioni, soprattutto nel caso in cui si includano nella ricerca gli indicatori strutturali con SearchExport. Per ogni operazione di ricerca di informazioni quali SearchFirstRevision, GetPageTalks, GetPageViews si fa una richiesta per ogni pagina (a meno che una richiesta non fallisca e debba erre rifatta). Dipendentemente dal timespan, le revisioni di ogni pagina possono essere numerose o meno, ma ci saranno da 0 a n revisioni per ogni pagina. Se si sceglie di ricevere anche le informazioni sugli indicatori strutturali quali links e sezioni, per ogni revisione si dovrà fare una richiesta tramite SearchExport. Ad esempio per 50 pagine con una media di 20 revisioni ognuno, nel caso pessimo si avranno 1000 richieste tramite SearchExport. Inoltre SearchExport è l'operazione più costosa in termini di risposta in quanto le informazioni di links e pagine arrivano in forma estesa e una pagina può avere da 0 a n links/sezioni. Più è corposa la pagina e più aumenta il tempo di risposta. Il tempo di esecuzione dipende dal numero di pagine, dal numero di revisioni e dal grandezza del timespan. Di seguito un test di analisi del caso pessimo dell'operazione che coinvolge più richieste e passaggi che è aggregateInfo, con le richieste di informazioni per tutti gli indicatori. Si fa il confronto tra del tempo di esecuzione di 20 pagine per tre timespan, uno di 1 anno, uno di 3 e uno di 5. Il test è stato svolto con una velocità di connessione in download di 60Mb/s.

La collezione di pagine è l'output della seguente operazione list:

- -h en.wikipedia.org -q category:computer science -l 1 -t 20180101,20190101 -n 100,* -c 1,* -v 10000,* -e list.json

Operazione aggregateInfo timespan 1 anno:

- -f list.json -t 20180101,20190101 -d aggregateInfo.json -i all
- numero revisioni: 3699
- tempo di esecuzione: 113 secondi
- tempo di esecuzione senza indicatori strutturali: 16 s

Operazione aggregateInfo timespan 3 anni:

- -f list.json -t 20160101,20190101 -d aggregateInfo.json -i all
- numero revisioni: 9638
- tempo di esecuzione: 233 secondi
- tempo di esecuzione senza indicatori strutturali: 19 s

Operazione aggregateInfo timespan 5 anni:

- -f list.json -t 20150101,20190101 -d aggregateInfo.json -i all
- numero revisioni: 12731
- tempo di esecuzione: 282.765 secondi
- tempo di esecuzione senza indicatori strutturali: 19 secondi

Da questo test si nota come il tempo di esecuzione cresce con il numero di revisioni nel caso in cui si includano indicatori strutturali, in quanto per ogni revisione si deve fare una richiesta molto costosa in termini di tempi di risposta tramite il componente SearchExport, più una richiesta per pagina per ogni tipo di indicatore (edit, views, comments, indicatori strutturali). Nel caso in cui non si includano indicatori strutturali, si deve fare solo una richiesta per pagina per ogni tipo di indicatore. I tempi di esecuzione in questo caso sono pressoché uguali (16 secondi, 19 secondi e 19 secondi), quindi per quanto riguarda questo test la grandezza del timespan per uno stesso numero di pagine non influisce in modo consistente sui tempi di esecuzione.

5.4.2 Memoria

Node di default per l'esecuzione ha una memoria dedicata di 512 MB e questo è un problema nel caso pessimo di utilizzo di MoW. Il caso pessimo riguarda l'operazione Info con richieste di informazioni per tutti gli indicatori: edits, comments, views e sopratutto indicatori strutturali quali links e sezioni. E' un caso pessimo in quanto le informazioni non vengono aggregate, quindi nel corso dell'esecuzione si devono tenere in memoria tutti i risultati in forma estesa, prima che questi vengano scritti nel file di output. Per richieste che riguardano molte revisioni di pagine molto corpose (molti links, molte sezioni), si potrebbe eccedere la memoria a disposizione. Una parziale soluzione a questo problema è estendere la memoria a disposizione per Node per l'esecuzione dello script. Per farlo, bisogna lanciare lo script con la seguente sintassi:

- node –max-old-space-size=8192 monitor-wiki.js

In questo caso si dedicano 8192MB di memoria per Node.

Per risolvere il problema è necessario utilizzare un database per salvare temporaneamente le informazioni, per non tenerle nella memoria dedicata a Node ed evitare così che si esaurisca la memoria e di conseguenza si interrompa l'esecuzione dello script.

Capitolo 6

Analisi

L'obiettivo di questo capitolo è mostrare esperimenti preliminari per testare il software MoW e verificare trend evidenti nell'evoluzione delle pagine. Questi esperimenti sono stati svolti prendendo in considerazione una sola categoria di pagine Wikipedia, per questo necessitano di ulteriori sviluppi, come saranno descritti nel prossimo capitolo. I casi discussi sono rappresentativi di situazioni comuni e riguardano: pagine in evoluzione, pagine allineate (bassa evoluzione o nulla), eventi nel mondo reale.

6.1 Pagine in evoluzione

Si vogliono studiare le pagine che sono in evoluzione ricercando quelle pagine che presentano alta attività nel tempo, cioè un numero elevato di interazioni da parte degli utenti con edits, comments, views.

Per farlo, bisogna definire una configurazione degli indicatori di attività che mi permetta di rilevare pagine di questo tipo in un timespan di 1 anno.

Edits	**Comments**	**Views**
365,*	50,*	200'000,*

Tabella 6.1: Configurazione indicatori di attività

La ricerca delle pagine avviene per la categoria Emerging technologies, in cui ci si aspetta che ci siano pagine con alta attività di evoluzione. I corrispondenti parametri di ricerca sono rappresentati nella tabella 6.2 (nel flag t, 1 anno sta ad indicare indicare un timespan di 1 anno, ad esempio per il 2018 il timespan 20180101,20181231).

Flag	h	q	l	t	n	c	v
Value	en.wikipedia	Category:Emerging technologies	1	1 anno	365	50,*	200'000,*

Tabella 6.2: Parametri di ricerca pagine in attività nel caso di pagine in evoluzione

Il risultato dela ricerca è indicato nella tabella 6.3.

Page	**Edits**	**omments**	**Views**
Artificial intelligence	855	146	3777297
Self-driving car	654	68	212735
5G	798	121	1717004
Blockchain	644	105	5592511

Tabella 6.3: Pagine in attività nella categoria Emerging technologies nell'anno 2018

Si studia ora la pagina con più attività, che risulta essere "Artificial intelligence" e si vuole studiare la sua evoluzione nell'anno 2016 e nel 2018.

Per farlo, si prendono le informazioni sugli indicatori quali edits, minor edit, authors, comments, views nei due anni ad intervalli (timespan) di 1 mese.

I grafici nelle figure 6.1 e 6.2 sono stati ricavati normalizzando ogni indicatore al suo valore massimo negli intervalli dei due anni.

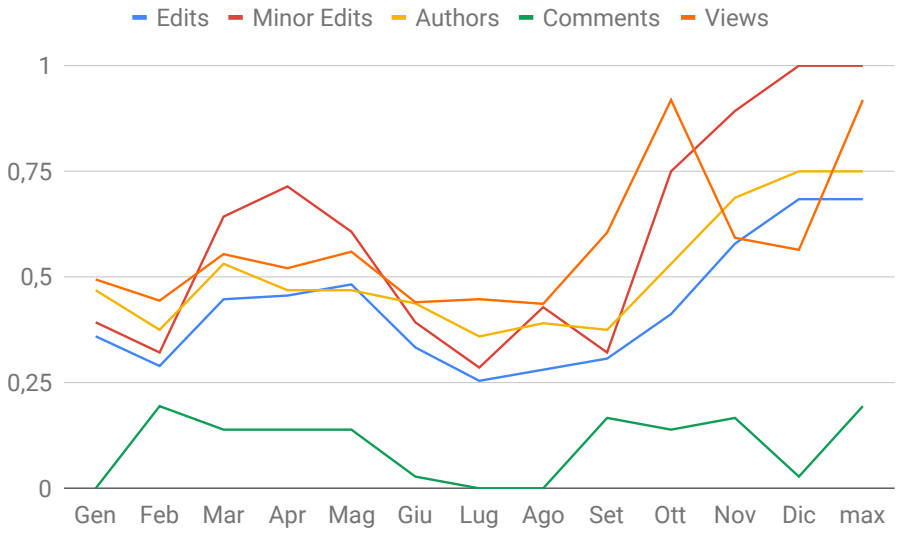

Figura 6.1: Indicatori attività della pagina "Artificial intelligence" nel 2016

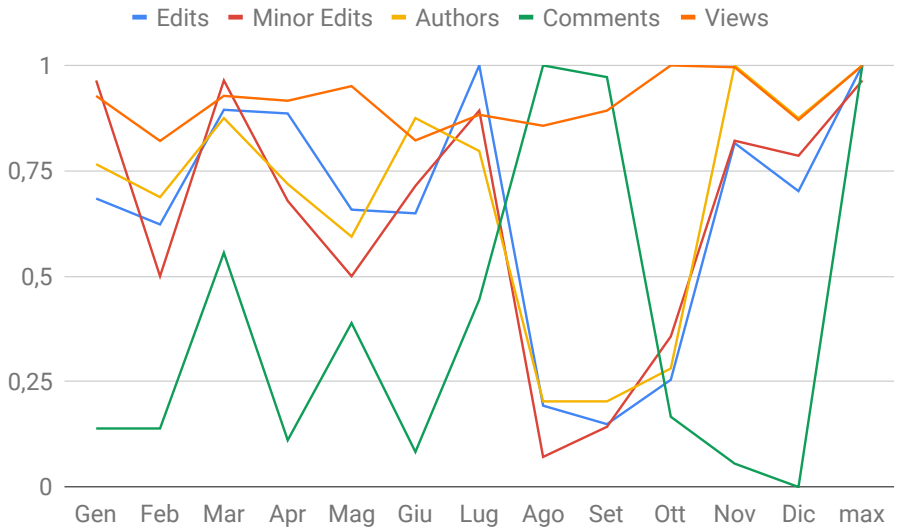

Figura 6.2: Indicatori attività della pagina "Artificial intelligence" nel 2016

Si può notare come in tutti e due gli anni gli indicatori derivati dallo storico revisioni quindi edits, minor edits, authors seguono lo stesso trend. Sono questi gli indicatori a cui bisogna guardare per vedere l'effettiva evoluzione della pagina, in quanto come evidenziato nella formula 2.1 del capitolo 2, l' evoluzione dei contenuti corrisponde storico revisioni.

Views e comments fanno parte dell'attività della pagina ma non propriamente della sua evoluzione. Infatti l'indicatore Comments deriva dal talk della pagina, anche se poi influirà nel suo cambiamento. Le views invece non influiscono sul cambiamento della pagina.

Ad un picco nei comments generalmente corrisponde (come nel Febbraio 2018) o si succede un trend positivo per gli altri indici, o comunque un loro mantenimento stabile. Lo si può notare evidentemente nel luglio 2018 in cui si ha picco massimo nei comments poi seguito da una impennata notevole in tutti gli altri indicatori. Questo è in linea con i risultati che ci si aspetta, in quanto ad un'alta attività nei commenti corrisponde organizzazione tra gli autori per nuovi edits o discutere/validare quelli già appoartati.

Nel grafico figura 6.3 a pagina seguente si fa il confronto tra attività della pagina nel 2016 e nel 2018. Attività è calcolata sommando gli indici normalizzati edits, authors, comments e views di ogni mese, normalizzando poi il risultato con il massimo tra tutti i mesi dei due anni. Si vede come nel 2018 si è avuta più attività, durante l'anno, tranne che per il mese di Ottobre.

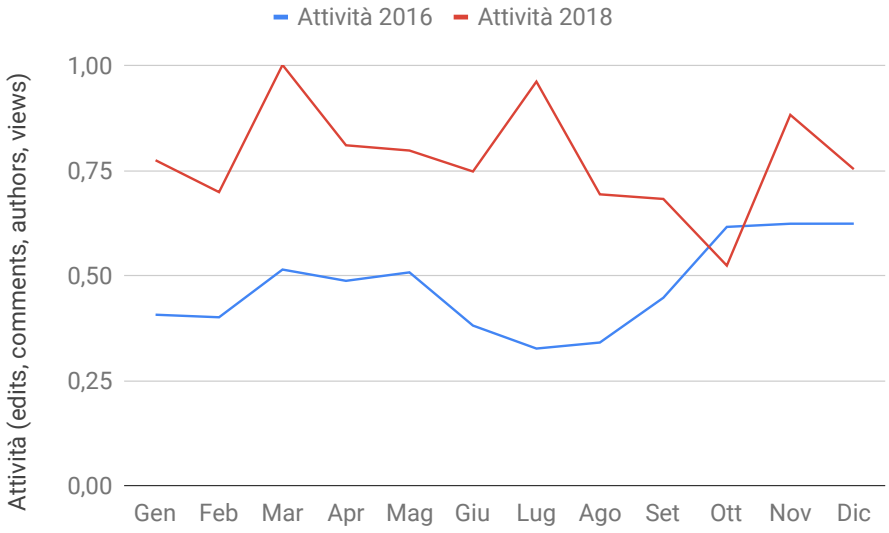

Figura 6.3: Attività della pagina "Artificial intelligence" nel 2016 e 2018

6.2 Pagine allineate

Si vogliono studiare le pagine in cui non c'è attività, quindi allineate (non in evoluzione) nella categoria Computer Science. Cerco pagine poco editate, poco commentate e molto visitate. La configurazione per gli indicatori usata per un timespan di un anno in questo caso è rappresentata nella tabella 6.4.

Edits	**Comments**	**Views**
0,20	2,7	150'000,*

Tabella 6.4: Configurazione indicatori di allineamento

La ricerca delle pagine avviene quindi con i seguenti parametri (nel flag t, per 2018 si intende l'anno 2018, cioè il timespan 20180101,20181231):

Flag	h	q	l	t	n	c	v
Value	en.wikipedia	Category:computer science	1	2018*	0,20	2,7	150'000,*

Tabella 6.5: Parametri di ricerca pagine in allineamento

Il risultato dela ricerca è rappresentato nella tabella 6.6.

Page	Edits	Comments	Views
Software configuration management	12	2	164'637
List of Unix commands	8	2	165'408

Tabella 6.6: Pagine allineate nella categoria Computer science nell'anno 2018

Si vuole studiare la sola pagina con meno edits e più views, cioè "List of Unix commands". Per farlo, si prendono le informazioni sugli indicatori quali edits, minor edit, authors, comments, views nell'anno 2018 ad intervalli (timespan) di 1 mese.

I risultati ottenuti sono riportati nei grafici nelle figure 6.4, 6.5 e 6.6.

Nella figura 6.4 si può notare come l'attività di edits sia molto bassa per un timespan di un anno. Solo in 5 mesi su 12 sono avvenuti edits, di cui in 1 mese 4 edits e negli altri 4 mesi un edit l'uno. Inoltre la maggior parte degli edits sono minor edits, quindi che non comportano modifiche sostanziali ma solo minori.

Nella figura 6.5 Si nota coma l'attività di discussione degli edits sia nulla, se non per 2 commenti avvenuti a Settembre. Questo vuol dire che non ci sono modifiche da apportare perché la conoscenza è allineata, quindi non si ha evoluzione dei contenuti.

Nella figura 6.6 si può notare come le views mensili si mantengano stabili senza cambiamenti rilevanti. Questo può essere un indicatore di come riguardo l'argomento della pagina non siano successi eventi nel mondo reale che abbiano creato disallineamento nel 2018.

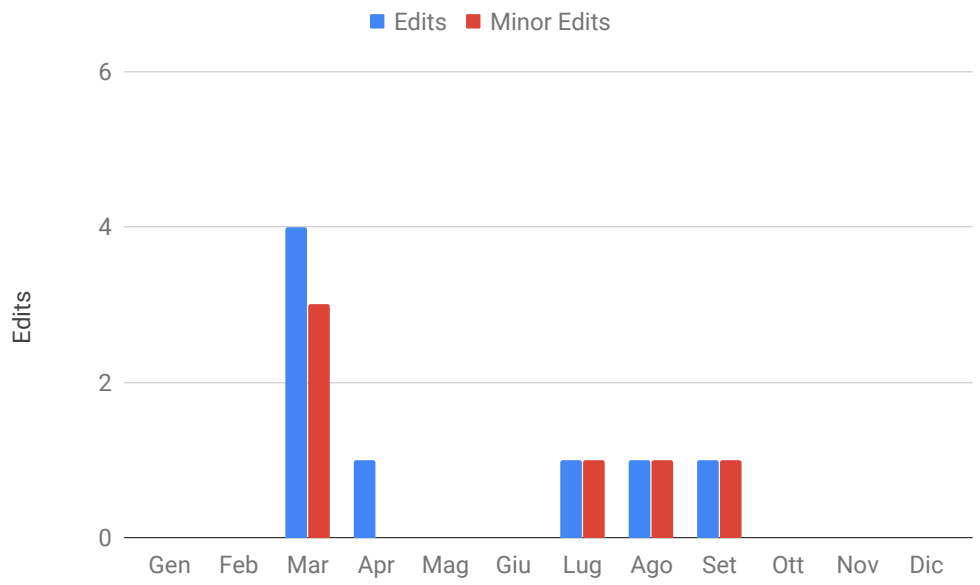

Figura 6.4: Edits e minor edits della pagina "List of Unix commands" nel 2018

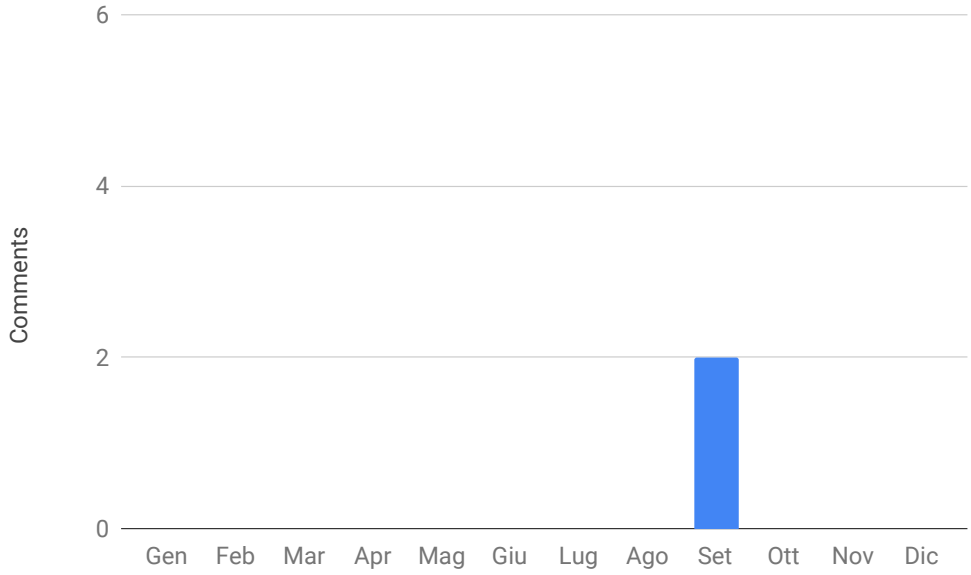

Figura 6.5: Comments della pagina "List of Unix commands" nel 2018

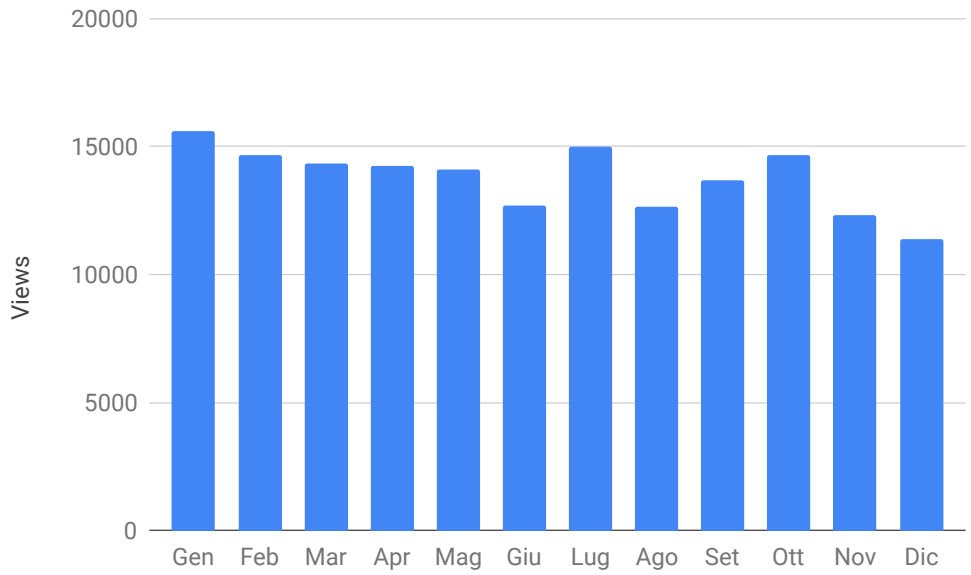

Figura 6.6: Views della pagina "List of Unix commands" nel 2018

6.3 Eventi nel mondo reale

Un caso interessante è l'impatto di eventi nel mondo reale sull'evoluzione dei contenuti in Wikipedia. Di seguito si analizza la pagina "Gravitational Wave" nell'anno 2016, anno in cui è stata annunciata la prima osservazione diretta delle onde gravitazionali. Pensando di non essere a conoscenza di tale evento e di voler "catturare" eventi che hanno inciso sull'evoluzione dei contenuti nella piattaforma, si adotta lo stesso modus operandi della sezione 6.1, cioè la ricerca delle pagine in evoluzione in un certo timespan tramite una configurazione di indicatori di alta attività. Si prende in considerazione la categoria "Physics", che al livello di sottocategorie 2 contiene la pagina "Gravitational Wave".

La ricerca delle pagine avviene quindi con i seguenti parametri:

Flag	h	q	l	t	n	c	v
Value	en.wikipedia	Category:Physics	2	2016*	365	50,*	200'000,*

Tabella 6.7: Parametri di ricerca pagine in attività per il caso eventi nel mondo reale

2016*: Per 2016 si intende l'anno 2016, cioè il timespan 20160101,20161231. Il risultato dela ricerca è indicato nella tabella 6.8.

Di seguito i risultati delle ricerca:

Page	**Edits**	**Comments**	**Views**
Ibn al-Haytham	468	127	111324
Dark matter	513	141	2308776
Physicist	369	132	229393
Gravity	457	82	1480299
RF resonant cavity thruster	691	489	660243
Gravitational wave	773	109	1582910

Tabella 6.8: Pagine in attività nella categoria Physics nell'anno 2016

Si studia ora la pagina con più edits, che risulta essere "Gravitational wave" e si vuole studiare la sua evoluzione nell'anno 2016 e nei 3 mesi precedenti. Per farlo, si prendono le informazioni sugli indicatori quali edits, minor edit, authors, comments, views nell'anno 2016 ad intervalli (timespan) di 1 mese. I grafici nelle figure 6.1 e 6.2 sono stati ricavati normalizzando ogni indicatore al suo valore massimo negli intervalli. Ne risulta il grafico in figura 6.7.

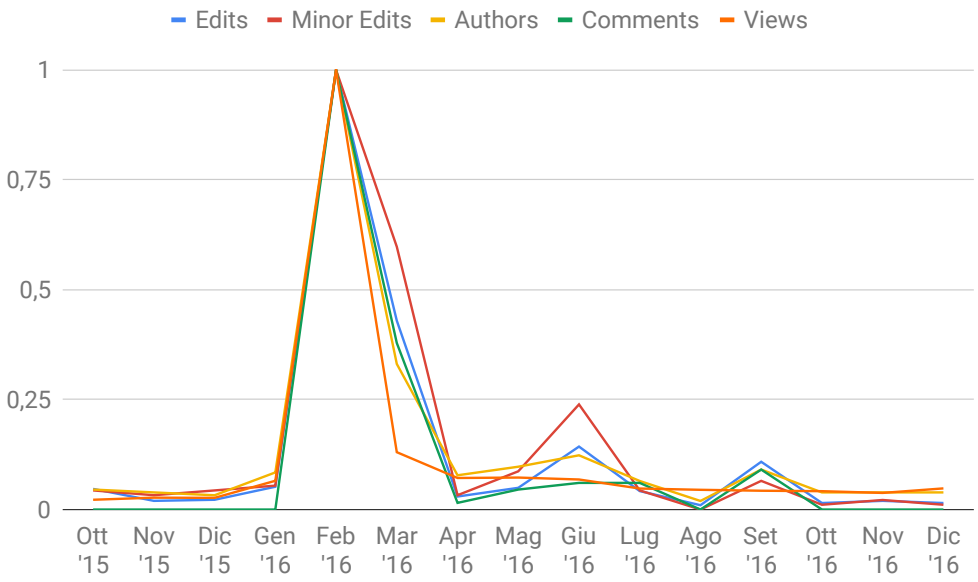

Figura 6.7: Indicatori di attività della pagina "Gravitational wave" nel 2016

Nella figura 6.7 si può vedere come l'evoluzione è compresa tra Gennaio e Ottobre 2016. Nel mese di Febbraio si può vedere un picco massimo preceduto da una situazione di allineamento e succeduto da una situazione con due picchi di "stabilizzazione" a Giugno e Settembre, per poi tornare ad una situazione di allineamento a partire da Ottobre. Il picco a Febbraio, rispetto al resto fa pensare ad un evento nel mondo reale per cui la conoscenza si sia disallineata, infatti nell'11 Febbraio 2016 è stata annunciata della prima osservazione diretta delle onde gravitazionali. Si può notare come da Gennaio gli indicatori subiscano lo stesso incremento fino al picco massimo, per poi subire una depressione, più accentuata per le views rispetto alle altre. Questo è dato dal fatto che le views sono assorbite in simultanea con l'avvenimento dell'evento mentre gli edits e comments vengono eseguiti più lentamente. I picchi seguenti al picco massimo sono di assestamento. Infatti il primo dei due è contraddistinto da alti indicatori di minor edits, che significano che le modifiche apportate nel picco massimo vengono perfezionate. Da Ottobre 2016 la conoscenza del mondo reale è allineata con quella della piattaforma, quindi si è tornati ad una situazione di stabilità, cioè una situazione di non evoluzione.

Si va a vedere ora la situazione nell'anno 2018, a due anni dall'evento (figura 6.8). Dalla figura 6.8 si evince che la situazione nel 2018 è stabile, a parte un picco in Aprile e Novembre per quanto riguarda i comments.

In figura 6.9 il confronto tra l'attività della pagina nel 2016 e nel 2018. Attività è calcolata sommando gli indici normalizzati edits, authors, comments, views di ogni mese, normalizzando poi il risultato con il massimo tra tutti i mesi dei due anni.

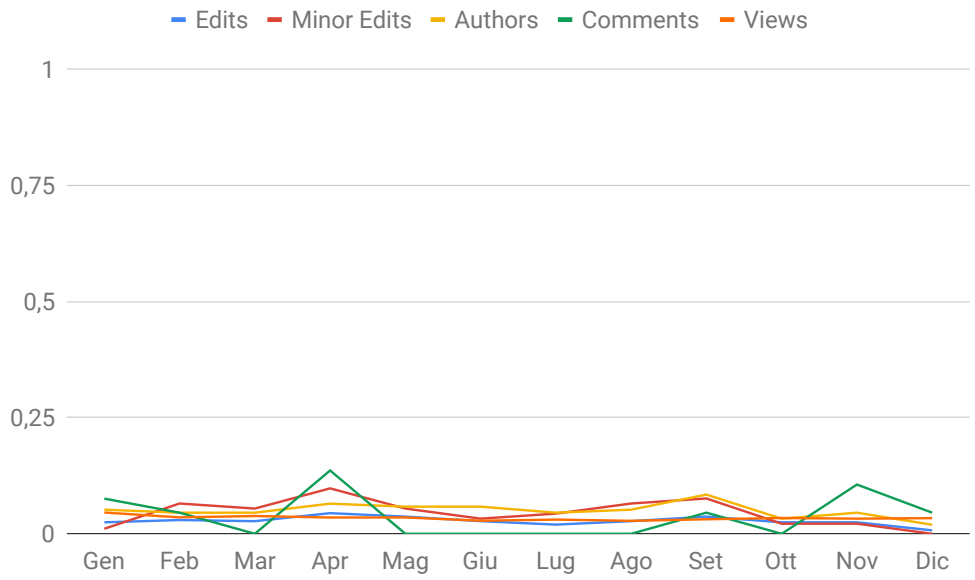

Figura 6.8: Indicatori di attività della pagina "Gravitational wave" nel 2018

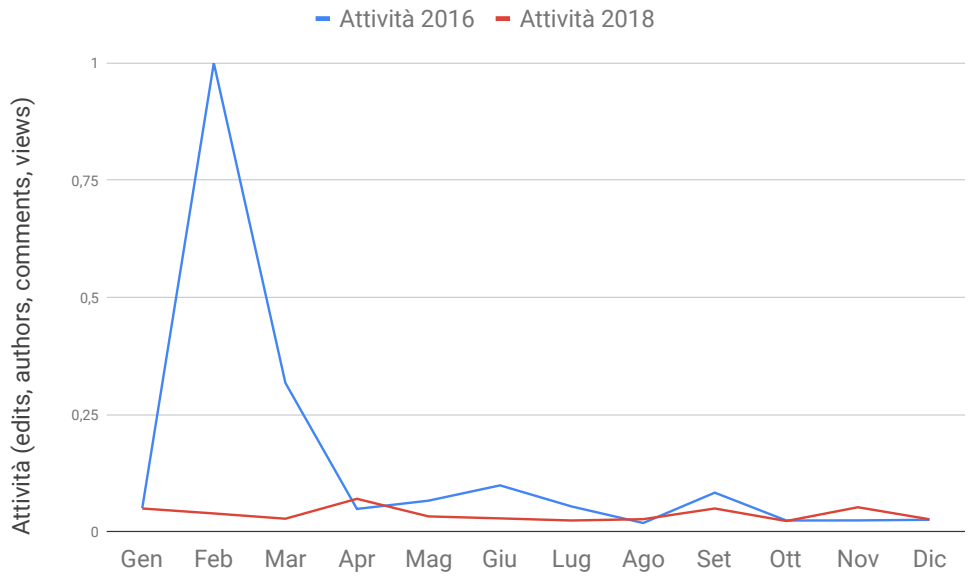

Figura 6.9: Attività della pagina "Gravitational wave" nel 2016 e 2018

Capitolo 7

Conclusioni e sviluppi futuri

In questo lavoro è stato proposto un modello per studiare l'evoluzione dei contenuti su piattaforme di editing collaborativo, cioè il cambiamento nei contenuti di piattaforme in cui lettori e scrittori sono figure sovrapposte cooperanti, ad esempio WIkipedia, che è la piattaforma presa in esame. Il modello studia l'evoluzione tramite situazioni di disallineamento tra la conoscenza del mondo reale e conoscenza espressa dai contenuti della piattaforma e lo fa tramite indicatori di potenziale disallineamento delle pagine Wikipedia quali edits, comments, views ed indicatori strutturali (links interni ed esterni, sezioni). Sono state discusse le caratteristiche e lo sviluppo del software (MoW) che implementa il modello e mostra alcuni esempi del suo utilizzo, concludendo con un'analisi dei dati (ricavati tramite MoW) riguardo gli indicatori.

Gli sviluppi futuri di questo lavoro sono molteplici e possono riguardare diversi aspetti. Potrebbero essere usate tecniche di machine learning per predire gli indicatori. Una volta analizzati gli indicatori su un arco temporale si posso dare i valori di questi in input ad algoritmi di machine learning per predire la loro evoluzione.

Un ulteriore sviluppo sono gli indicatori sul contenuto. Al momento lo studio dell'evoluzione avviene sono tramite indicatori di quantità, mentre in futuro potrebbe essere fatta anche con indicatori che riguardano la forma del contenuto, come ad esempio la qualità del testo.

L' analisi svolta in questo lavoro è stata fatta come esempio di utilizzo del modello e della sua implementazione software. Il prossimo passo è ampliare l'utilizzo del software su ampia scala e fare delle analisi più esaustive che riguardano grandi collezioni di pagine.

Un altro possibile lavoro è integrare le informazioni che riguardano le sorgenti esterne. Questo potrebbe essere utile per automatizzare il riconoscimento dell'evoluzione nella piattaforma data da eventi esterni. Per cui ad esempio se si mette in collegamento la piattaforma con Twitter e su Twitter c'è una determinata discussione su un argomento che è lo stesso di un contenuto della piattaforma, si cerca di capire che relazione c'è tra Twitter e piattaforma.

In ottica futura, un altro lavoro possibile è testare il modello su altri contesti diversi da Wikipedia. La piattaforma Wikipedia è un ambiente aperto in cui gli utenti intervengono per migliorare i contenuti, cosa che non si può dare per scontata in un ambiente chiuso come ad esempio la knowledge base di una azienda. Per questo sarebbe interessante studiare le analogie e le differenze tra ambienti aperti e chiusi nell'applicare il modello proposto in questo lavoro.

Bibliografia

[1] Martin Eppler. *Managing Information Quality. Increasing the Value of Information in Knowledge-Intensive Products and Processes.* 01 2009.

[2] Yang W. Lee, Diane M. Strong, Beverly K. Kahn, and Richard Y. Wang. Aimq: a methodology for information quality assessment. *Information Management*, 40(2):133 – 146, 2002.

[3] Jacob Marshak. *Economics of Information Systems (1971)*, pages 270–341. Springer Netherlands, Dordrecht, 1974.

[4] Diane M. Strong, Yang W. Lee, and Richard Y. Wang. Data quality in context. *Commun. ACM*, 40(5):103–110, May 1997.

[5] Besiki Stvilia, Les Gasser, Michael B. Twidale, and Linda C. Smith. A framework for information quality assessment. *Journal of the American Society for Information Science and Technology*, 58(12):1720–1733, 10 2007.

[6] Besiki Stvilia, Michael B Twidale, L Gasser, and Linda C Smith. Information quality discussions in wikipedia. In *Knowledge Management: Nurturing Culture, Innovation, and Technology - Proceedings of the 2005 International Conference on Knowledge Management*, pages 101–113. Graduate School of Library and Information science, University of Illinois at Urbana-Champaign, 2005.

[7] Richard Y. Wang and Diane M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, 12(4):5–33, 1996.

[8] Dennis M. Wilkinson and Bernardo A. Huberman. Cooperation and quality in wikipedia. In *Proceedings of the 2007 International Symposium on Wikis*, WikiSym '07, pages 157–164, New York, NY, USA, 2007. ACM.

[9] K. Wu, Q. Zhu, Y. Zhao, and H. Zheng. Mining the factors affecting the quality of wikipedia articles. In *2010 International Conference of Information Science and Management Engineering*, volume 1, pages 343–346, Aug 2010.

[10] Eti Yaari, Shifra Baruchson-Arbib, and Judit Bar-Ilan. Information quality assessment of community generated content: A user study of wikipedia. *Journal of Information Science*, 37(5):487–498, 2011.

[11] Mitsuo Yoshida, Yuki Arase, Takaaki Tsunoda, and Mikio Yamamoto. Wikipedia page view reflects web search trend. In *Proceedings of the ACM Web Science Conference*, WebSci '15, pages 65:1–65:2, New York, NY, USA, 2015. ACM.