Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Matej Skyčák ID: 111652

Dokumentácia k semestrálnemu projektu

Vyhľadávanie informácii

Cvičenie: Pondelok 16:00

Školský rok: 2023/2024

Úvod

V tomto semestrálnom projekte sme mali za úlohu spraviť program, ktorý bude schopný nacrawlovať zvolené dáta a následne ich obohatiť o nejakú ďalšiu informáciu z Wikipédie. Nad týmito dátami bolo potom potrebné spraviť index a pomocou neho vedieť jednotlivé entity prehľadávať.

Dáta - ako zdroj dát odkiaľ som ich crawloval vybral stránku openlibrary.org. *Táto* stránka slúži ako databáza kníh, kde je možné ich nájsť obrovské množstvo.

Programovacie prostredie – na riešenie projektu som používal Visual Studio Code, v ktorom som program napísal v programovcom jazyku Python (3.12). Na funkcionalitu PyLucene som použil Docker, kde som si vytvoril kontajner a cez VSC som vedel projekt otvoriť v ňom. PySpark som si nainštalovať lokálne.

Crawlovanie

Dáta som crawloval zo stránky openlibrary.org, tu som si vybral kategóriu kníh Fantasy (funkcionalita je rovnaká aj pre ostatné kategórie) a následne som HTML stránky sťahoval a ukladal celé vo formáte, akom som ich stiahol. Podarilo sa mi nacrawlovať viac ako 3GB HTML stránok, čo predstavovalo 40 000 kníh.

Následne som z HTML súborov vyparsoval pre mňa dôležité informácie pomocou regulárnych výrazov, výstupom bol jeden CSV súbor. Formát CSV súbora je nasledovný: 'ID', 'Title', 'Author', 'Publisher', 'Date of Publishment', 'ISBN', 'Rating'. Jednotlivé polia v spomínanom poradí predstavujú: ID knihy na stránke openlibrary.org, autor knihy, vydavateľ, dátum vydania, ISBN a hodnotenie knihy.

Duplicitné záznamy som kontroloval na základe ID, ak už kniha s rovnakým ID bola spracovaná, tak som ju neuložil.

Parsovanie Wikipédie

Wikipédiu som parsoval pomocou PySparku (3.5.0), kde som si najprv vytvoril SparkSession. Jednotlivé dumpy Wikipédie sú uložené vo formáte XML, teda bolo do SparkSession potrebné pridať package na prácu s XML súbormi, ja som použil „com.databricks:spark-xml_2.12:0.14.0“. Potom som pre každú entitu pomocou regulárnych výrazov hľadal rok, kedy boli založené, tá sa nachádzala vždy v hlavičke v atribúte „founded“. Ak entita informáciu neobsahovala tak som ju neextrahoval a atribút som uložil s hodnotou null, čo znamenalo že hodnota toho atribútu ostala prázdna. Následne som dataset uložil vo formáte CSV v tvare „title“ a „Publisher founded in“, čo predstavovalo názov entity na Wikipédii a rok jej založenia.

Po extrahovaní informácii z Wikipédie mi vzniklo viacero CSV súborov, ktoré som zjednotil do jedného a ten som uložil. CSV súbor s entitami z Wikipédie som potom prepojil s CSV súborom s nacrawlovanými dátami. Prepojenie bolo robené na základe podmienky, že vydavateľ s nacrawlovaných dát sa musí rovnať entite z dát vyparsovaných z Wikipédie. Tým sa mi do

nacrawlovaných dát pridal atribút hovoriaci o tom, kedy bolo vydavateľstvo založené. Formát nových dát je vo formáte: 'ID', 'Title', 'Author', 'Publisher', 'Date of Publishment', 'ISBN', 'Rating', 'Publisher founded in'.

Celkovo som lokálne parsoval 5 dumpov, z ktorých som obohatil o dodatočnú informáciu niečo cez 400 záznamov. Lokálne spúšťať celý Wikipédia bolo moc náročné pre môj počítač, preto som písal pánovi prednášajúcemu. Začal som to už riešiť pomerne neskoro a odpoveď som dostal až v deň odovzdania a potreboval som svoj program ešte dodatočne upraviť. Z tohoto dôvodu som vychádzal iba z 5 dumpov, na ktorých som to prvotne spúšťal.

Indexovanie

Indexovanie som robil pomocou PyLucene a to pri oboch odovzdaniach projektu. Indexoval som pomocou všetkých polí s tým, že som ich do indexu aj ukladal, teda som ich potom vedel pri prehľadávaní z neho vytiahnuť. Na tokenizáciu slov som použil StandardAnalyzer.

Vyhľadávanie

Pri vyhľadávaní som taktiež použil StandardAnalyzer na parsovanie dopytu, potom som si zo súboru index otvoril a vytvoril IndexSearcher, ktorý som ďalej používal na vyhľadávanie v ňom.

Vyhodnotenie

Vyhodnocovanie metrík presnosti pre môj projekt nemá moc zmysel, keďže je u mňa problém čo si nastaviť ako celkovú množinu, o ktorú sa opierať. Aby som si porovnal výsledky vyhľadávania zo stránkou openlibrary.org, odkiaľ som crawloval dáta, potreboval by som všetky dáta, čo ja bohužiaľ nemám.

Jediné, čo mi ako tak dávalo zmysel bolo pozrieť sa na samotné výsledky, ktoré vracia môj program. Pri týchto výsledkoch som sa pozrel, či sú relevantné pre používateľa a na základe toho vypočítal presnosť pomocou vzorca **presnosť = relevantné výsledky / všetky výsledky**.

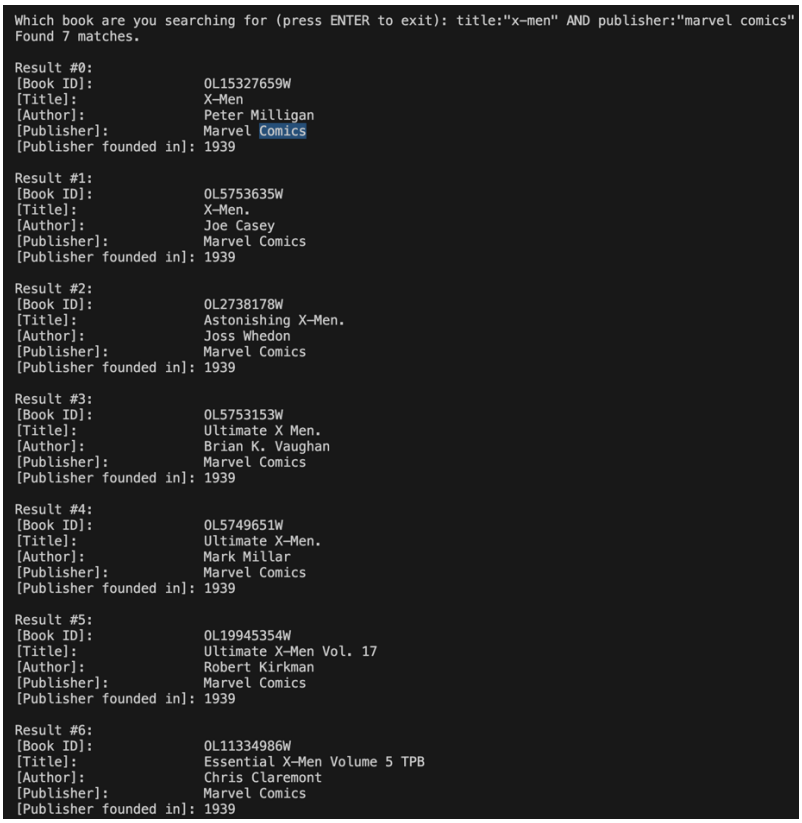

Figure 1 Výsledok dopytu: title"x-men" AND publisher:"marvel comics"

Na výsledku dopytu " title:"x-men" AND publisher:"marvel comics" " môžeme vidieť, že **presnosť** nášho program v takomto prípade je **100%**, keďže každý vrátený výsledok obsahuje nami zadané parameter. Týchto dopytov som otestoval viacero pričom pre každý som vypočítal 100% presnosť.

Ak by sme zadali dopyt v tvare " title:x-men AND publisher:"marvel comics" ", teda titul knihy nie je v úvodzovkách naše výsledky sa dosť zmenia. Počet vrátených výsledkov sa zvýšil o 6 na 13 a obsahuje tituly ako District X Vol. 1, Generation X Classic alebo Universe X Volume 2 TPB (New Printing). Výsledky sú správne, lebo lebo Query parser reťazec "x-men" rozdelí na dve slová "x" a "men". Čo pre používateľa nemusí byť úplne intuitívne a teda tieto výsledky pre neho nemusia byť relevantné, čo má za následok, že presnosť klesla na **56%**.

Používateľská príručka

Hlavná časť projektu sa spúšťa v skripte „**main.py**“, kde sa najprv vytvorí index nad nazbieranými dátami obohatenými o informáciu z Wikipédie a následne používateľ vie zadať dopyt v Query Parser formáte. Program sa ukončí zadaním prázdneho reťazca.

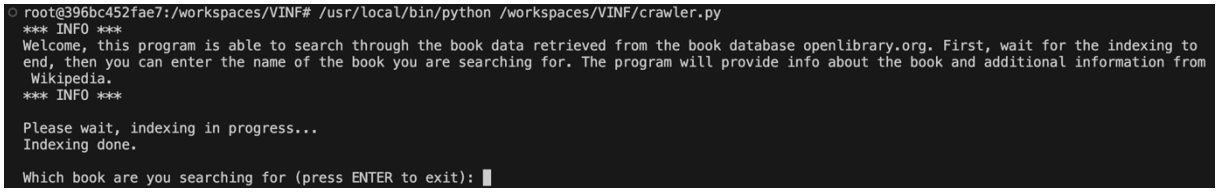

Figure 2 Používateľské rozhranie

Príklad dopytu, kde sa nám majú vrátiť knihy obsahujúce slovo „hercules“ v názve, od autora s menom obsahujúcim slovo „layton“, publikované vydavateľom s názvom obsahujúcim slovo „marvel“. Po spracovaní dopytu sa nám zobrazí informácia o počte vrátených záznamov a taktiež naformátované samostatné záznamy.

```
Which book are you searching for (press ENTER to exit): title:hercules AND publisher:marvel AND author:layton
Found 3 matches.

Result #0:
[Book ID]:              OL4732980W
[Title]:                Hercules
[Author]:               Bob Layton
[Publisher]:            Marvel Enterprises
[Publisher founded in]: Unavailable

Result #1:
[Book ID]:              OL4732981W
[Title]:                Hercules, prince of power
[Author]:               Bob Layton
[Publisher]:            Marvel Comics
[Publisher founded in]: 1939

Result #2:
[Book ID]:              OL4732982W
[Title]:                Hercules Prince of Power Circle
[Author]:               Bob Layton
[Publisher]:            Marvel Books
[Publisher founded in]: Unavailable
```

Figure 3 Príklad výsledku dopytu

Ostatné súbory:

- Crawler.py – skript služiaci na crawlovanie stránky openlibrary.org, stránky sú ukladané ako HTML súbory do priečinka „raw_html_pages“
- Parser.py – skript na parsovanie nacrawlovaných dát
- Spark_indexing.py – skript slúžiaci na parsovanie wikipédie a obohatenie nacrawlovaného datasetu o dodatočnú informáciu

Záver

Zadanie sa mi podarilo pomerne úspešne dokončiť a aj napriek problémom, napríklad s PyLucenom, ktorý mi nechcel fungovať, som s výsledným projektom spokojný. Podarilo sa mi nacrawlovať pomerne veľké množstvo dát, ktoré som následne sparsoval, naindexoval a vedel v nich prehľadávať. V druhej časti projektu som nazbierané dáta obohatil o dodatočnú informáciu o tom, kedy bolo vydavateľstvo, ktoré knihu vydalo, založené. Používateľovi je poskytnuté v rámci možnosti príjemné grafické rozhranie, vie si zadať dopyt na ľubovoľné knihy a prehľadne si výsledky prezerať. Nakoniec som sa snažil aspoň niekde nájsť niečo čo by som vedel vyhodnotiť z hľadiska presnosti, tak som sa pozrel na samotné výsledky a ako sú relevantné pre dopyt, ktorý zadal používateľ.