Sistemi Informativi Evoluti e Big Data – A.A. 2025-2026

Homework #3 – Neo4j

Introduzione - L'esercizio richiede di creare un grafo di proprietà in Neo4j partendo da un dataset reale di articoli scientifici pubblicati su arXiv nel dominio dell'Intelligenza Artificiale. Il dataset utilizzato è Artificial Intelligence Publication Trends ed è reso disponibile in formato CSV (`arxiv_papers.csv`), contenente un articolo per riga. Ogni articolo è descritto da un titolo (`title`), un abstract (`abstract`), la data/ora di pubblicazione (`published`), l'elenco degli autori separati da virgole (`authors`), il link alla pagina arXiv (`url`). Non sono presenti informazioni su riviste, istituzioni o citazioni, che non verranno quindi richieste nel seguito. Sono inoltre disponibili altri due file in formato CSV: (a) il file `papers_with_topics.csv`, che associa ad ogni url l'ID del topic principale (`topic_main`), il nome del topic (`topic_main_label`) e la probabilità che il suddetto topic sia adeguato all'articolo[1] (`topic_main_probability`), un elenco di tre topic tra i più probabili, incluso quello principale (`top_topics`) con relative probabilità (`top_probabilities`) ed etichette (`top_topic_labels`); (b) il file `topic_definitions.csv`, che contiene, per ogni topic, l'ID (`topic`), l'etichetta (`name`), il numero di documenti a cui è associato quel topic (`count`), la lista delle parole chiave che meglio identifica quel topic (`representation`) e l'abstract che meglio rappresenta il topic (`representative_docs`).

Descrizione del compito - Scaricare il dataset e, utilizzando Python e Cypher, creare un nuovo database in Neo4j:

- creare un nuovo nodo `Paper` per ogni articolo, con le proprietà `title`, `abstract`, `date` (solo la data di pubblicazione, non l'intero timestamp presente nel file CSV), `url`;
- creare un nuovo nodo `Author` per ogni autore, con `name` come proprietà;
- creare dei nodi di tipo `Topic` estratti dagli abstract dei paper;
- creare diversi tipi di relazioni tra i nodi sopra-citati, ovvero `:AUTHORED` tra un nodo di tipo `Paper` e un nodo di tipo `Author`, `:ON_TOPIC` tra nodi di tipo `Paper` e nodi di tipo `Topic` (sulla relazione, va riportata la probabilità che il topic rappresenti il contenuto dell'abstract), `:COAUTHORED_WITH` tra attori che hanno lavorato agli stessi paper (con indicazione del numero di paper che hanno in comune), `:SIMILAR_TOPIC`, tra nodi di tipo `Paper` che condividono gli stessi topic; in particolare, la relazione di similarità tra articoli sulla base dei topic che condividono va calcolata tramite l'utilizzo della similarità di Jaccard pesata (o similarità di Ruzicka), per tener conto della probabilità:

$$sim_T(p,q) = \begin{cases} 1 & se\ p = q \\ \dfrac{\sum_{i \in U_T} \min(p_i, q_i)}{\sum_{i \in U_T} \max(p_i, q_i)} & altrimenti \end{cases}$$

[1] Per l'estrazione dei topic da un insieme di abstract, è stata utilizzata Python la classe `BERTopic` (libreria `bertopic`) e l'integrazione di tecniche di *embedding semantici* (tramite la classe `SentenceTrasnformer` della libreria `sentence_transformers`); i file in formato CSV sono tutti forniti in allegato a questo homework; tuttavia, viene lasciata allo studente la facoltà di testare ed eventualmente personalizzare lo script Python utilizzato per estrarre i topic dagli abstract (tale script è a sua volta allegato a questo homework). Per dettagli sulle librerie utilizzate, si rimanda alla documentazione associata.

dove p e q sono due articoli, p_i è la probabilità che il topic i sia assegnato a p e q_i è la probabilità che il topic i sia assegnato a q, U_T è l'unione dei topic assegnati a p e/o a q.

Eseguire le seguenti interrogazioni sul database appena popolato:

1. Visualizzare lo "schema" del database (tipi di nodi e di relazioni tra i nodi) tramite Neo4j Desktop.
2. Estrarre i primi 5 autori come numero di articoli di cui sono autori o co-autori.
3. Estrarre l'andamento temporale del numero di paper pubblicati nel corso degli anni e visualizzare il risultato in un grafico a barre in Python.
4. Trovare il percorso più breve tra due autori (a scelta dello studente) e visualizzarlo come sottografo tramite Neo4j Desktop.
5. Calcolare la similarità tra tutte le possibili coppie di autori a_1 e a_2, applicando la seguente formula (dove P_1 e P_2 rappresentano l'insieme degli articoli associati ai due attori, rispettivamente, e |.| rappresenta la cardinalità di un insieme):

$$Sim(a_1, a_2) = \frac{1}{|P_1| \cdot |P_2|} \sum_{p_i \in P_1} \sum_{p_j \in P_2} sim_T(p_i, p_j)$$

6. Visualizzare come grafo la rete di co-authorship filtrata ai primi 20 autori più prolifici.

Dettagli sulla consegna – Predisporre un notebook Python (o un file .py) in cui è riportato lo svolgimento di tutti gli esercizi e un file PDF in cui sono riportati i commenti sui risultati o e l'output di Neo4j Desktop, laddove esplicitamente richiesto (per esempio, l'immagine di un sottografo o del percorso più breve tra due nodi). Includere tutti i file in uno zip e caricare l'archivio su Moodle.