

Metodi di Ingegneria della Conoscenza applicati alle Homepage delle scuole superiori italiane

# Corso didattico

* Ingegneria della Conoscenza [063507], Facoltà di “Informatica”
* A.A. 2022/23

# Gruppo di lavoro

* Vincenzo Di Bisceglie [745751] [v.dibisceglie3@studenti.uniba.it](mailto:v.dibisceglie3@studenti.uniba.it)

# Repository

* <https://github.com/vodibe/icon-74571>

# Sommario

Sommario

[Corso didattico 1](#_Toc150282145)

[Gruppo di lavoro 1](#_Toc150282146)

[Repository 1](#_Toc150282147)

[Sommario 2](#_Toc150282148)

[Introduzione 3](#_Toc150282149)

[Idea del progetto 3](#_Toc150282150)

[Metriche di usabilità già esistenti 3](#_Toc150282151)

[Metrica di usabilità adottata per questo progetto 3](#_Toc150282152)

[Elenco argomenti di interesse 4](#_Toc150282153)

[Costruzione del ground truth 5](#_Toc150282154)

[Sommario 5](#_Toc150282155)

[Decisioni di progetto 5](#_Toc150282156)

[Preprocessing del dataset delle scuole 7](#_Toc150282157)

[Rappresentazione dello spazio di ricerca con grafo e ricerca soluzioni 8](#_Toc150282158)

[Sommario 8](#_Toc150282159)

[Strumenti utilizzati: modello NaiveDOM 8](#_Toc150282160)

[Decisioni di Progetto 11](#_Toc150282161)

[Valutazione 11](#_Toc150282162)

[Sezione argomento 1 12](#_Toc150282163)

[Sommario 12](#_Toc150282164)

[Strumenti utilizzati 12](#_Toc150282165)

[Decisioni di Progetto 12](#_Toc150282166)

[Valutazione 12](#_Toc150282167)

[Conclusioni 13](#_Toc150282168)

[Bibliografia 14](#_Toc150282169)

# Introduzione

## Idea del progetto

**L’idea di fondo** da cui si è partiti per lo sviluppo di questo progetto è l’applicazione di alcuni metodi di Ingegneria della Conoscenza su un dominio di interesse, in questo caso l’usabilità di una pagina web. Questo richiede che prima si vada a circoscrivere un ambito di riferimento, che nel nostro caso, è l’insieme delle Homepage delle scuole superiori pubbliche italiane (09/2023).

## Metriche di usabilità già esistenti

Le **metriche rilevanti proposte in letteratura** e che potrebbero essere applicate nel contesto di questo progetto sono le Euristiche di Nielsen  [[1]](#b01) le WCAG 2.1  [[2]](#b02) per le quali però gli strumenti software ad essi correlati ([qui elencati](https://www.w3.org/WAI/ER/tools/?q=wcag-21-w3c-web-content-accessibility-guidelines-21)) non sono adatti alla natura di questo progetto perché analizzano il codice sorgente della pagina e danno consigli per rimediare le linee guida non rispettate. Altri strumenti controllano condizioni di accessibilità da parte di utenti con handicap (ad es. verificano che la palette di colori sia accessibile, che supportino l’interazione con hardware ausiliari, …)

Altre metriche rilevanti (SUS Score  [[3]](#b03)) non sono state prese in considerazione perché richiedono un campione di persone alle quali sottoporre un questionario.

## Metrica di usabilità adottata per questo progetto

Ai fini del progetto assumeremo che questa **nuova metrica di usabilità** corrisponde a un voto assegnato da una persona che non ha mai interagito con la Homepage prima d’ora, tenendo conto di quanto l’interfaccia sia ordinata e funzionale. **Dobbiamo precisare che questa metrica è vista come un qualcosa di condiviso dai visitatori (concetto oggettivo), infatti in questo progetto non si parla di valutazione (soggettiva) che uno specifico utente dà alla pagina**. Approfondiremo questa metrica nelle sezioni successive.

# Elenco argomenti di interesse

In questa sezione indichiamo le fasi del progetto e per ciascuna di esse gli argomenti coinvolti:

1. [Costruzione del ground truth](#_Costruzione_del_ground).  
   Poiché all’inizio non disponiamo di una valutazione per tutte le Homepage, ci immedesimiamo in un visitatore della pagina, ne osserviamo gli aspetti grafici e funzionali (in altre parole osserviamo il valore di alcune feature iniziali), e diamo una valutazione.
   1. Preprocessing del dataset delle scuole.
   2. Creazione di un dataset rappresentante il ground truth.
2. Emulazione del ground truth.  
   La fase 1 prevede un’osservazione diretta della grafica, e ciò ovviamente non può essere automatizzato, ma deve essere valutato con criterio. Pertanto in questa fase riproduciamo il ground truth utilizzando strumenti che si prestano meglio all’elaborazione e apprendimento automatico. Ci sono due sotto-fasi:
   1. Osservazione di caratteristiche della pagina ottenibili in modo automatico per ciascun sito, mediante **rappresentazione dello spazio di ricerca tramite albero**.
   2. Costruzione e valutazione di **modelli di apprendimento supervisionato** che, a partire dalle feature per ciascun sito (individuate al punto 2.1) simulano la sua valutazione.
3. Deduzione di informazioni utili e statistiche.
   1. Costruzione + Valutazione di modelli di apprendimento non supervisionato.
   2. Operazioni di deduzione con ragionamento relazionale e basi di conoscenza.

# Costruzione del ground truth

## Sommario

Si è ipotizzato che in generale, un visitatore osservando la pagina, può valutare il grado di usabilità con una scala [1, 5] con step di 0.1 punti.

* [1, 1.9]: **Sito estremamente confuso**  
  Non esiste un menu; la disposizione di tutti gli elementi è disordinata, per cui è difficile individuare le sezioni che l’utente vuole visitare.
* [2, 2.9]: **Sito confuso** Es. <https://www.galileiferrari.it/>   
  Esiste un menu; la disposizione di quasi tutti gli elementi della pagina è disordinata e la pagina dà l’impressione di essere troppo lunga.
* [3, 3.9]: **Sito accettabile** Es. <https://www.isii.it/>  
  Esiste un menu che reindirizza il visitatore a gran parte delle sezioni di suo interesse; la pagina però contiene un discreto numero di elementi non raggruppati e quindi confusionari.
* [4, 5]: **Sito ordinato** Es. <https://www.einsteinrimini.edu.it/>   
  Sito accettabile e che inoltre contiene pochi o nessun elemento non raggruppato.

## Decisioni di progetto

Si suppone che l’utente vada ad assegnare un valore di usabilità alla pagina ragionando su alcuni fattori. Per comodità, è utile raccogliere i fattori di decisione e la valutazione finale in un dataset che chiamiamo ds2\_gt.

La prima cosa che consideriamo vedendo una pagina web scolastica può essere la presenza di un menu, per cui si introduce la feature discreta **page\_menu\_or** che ne descrive l’orientamento.

A questo punto, viene introdotto un secondo fattore, il più rilevante, dovuto al fatto che nella quasi totalità dei siti scolastici ritroviamo il “trend” di inserire dei banner che linkano a una sezione del sito. Spesso, tali banner sono difficili da leggere e posti sulla pagina in modo disordinato. Per generalizzare (includendo qualsiasi contenuto multimediale, e quindi anche video) introduciamo la feature **page\_ungrouped\_multim**, che indica il numero di elementi multimediali che non compaiono raggruppati in un menu o in una sezione specifica della pagina.

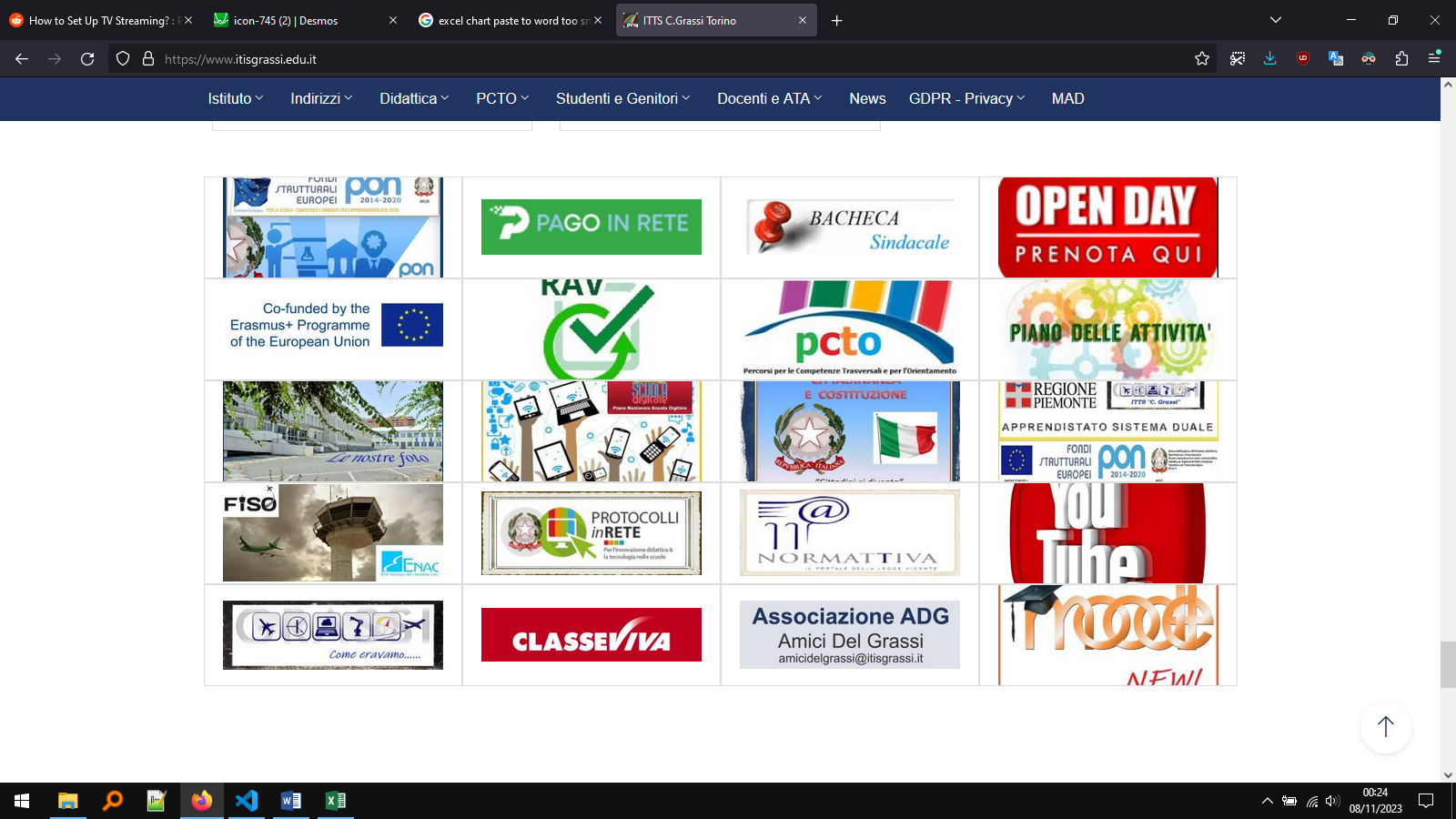


Figura . Alcuni banner.

Per ultimo, c’è la feature **page\_template**, utile a fornire un contesto in cui “inquadrare” la feature page\_ungrouped\_multim. Questo accade in quanto possono esistere più pagine che, seppur hanno lo stesso numero di elementi multimediali non raggruppati, risultano in una valutazione diversa perché basate appunto su template diversi.

Potremmo ipotizzare che la valutazione possa dipendere anche da quanto sia lunga la pagina, tuttavia un visitatore non viene mai a conoscenza dell’altezza precisa (in pixel). Pertanto non è stata considerata.

Riassumendo, i fattori di decisione sono i seguenti:

|  |  |  |  |
| --- | --- | --- | --- |
| *Fattore di decisione* | *Descrizione* | *Dominio* | *u.m.* |
| page\_template | Template adottato. 1,…,8=template ID 9=non segue un template |  |  |
| page\_menu\_or | Orientamento menu. 0=non esiste 1=solo orizzontale 2=solo verticale 3=orizzontale e verticale |  |  |
| page\_ungrouped\_multim | Elementi grafici non raggruppati. |  |  |

A quale pagina web sono associati questi fattori?

|  |  |  |  |
| --- | --- | --- | --- |
| *Feature* | *Descrizione* | *Dominio* | *u.m.* |
| school\_id | Codice della scuola. | Stringhe |  |
| page\_url | URL della pagina. | Stringhe |  |

La valutazione è la seguente:

|  |  |  |  |
| --- | --- | --- | --- |
| *Feature* | *Descrizione* | *Dominio* | *u.m.* |
| metric | Valutazione di usabilità della pagina. |  |  |

## Preprocessing del dataset delle scuole

Vedi: /agent/preproc/dataset\_creator.py

Una volta introdotta la metrica, è necessario parlare della fase di preprocessing del dataset iniziale. Il catalogo offerto dal MIUR raggruppa le informazioni su tutte le scuole (elementari, medie e superiori) pubbliche. Durante la fase di preprocessing si va a creare, in ordine, i seguenti DS:

1. **ds1**: <https://dati.istruzione.it/opendata/opendata/catalogo/elements1/?area=Scuole>  
   Per ora le features di questo DS (descritte [qui](https://dati.istruzione.it/opendata/opendata/catalogo/elements1/leaf/?area=Scuole&datasetId=DS0400SCUANAGRAFESTAT#tracciato)) non sono rilevanti.
2. **ds1\_clean** ottenuto inserendo solo le scuole superiori ed effettuando un preprocessing sull’URL che consiste nel vedere se il sito corrente è rintracciabile con una semplice richiesta HTTP. Se non lo è, ed inoltre il sito ha un TLD diverso da .edu, si sostituisce il TLD corrente con .edu. Se un sito non è rintracciabile neanche dopo aver effettuato la sostituzione, lo si esclude dal DS.
3. **ds1\_clean\_unique** ottenuto rimuovendo i siti duplicati. Operazione necessaria in quanto se un plesso scolastico offre più corsi di studio (e ha un singolo sito web), ciascuno degli indirizzi ricopre una riga nel ds1.
4. **ds2\_gt** ottenuto richiedendo i fattori di valutazione e la valutazione per ciascun sito presente in ds1\_clean\_unique.
5. **ds3\_gt** ottenuto inserendo tutte le features necessarie per addestrare un modello di apprendimento.
6. **ds3\_gt\_final**. E’ possibile che il ds3\_gt contenga qualche URL non valido, dovuto al fatto che il sito è in manutenzione, ha subito un cambio dominio o che faccia riferimento a una scuola superiore erroneamente catalogata nel ds1 (ad es. esclusivamente serale). Pertanto, queste righe vengono rimosse in questo nuovo DS. Le features presenti in questo DS sono descritte più avanti.

In sintesi:

* Siti di scuole superiori ma che non sono raggiungibili, oppure siti di altre scuole (elementari, …)
* (accanto a unique) Siti duplicati.
* (accanto a final) Siti raggiungibili ma non validi.
* Siti raggiungibili e validi.

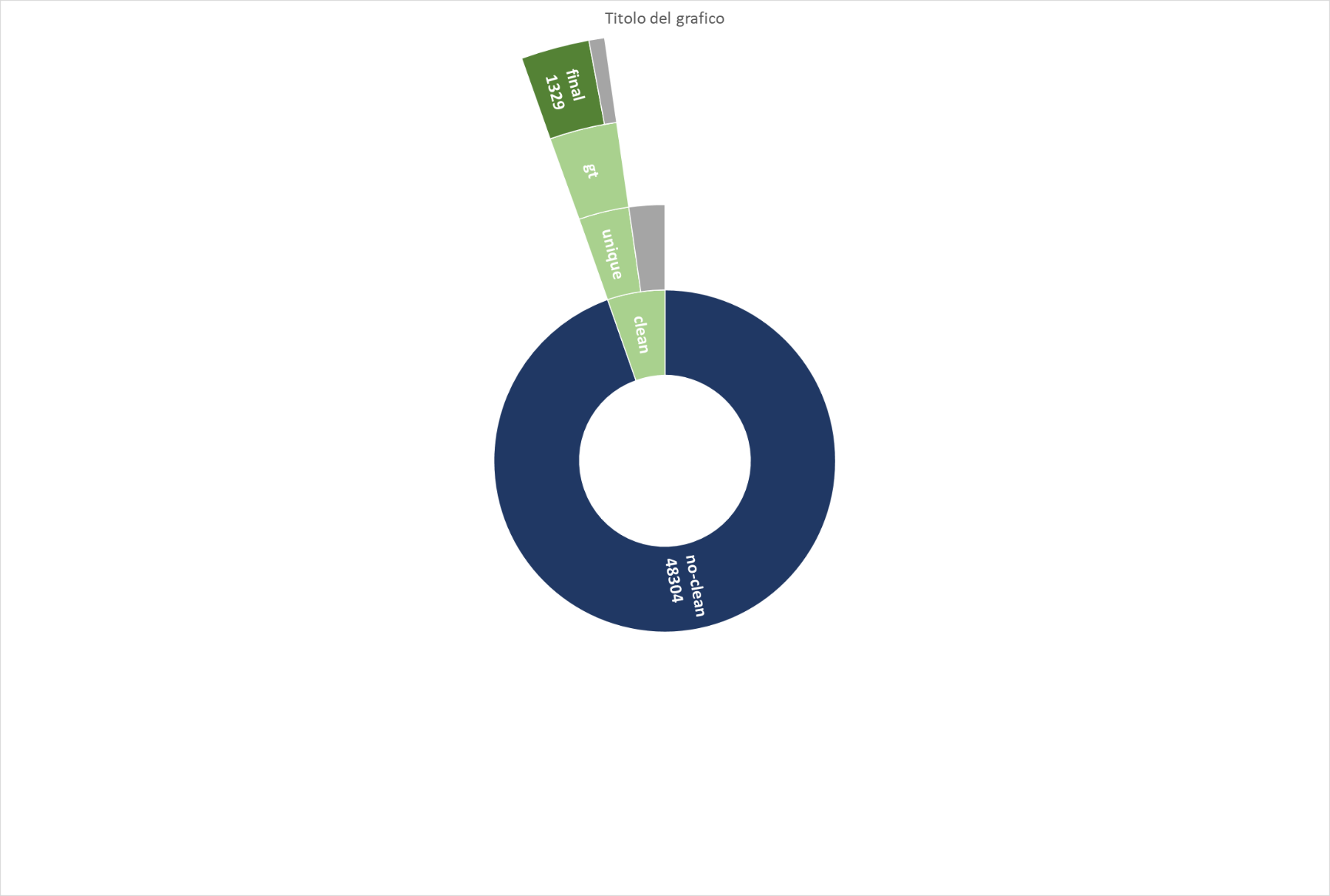


Figura . Proporzione del numero di elementi dei vari DS.

# Rappresentazione dello spazio di ricerca con grafo e ricerca soluzioni

## Sommario

Per eseguire il task di apprendimento del ground truth, dobbiamo individuare alcune features che possano essere osservate automaticamente a partire dalla pagina web. Per cui, notando che una pagina web equivale a un DOM, possiamo usare in questo progetto il concetto di rappresentazione dello spazio di ricerca tramite grafo.

## Strumenti utilizzati: modello NaiveDOM

Vedi: /agent/ndom/NaiveDOM.py

In questo progetto è stato introdotto il concetto di NaiveDOM (NDOM) che è un modello DOM semplificato della pagina ottenuto dal parsing del codice sorgente HTML.

### Struttura

Un NDOM è un grafo diretto e pesato, avente struttura ad albero. E’ tale per cui:

* **Ha numero finito di nodi ed è aciclico** (diretta conseguenza del fatto che è un DOM semplificato)
* Ciascun **nodo** è un elemento della pagina, e quindi è identificato univocamente dal suo XPath  [[4]](#b04).

A ciascun nodo sono associati una label (per fini di rappresentazione grafica) e le sue coordinate (x, y) all’interno della pagina renderizzata.

* Il **nodo radice** è l’XPath del tag <body>.
* I **nodi interni** sono gli XPath dei tag che contengono potenzialmente, tra i loro discendenti, un testo leggibile. Ad es. <body>, <header>, <section>, <nav> ecc…

Sono esclusi i tag <div>, visto che sono assai frequenti e non semplificano (ma complicano) la struttura del NDOM.

* I **nodi foglia** possono essere di tre tipi:
  + XPath dei tag che non contengono un testo leggibile, ad es. <img>, ecc…
  + XPath dei tag che contengono sicuramente un testo leggibile, ad es. <a>, <h1>, ecc…
  + Il testo leggibile, a patto che abbia una lunghezza breve.

Per quanto riguarda gli **archi** del NDOM e il loro costo, è necessario prima osservare direttamente un esempio di NDOM costruito per una pagina.

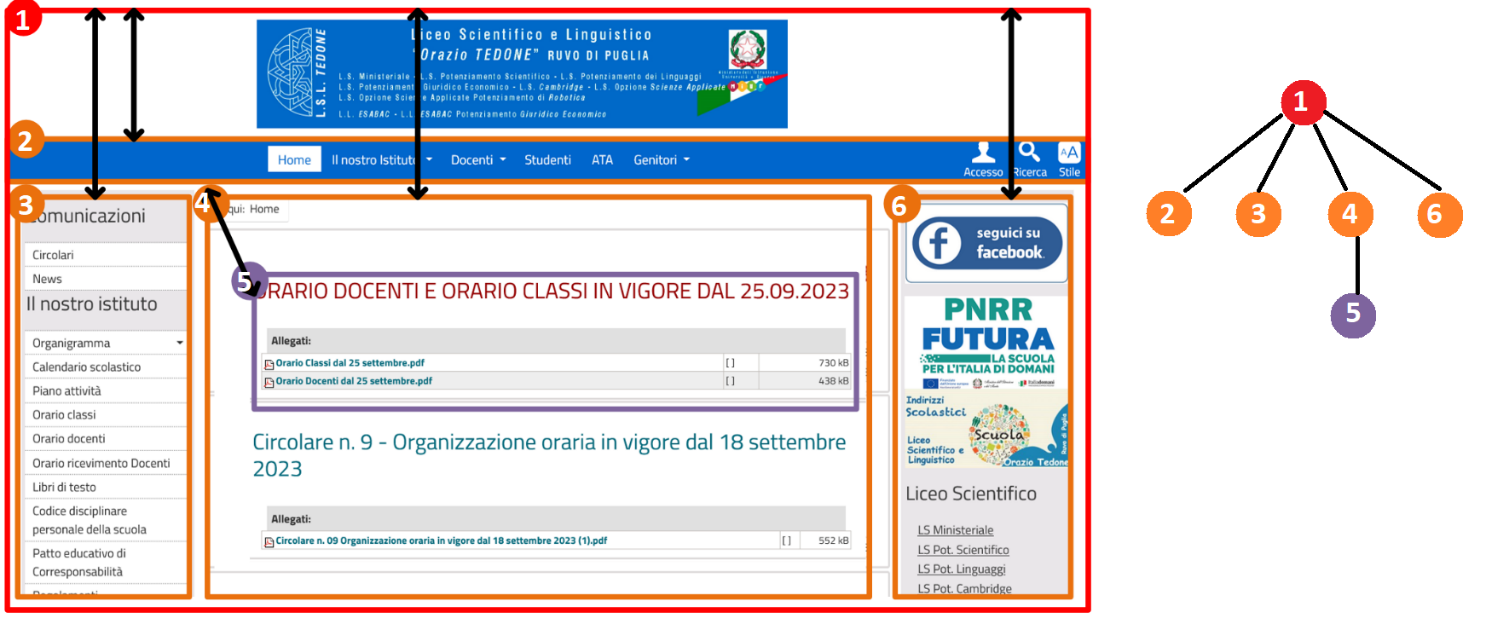


Figura . Esempio di NDOM per una pagina web.

Osservando lo screenshot, notiamo che il nodo radice <body> ha ovviamente coordinate (0,0). I figli diretti di <body> sono gli elementi arancioni e, solo in questo caso, la distanza tra padre e figlio è puramente verticale: questo è ovvio perché l’occhio umano inizia osservando dal basso verso l’alto. In tutti gli altri casi si provvede a calcolare la distanza euclidea.

Il **costo dell’arco tra padre-figlio** è una funzione della distanza padre-figlio (di seguito chiamata ), ed è calcolata in \_calc\_arc\_cost. Essenzialmente si riconduce alla seguente funzione:

Le feature della pagina che possiamo estrarre automaticamente grazie alla costruzione del NDOM sono:

|  |  |  |  |
| --- | --- | --- | --- |
| *Feature* | *Descrizione* | *Dominio* | *u.m.* |
| page\_load\_time\_ms | Tempo di caricamento della pagina. |  | ms |
| page\_width | Larghezza della pagina. |  | px |
| page\_height | Altezza della pagina. |  | px |
| NDOM\_nodes | Numero di nodi del NDOM associato alla pagina. |  |  |
| NDOM\_height | Altezza del NDOM associato alla pagina. |  |  |
| task1 | Costo in termini di usabilità per svolgere il task #1. |  |  |
| … |  |  |  |
| task8 | Costo in termini di usabilità per svolgere il task #8. |  |  |

### Calcolo di un task con algoritmo di ricerca

Vedi /agent/ndom/NaiveDOM.py (\_calc\_task\_cost)  
 /agent/ndom/NaiveDOMSearcher.py

Come si calcola il valore della feature taskx?

Innanzitutto, si deve **progettare un** **dizionario dei Task** e per ciascun Task deve essere presente un task ID e una lista di keywords. L’algoritmo di ricerca associato al NDOM trova il taskx se, partendo dal nodo radice individua un nodo la cui label contiene una qualunque keyword del taskx. Ad esempio:

TASKS\_DEFAULT = {

"task1": ["circolari", "comunicazioni", "circolare"],

"task2": ["organigramma", "organizzazione", "schema organizzativo"],

"task3": ["notizie", "news", "eventi"],

"task4": ["progetti", "progetto", "projects"],

"task5": ["regolamento", "regolamenti", "regolamentazione"],

"task6": ["amministrazione trasparente", "ammin. trasparente"],

"task7": ["registro"],

"task8": ["indirizzo", "i luoghi", "dove siamo", "contatti"]

}

**L’algoritmo di ricerca** proposto cerca di emulare il comportamento dell’occhio umano, e gode delle seguenti proprietà:

* Un percorso per il quale l’ultimo nodo ha profondità 0 o 1 (vale a dire, il nodo radice e tutti i percorsi dal nodo radice ai suoi figli diretti) viene aggiunto a una coda con priorità, in cui il percorso a costo minore sarà il primo ad essere esaminato. Questo è ovvio perché una persona passa ad esaminare prima le voci del menu principale rispetto alle voci del footer (che si trovano a fine pagina). Successivamente, gli alberi radicati nei figli diretti della radice vengono esaminati in modalità DFS.
* E’ completo, cioè certo di trovare una soluzione se essa esiste.
* E’ ottimale ??
* Non va in loop (diretta conseguenza della struttura del NDOM)
* Come l’algoritmo DFS, ha complessità di spazio ove è il branching factor e è la profondità del nodo goal; complessità di tempo .

Una volta aver descritto l’algoritmo di ricerca, gli **step per il calcolo della feature taskx** sono i seguenti (funzione \_calc\_task\_cost):

1. Applicare l’algoritmo di ricerca sul NDOM della pagina per trovare un percorso valido per il Task #x.
2. Se un tale percorso esiste, chiamiamo il suo costo (cioè la somma dei costi degli archi di cui si compone il percorso) e in funzione di calcoliamo il **costo totale del Task**.
3. Se non esiste un percorso, applicare un **costo di default del Task**.

## Decisioni di Progetto

Le librerie utilizzate in questa sezione sono Selenium per l’associazione tra nodo del NDOM e relative coordinate (fattibile solo dopo aver renderizzato la pagina) e Beautifulsoup per il parsing del codice sorgente e la creazione dei nodi.

In merito a Beautifulsoup, è necessario scegliere il parser lxml o html5lib, che sono gli unici in grado di gestire eventuali tag non chiusi.

Con questa libreria , inoltre, si è cercato di rendere la creazione del NDOM più efficiente possibile in termini di tempo richiesto, rimuovendo dal codice sorgente tag inutili (prima di darlo al parser).

In sintesi, di seguito vengono raccolti i parametri del NDOM (descritti nella sezione precedente) che è possibile cambiare qualora i risultati di un modello di apprendimento non siano abbastanza soddisfacenti.

* **Costo dell’arco tra padre e figlio** (in funzione della distanza).([Link al grafico](https://www.desmos.com/calculator/wf68sucipn?lang=it))
* **Dizionario dei Task.**
* **Costo totale del Task** (in funzione del costo del percorso trovato).([Link al grafico](https://www.desmos.com/calculator/aqm2x31us4?lang=it))
* **Costo default del Task.** ([Link al grafico](https://www.desmos.com/calculator/aqm2x31us4?lang=it))

## Valutazione

Paragrafi che richiamino (non spieghino, se standard) le metriche adottate   
+ tabelle sui risultati e loro discussione

# Sezione argomento 1

## Sommario

Un paragrafo che chiarisca la rappresentazione della conoscenza scelta per KB (modelli di ragionamento / apprendimento), dati, BK, ...

## Strumenti utilizzati

Breve: non serve spiegare come funzionano se implementano modelli ben noti   
(basta indicare dei riferimenti bibliografici)

Dilungarsi solo su eventuali modelli/algoritmi originali ideati dal gruppo

## Decisioni di Progetto

Configurazione dei componenti (e.g. API/librerie) utilizzati, ad es. parametri, soglie, ecc.   
e di eventuali metodi specifici utilizzati a tale scopo

## Valutazione

Paragrafi che richiamino (non spieghino, se standard) le metriche adottate   
+ tabelle sui risultati e loro discussione

# Conclusioni

Un paragrafo che riassuma le valutazioni e delinei possibili sviluppi, ad. es. problematiche non affrontate per questioni di tempo (per eventuali estensioni da parte di altri gruppi).

# Bibliografia

|  |  |
| --- | --- |
| [1] | [Online]. Available: https://en.wikipedia.org/wiki/Heuristic\_evaluation#Nielsen's\_heuristics. |
| [2] | [Online]. Available: https://www.w3.org/TR/WCAG21/. |
| [3] | [Online]. Available: https://en.wikipedia.org/wiki/System\_usability\_scale. |
| [4] | [Online]. Available: https://en.wikipedia.org/wiki/XPath. |