Ontology-based query graph matching

Doan Quang Tu *

Institute of Information Technology and Electronics, Academy of Military Science and Technology, 17 Hoang Sam, Nghia Do, Hanoi, Vietnam.

*Corresponding author: Upgradetome@gmail.com

Received 16 Apr. 2025; Revised 2 Jun. 2025; Accepted 9 Jun. 2025; Published 25 Aug. 2025.

DOI: https://doi.org/10.54939/1859-1043.j.mst.105.2025.130-138

ABSTRACT

The requirement for business competition among organizations has become more complicated in the past decade. One of these issues is how to react and response to queries from clients with the aim of producing quick and timely decision-making. To address this problem, this paper proposes a framework for identifying the frequent queries from multiple sources. The main contribution of this work is (1) query matching based on the ontology. In fact, query matching has been widely explored using various methods, most of which are based on a labeling approach and utilize isomorphism to map a query to subgraphs in a graph database. However, these traditional methods may not capture enough semantic similarity when mapping terms together. Ontology-based matching can address this issue more effectively than mere labeling isomorphism. In this work, three pruning techniques and their combination within dynamically weighted sets of vertex-associated elements are proposed. In addition, (2) a subgraph/subontologies clustering technique is recommended to extract the results of the framework. This technique is based on an existing clustering method, but requires improvements in the data processing step to be applicable to clustering subontologies/subgraph objects. Finally, (3) we experimentally verify the effectiveness and efficiency of our pruning techniques using synthetic data and compare the techniques themselves.

Keywords: Ontology; Query matching; Semantic similarity; Pruning techniques.

1. INTRODUCTION

Companies face challenges in market understanding and product optimization, often addressed through business intelligence tools that analyze sales trends and customer demographics. Quick decision-making relies on optimizing queries and predicting future trends. Data warehouses (DW) play a key role in managing data from multiple sources, with frequent queries identified to improve DW efficiency. However, frequent queries should be identified based on semantics, not just structure, for more meaningful results. Recent research suggests representing queries as graphs stored in graph databases, though traditional methods often miss semantic matches. Mapping query graphs to an ontology can address this issue. These approaches may miss the semantic aspects and may overlook important/necessary matching of semantically identical terms because there may be no identical labels between them. To address this limitation, query graphs can be mapped to an ontology to ensure the semantic characteristics of the matching.

This paper presents a method for matching a query graph from an ontology using a similarity function that accounts for both vertices and edges. Our approach employs a "query-trend-filter" framework, which processes queries and identifies trends through clustering. Our key contributions include the definition of a vertex's related set based on its connected relationships, along with a dynamic weighted set similarity function for measuring concept distances. We introduce pruning techniques leveraging this related set to efficiently filter and select candidate vertices from the ontology. We propose a "query-trend-filter" framework that transforms domain-specific queries into unified query graphs, matches them with sub-ontology candidates, and clusters the extracted sub-ontologies to identify query trend centroids. Our approach is validated through experiments that evaluate the clustering technique using recall and precision metrics.

This paper presents a novel method for matching query graphs from an ontology, offering a more suitable approach than existing methods. Section 1 illustrates this with two examples. Section 2 reviews related work, including sub-graph and ontology-based matching. Section 3 introduces a framework for identifying trending queries, exploring pattern characteristics like frequent patterns and pruning techniques, discussed in section 4. Section 5 details experiments, and section 6 concludes with future work.

2. RELATED WORK

Traditional subgraph matching methods rely on exact or approximate matches of vertices and edges. Early approaches [1] as well as those by Krissinel et al. [2] and Abu-Khzam [3] use backtracking or maximum common subgraphs, while others apply maximum clique techniques for approximate matching [4, 5]. These focus on general graphs. More recent work, such as Mottin et al. [6], introduces user-data-driven query matching, and [7] improves matching by minimizing graph distances. Ontology has also been widely used in graph query research. Cakmak and Ozsoyoglu [8] apply it to graph mining, and [9] use it for keyword search. Further studies enhance query processing with ontological structures [10, 11], with Cakmak et al. [11] mining RDF patterns via SPARQL and class hierarchies. More recent approaches, such as [12] and [13], interpolate ontologies in template matching or index data graphs via ontology graphs. Our approach differs by using ontology graphs directly as query references, removing the need for data graphs. Sub-ontology extraction was first proposed by Wouter et al. [14, 15] through rule-based selection, later improved by Minimum and Maximum Extraction methods [16, 17] to balance size and relevance. We build on this by extracting key vertices and relations, preserving structure and semantics while enabling scalable query matching.

3. SOLUTION FRAMEWORK

3.1. Preliminaries

This subsection defines terms and techniques related to and contributing to a clearer understanding of the approach in this paper.

Definition 1 (Query graph). *A query graph Q is a directed graph defined as Q = (Vq,Eq) where Vq is the set of query vertices, and Eq is a set of vertex pairs or edges.*

In real-life applications, an ontology including main elements (concepts, properties of concepts, property mappings and relationships) can typically be represented under a unite ontology graph. In this paper, an ontology graph is simply defined to adapt the approach of query matching as below.

Definition 2 (Ontology graph). *An ontology graph O = (Vo,Eo) where Vo is a set of vertices that represent terms, concepts or entities, **and** Eo is a set of edges that represent semantic relationships between vertex pairs.*

Definition 3 (Vertex-related set). *If v is a vertex of a query graph or an ontology, then S(v) = (e1,...,em) is the set associated with v, where ei is an element that represents an edge label related to v, for example, the vertex Herb may have the following elements: "found_in", "is_a" and "subclass_of".*

Definition 4 (Sub-ontology matching to query graph). *Given a query graph Q with n vertices (q1,...,qn), an ontology graph O, and a user-specified similarity threshold τ, a subgraph match of Q is a subontology Go of O containing n vertices (o1,...,on) of Vo, if the following conditions hold:*

- *There exists a mapping function f such that $f(q_i) = o_j$ for each $q_i \in V_q$ and $o_j \in V_o$;*
- *$sim(S(q_i),S(o_j)) \geq \tau$, where $sim(S(q_i),S(o_j))$ is a set similarity score between $S(q_i)$ and $S(o_j)$;*
- *For an edge $(q_i,q_k) \in E_q$, there is $(f(q_i),f(q_k)) \in E_o$ $(1 \leq k \leq n)$.*

Note that, in practice, a function sim(...) can be chosen by an application domain, for example, domain experts, a semantic Web mining application or Web mining. While research on a novel similarity function is beyond the scope of this paper, the weighted Jaccard similarity, which is one of the most popular similarity functions, is chosen.

Definition 5 (Similarity function). *For element sets S(q) and S(o) of vertices q and o, the weighted Jaccard similarity between S (q) and S (o) is*

$$sim(S(q),S(o)) = \frac{\sum_{e\in S(q)\cap S(o)} \mathrm{W}(e)}{\sum_{e\in S(q)\cup S(o)} \mathrm{W}(e)} \quad (1)$$

where W(e) is the weight of element e, W(e) ≥ 0.

In practice, the Jaccard similarity function can be converted from other widely used similarity functions, such as *cosine similarity*, *Hamming distance*, and *overlap similarity*. Thus, the Jaccard similarity in definition 5 can be used with a threshold τ, allowing the approach for query matching in this paper to use it as a constant lower bound, as introduced in section 4.

3.2. A framework for identifying query trends

In this subsection, we present a framework for identifying query trends using our approach to We present a framework for identifying query trends using ontology-based query extraction, as illustrated in figure 1. In **step 1**, queries from multiple databases across different servers are represented within a unified domain. These queries reflect business-related trends and intentions. In **step 2**, they are transformed into query graphs for uniform manipulation. Our proposed method for query graph matching is introduced in **step 3**, where we apply pruning techniques based on element set similarities and their combinations. A matching algorithm updates the state iteratively upon each vertex match. While **step 4** builds on prior subgraph matching techniques (see subsection 2.1), our emphasis lies in **step 5**, where we introduce a novel K-medoids-based clustering process tailored for graph objects. Finally, **step 6** (detailed in section 5.3) analyzes the clustered results to extract and interpret query trends.

4. SUBGRAPH QUERY MATCHING PROCESSING

4.1. Candidate filter - Pruning techniques

The purpose of this section is to find a candidate vertex in an ontology G_o for a vertex q in a query graph G. Recall Definition 4 (Sub-ontology matching to query graph), if a vertex o in ontology graph G_o, then $sim(S(q),S(o)) \geq \tau$. Therefore, this section focuses on finding candidate vertices *o* and *q* such that $sim(S(q),S(o)) \geq \tau$. Moreover, in the approach, weights are assigned to the elements dynamically, which does not make a change in the *inclusion relation* between two sets. The inclusion relation of two element sets *S(v)* and *S(v')* of vertex v and v', respectively, is defined by the relation $S(v) \subseteq S(v')$, which means *S(v)* is a subset of *S(v')*. Taking advantage of the inclusion relation, a similarity upper bound is introduced below to contribute to the pruning techniques. These contributions are presented later in subsections, and their demonstrations as well as examples are presented in section 6.

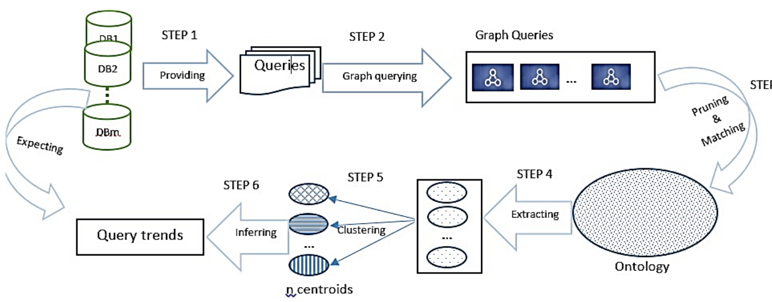

Figure 1. *Framework for identifying query trends.*

Definition 6 (Upper bound for similarity). *For element sets S(q) and S(o) of vertices q and o, an upper bound for similarity (UB-S) is*

$$UB(S(q),S(o)) = \frac{\sum_{e\in S(q)} \mathrm{W}(e)}{\sum_{e\in S(q)\cup S(o)} \mathrm{W}(e)} \geq sim(S(q),S(o)) \tag{2}$$

where W(e) is the weight of element e, W(e), and the function sim() is denoted by equation (1).

Intuitively, for a query given the value of $\Sigma_{e\epsilon S(q)}$W(e) does not change, the upper bound UB(S(q), S(o)) will only depend on S(o). In other words, UB(S(q), S(o)) is in inverse proportion to the size of S (o). Therefore, for every set S(o) ⊑ S (o'), we have: UB(S (q),S (o')) < UB(S (q),S (o)) < τ.

4.2. Frequent-pattern-based pruning

Definition 6, along with the characteristics of the pattern space discussed above, leads us to infer the following rule.

Rule 1. *For a query vertex q and a pattern space with number of P_s representing for pat- terns, if UB(S(q),P) < τ, all vertices that contain all element**s** in P can be pruned.*

Explanation. The rule above controls how the pattern space distributes patterns. Since all of Ps in the space are the subsets of descendant P_s', all elements in P are associated with vertices containing P_s'. Therefore, according to Rule 1, all the vertices within P_s' (vertices that contain only the element set P) can also be safely pruned.

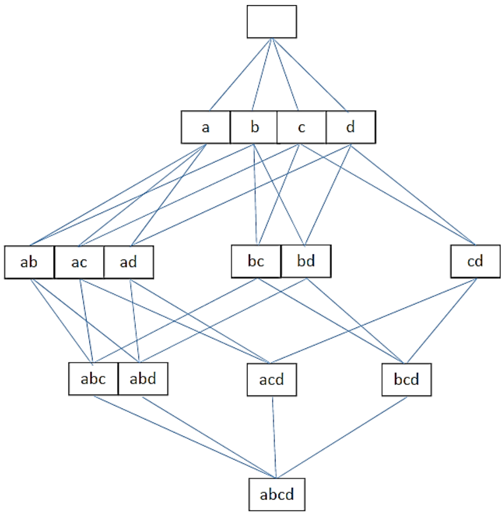

Figure 2. *An example of the pattern space.*

4.3. Prefix-filtering pruning

Consider S(q) as an element set associated with vertex q, where the elements are arranged according to their weights, as identified by domain experts, in descending order. In this paper, the k-prefix of S(q) denotes the first k elements in the set S(q). The goal of the technique is to find the maximum value of k such that if S(q) and S(o) have no overlap in their k-prefixes, S(o) can be pruned. The method to find k is based on the similarity threshold, as presented below.

Finding k-prefix of S(q), where S(q) consists of elements arranged in descending or- der of weights, the first element of S(q) (the element with the largest weight) is removed. After that, the new/remaining set S'(q) is compared with S(q) to determine if it meets the similarity threshold

$$\sum_{e\in S'(q)} \mathrm{W}(e) > \sum_{e\in S(q)\cap S(o)} \mathrm{W}(e) \tag{3}$$

with S(q). In particular, the sum of weights of all elements associated with the query vertex q is denoted as $\|S(q)\| = \Sigma_{e \in S(q)} W(e)$. The removal process will stop when $\|S'(q)\| \leqslant \tau * \|S(q)\|$. The value of k is denoted by |S'(q)| - 1, where |S'(q)| represents the number of elements in S'(q). For any set S(o) that does not contain the elements in the k-prefix of S(q)'s (i.e., S(o) and S(q) do not overlap on the elements of S'(q), or their overload is just a subset of S'(q), or a set of elements having smaller weights than those in S'(q)), we have:

We know that:

$$\sum_{e \in S(q)} \mathrm{W}(e) < \sum_{e \in S(q) \cup S(o)} \mathrm{W}(e) \tag{4}$$

$$\frac{\|S'(q)\|}{\|S(q)\|} = \frac{\sum_{e \in S'(q)} \mathrm{W}(e)}{\sum_{e \in S(q)} \mathrm{W}(e)} \tag{5}$$

so from (3),(4),(5) and (1):

$$sim(S(q), S(o)) < \frac{\|S'(q)\|}{\|S(q)\|} < \tau \tag{6}$$

Hence, S(q) and S(o) will not meet the set similarity threshold, or vertex o does not match vertex q.

Rule 2. *For a query vertex q, its associated set S(u) and a frequent pattern P in a pattern space, if P is not a one-frequent pattern (or one of its descendants) in the k-prefix of S(q), all vertices containing all elements of P can be pruned.*

Explanation. For each vertex o containing a pattern P or $P \subseteq S(o)$. According to the principle of prefix filtering and Rule 1 presented above, all vertices that contain pattern P, where P is not a one-frequent pattern in S(q)'s k-prefix can be pruned. Thus, when the p-prefix of S(q) is found, only the descendants of the relevant one-frequent patterns in p-prefix can be accessed, and others are pruned.

4.4. Pattern-size-based pruning

We observe that the element set S(q) of a query vertex is unlikely to be similar to very large or very small frequent patterns. Thus, pattern size can be used to prune irrelevant vertices. We define a length upper bound (LU), representing the maximum size of candidate sets for q. To determine LU(q), elements are incrementally added to S(q)S(q)S(q) in order of increasing weight. After each addition, if the similarity sim(S(q),S′(q))<τ, the process stops and LU(q)=∣S′(q)∣−1. Since frequent patterns at the same level share the same size, any vertex with patterns of size k>LU(q)k > LU(q)k>LU(q) can be safely pruned.

5. EXPERIMENT EVALUATION

5.1. Dataset

Our graph generator was developed to create synthetic graphs based on two parameters: the number of vertices/terms (V) and the number of relationships per vertex (R). The graph size is denoted by the pair (V, R). To map the relationships and the universal set of elements to all vertices, the relationship types are defined as *a*, *treat*, *cover*, and *found*, which are based on a real ontology called Herb Ontology. The elements, which have not been identified by ontology experts, are assumed to be characters *a*, *b*, *c*, *d*.

In the synthetic ontologies, each vertex is randomly labeled with a character and a string representing its related elements, limited to a set of 4 universal elements. These characters simulate ontology terms, while the string elements represent associated concepts. Edges between terms are randomly assigned one of several relationship types. Figure 3 illustrates a synthetic ontology (20,4) with 20 terms and degree 4. Similarly, queries are generated in the same way; For example, figure

4 shows a (3,2) query used to demonstrate the pruning and sub-ontology matching techniques discussed in the next section.

5.2. Implementation

With the data as discussed above, for (1), we set the weights of the elements as W(a) = 0.5, W(b) = 0.5, W(c) = 0.4, W(d) = 0.2, and the similarity threshold τ as 0.6. Initially, each pruning technique is utilized (section 4) on a vertex, for example, Q1 in figure 4, yielding the following results:

- **Pruning 1 (prefix filtering-based pruning)** with vertex Q1(ac): The remaining pattern after pruning is *{abc, abcd, a, d}*.
- **Pruning 2 (pattern-size-based pruning)** with vertex Q1(ac): The length upper bound Lu(Q1) is found to be 2, so the remaining pattern after pruning is *{a, d}*.
- **Pruning 3 (frequent-pattern-based pruning)** with vertex Q1(ac): The remaining pattern after pruning is *{a, d}*.

To optimize the state in the approach, we consider each state a successful instance of finding the relationships between terms. A result of subontology matching is shown in figure 5.

5.3. Effectiveness and flexibility

Effectiveness: We evaluated our pruning technique by comparing the number of matched vertices while varying the similarity threshold from 0.6 to 1. All potential matches for each query vertex were recorded. As shown in table 1, the number of matches decreases as the weight of associated elements increases, indicating that more significant elements contribute to stricter semantic similarity. For example, with weights ordered as W(a)=W(b)>W(c)>W(d), Query 5 (abcd) resulted in no matches, likely due to size constraints in size-based pruning. Furthermore, combining tree-based pruning techniques significantly reduces computational cost. Unlike traditional methods that greedily scan vertices based on label comparisons, our method avoids exhaustive matching by focusing on semantically relevant candidates. This reduces the complexity from O(m×n), where mmm and nnn are the numbers of query and data vertices, respectively, to O(k), where k is the size of the universal element set - typically much smaller than nnn. This demonstrates the effectiveness and scalability of our approach.

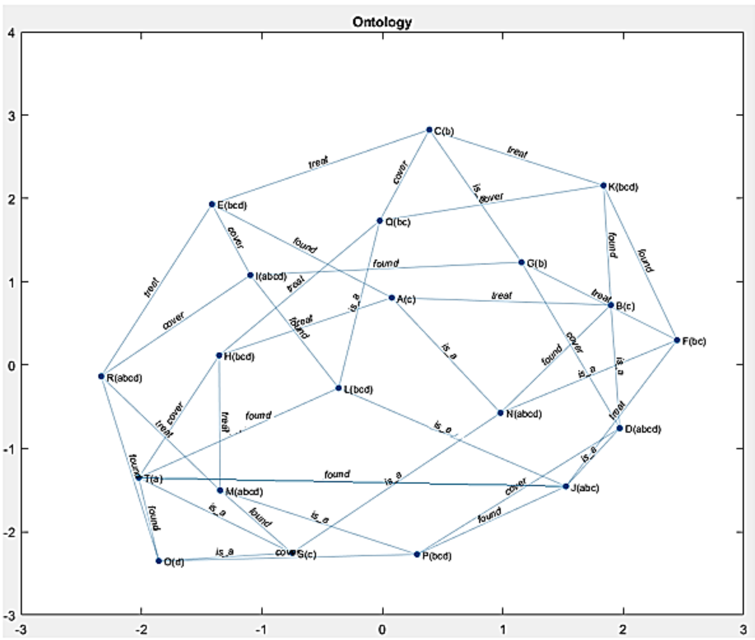

Figure 3. A synthetic ontology created by the graph generator.

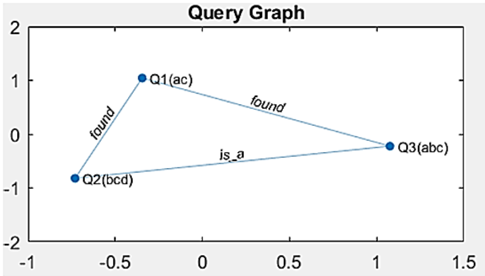

Figure 4. A synthetic query.

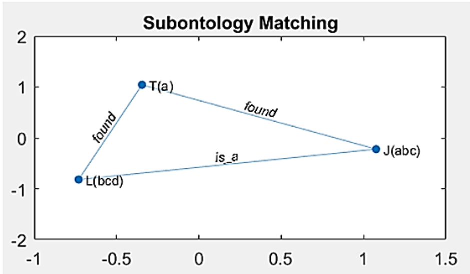

Figure 5. The matching subontology.

***Table 1.** Table to test captions and labels.*

Order	Patterns in Query	τ =0.6	τ =0.8	τ =1
1	ac	2	2	1
2	bcd	3	3	2
3	abc	2	2	2
4	ad	1	1	1
5	abcd	0	0	0

Efficiency: We evaluated the pruning techniques by measuring their ability to reduce candidate matches per query vertex at thresholds of 0.6, 0.8, and 1. Using a random five-vertex query and a fixed ontology size (50, 4), figures 6–8 show results for individual techniques (p1, p2, p3) and their combination (p). The combined method consistently yields fewer, more relevant candidates, demonstrating higher precision and filtering efficiency than any single technique.

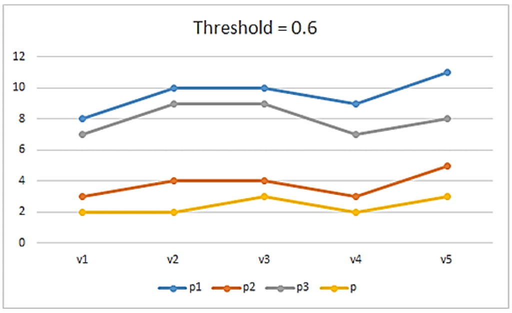

***Figure 6.** Number of candidates. Thredshold = 0.6.*

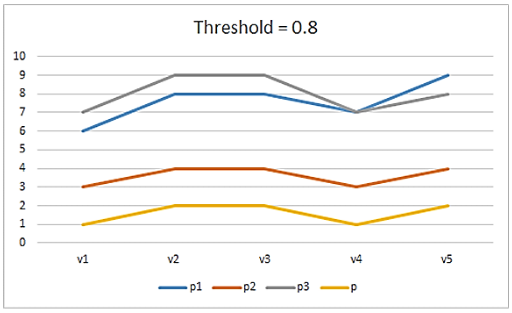

***Figure 7.** Number of candidates. Thredshold = 0.8.*

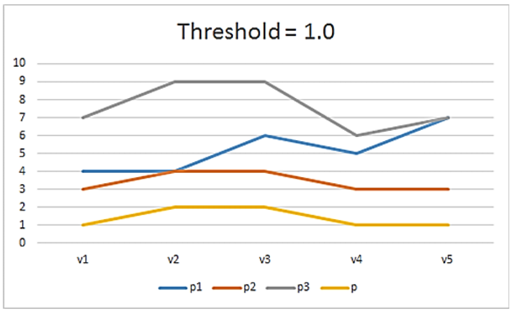

***Figure 8.** Number of candidates. Thredshold = 1.0.*

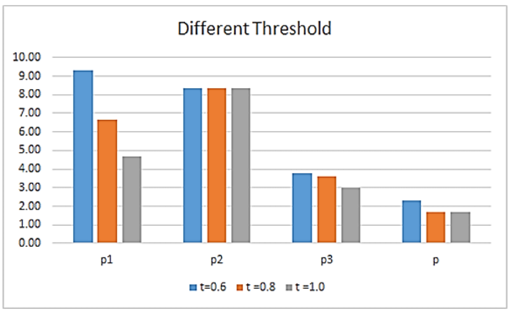

***Figure 9.** Flexibility of various thresholds.*

Flexibility: As shown in table 1, for all queries, the number of matches increases as the similarity threshold decreases. This is because a lower threshold increases the likelihood that the similarity between a query vertex and a compared ontology vertex will exceed it.

Additionally, figure 9 shows the average number of candidate patterns that match with vertices. The number of candidates increases as the similarity threshold decreases. Specifically, the figure shows that the blue column, representing the number of candidates within a 0.6 threshold, is always higher than those within thresholds of 0.8 or 1. Therefore, the flexibility lies in the ability to adjust the similarity threshold to yield more or fewer candidates, depending on specific cases.

6. CONCLUSIONS AND FUTURE WORK

This paper proposes a framework for identifying frequent client queries to reveal business trends, based on (1) ontology-based query graph matching and (2) subontology clustering. Unlike traditional graph-based methods, our approach uses set similarity and ontologies to capture semantic relationships. Pruning techniques consider both vertex properties and relationships for efficient matching. Experiments confirm the effectiveness of our pruning strategy and the suitability of K-medoids for clustering.

REFERENCES

[1]. Fudholi, Dhomas & Rahayu, Wenny & Pardede, Eric. *"Ontology-Based Information Extraction for Knowledge Enrichment and Validation"*. Pattern Recognition, 48(4):1364–1373, (2016).

[2]. Evgeny B Krissinel and Kim Henrick. *" Common subgraph isomorphism detection by backtracking search"*. Software: Practice and Experience, 34(6):591–607, (2004).

[3]. Abu-khzam, Faisal & Samatova, Nagiza & Rizk, Mohamad & Langston, Michael. *"The Maximum Common Subgraph Problem: Faster Solutions via Vertex Cover"*. IEEE/ACS International Conference on Computer Systems and Applications, pages 367–373. IEEE, (2007).

[4]. Ina Koch. *"Enumerating all connected maximal common subgraphs in two graphs"*. Theoretical Computer Science, 250(1):1–30, (2001).

[5]. John W Raymond, Eleanor J Gardiner, and Peter Willett. *"Rascal: Calculation of graph similarity using maximum common edge subgraphs"*. The Computer Journal, 45(6):631– 644, (2002).

[6]. Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. *"Exemplar queries: Give me an example of what you need"*. Proceedings of the VLDB Endowment, 7(5):365–376, (2014).

[7]. Md Saiful Islam, Chengfei Liu, and Jianxin Li. *"Efficient answering of why-not questions in similar graph matching"*. IEEE Transactions on Knowledge and Data Engineering, 27(10):2672–2686, (2015).

[8]. Ali Cakmak and Gultekin Ozsoyoglu. *"Taxonomy-superimposed graph mining"*. In Proceedings of the 11th International Conference on Extending Database Technology: Advances in Database Technology, pages 217–228. ACM, (2008).

[9]. Boanerges Aleman-Meza, Christian Halaschek-Wiener, Satya Sanket Sahoo, Amit Sheth, and I Budak Arpinar. *" Template-based semantic similarity for security applications"*. In International Conference on Intelligence and Security Informatics, pages 621–622. Springer, (2005).

[10]. O. Corby, R. Dieng-Kuntz, F. Gandon, C. Faron-Zucker. *"Searching the semantic web: Approximate query processing based on ontologies"*. IEEE Intelligent Systems, 21(1):20–27, (2006).

[11]. Eric Little, Kedar Sambhoos, and James Llinas. *"Enhancing graph matching techniques with ontologies"*. In Information Fusion, 2008 11th International Conference on, pages 1–8. IEEE, (2008).

[12]. Yinghui Wu, Shengqi Yang, and Xifeng Yan. *" Ontology-based subgraph querying"*. In Data Engineering (ICDE), 2013 IEEE 29th International Conference on, pages 697–708. IEEE, (2013).

[13]. Carlo Wouters, Tharam Dillon, Wenny Rahayu, and Elizabeth Chang. *" A practical walkthrough of the ontology derivation rules"*. In International Conference on Database and Expert Systems Applications, pages 259–268. Springer, (2002).

[14]. Carlo Wouters, Tharam Dillon, Wenny Rahayu, Elizabeth Chang, and Robert Meersman. *" Ontologies on the move"*. In International Conference on Database Systems for Advanced Applications, pages 812–823. Springer, (2004).

[15]. Andrew Peter Flahive. *"Tailoring large domain ontologies in the semantic grid environment"*. PhD thesis, La Trobe University, (2007).

[16]. Lambrix, P., Armiento, R., Li, H., Hartig, O., Abd Nikooie Pour, M., & Li, Y. *"The materials design ontology"*. Semantic Web, 15(2), 481-515, (2024).

TÓM TẮT

Kết nối đồ thị truy vấn dựa trên ontology

Yêu cầu về cạnh tranh kinh doanh đang ngày càng trở nên phức tạp hơn trong thập kỷ qua. Một trong những vấn đề này là cách phản hồi và đáp ứng các truy vấn từ khách hàng nhằm đưa ra quyết định nhanh chóng và kịp thời. Để giải quyết vấn đề này, bài báo đề xuất một framework để xác định các truy vấn thường xuyên từ nhiều nguồn khác nhau. Đóng góp chính của công trình này là (1) kết nối truy vấn dựa trên ontology (bản thể học). Thực tế, kết nối truy vấn đã được nghiên cứu rộng rãi bằng nhiều phương pháp khác nhau, hầu hết đều dựa trên cách tiếp cận gán nhãn và sử dụng đẳng cấu để ánh xạ một truy vấn đến các đồ thị con trong cơ sở dữ liệu đồ thị. Tuy nhiên, các phương pháp truyền thống này có thể không nắm bắt đủ sự tương đồng về ngữ nghĩa khi ánh xạ các thuật ngữ với nhau. Kết nối dựa trên ontology có thể giải quyết vấn đề này hiệu quả hơn so với chỉ sử dụng đẳng cấu gán nhãn. Trong nghiên cứu này, ba kỹ thuật cắt tỉa và sự kết hợp của chúng trong các tập yếu tố liên kết với đỉnh có trọng số được đề xuất. Ngoài ra, (2) một kỹ thuật phân cụm đồ thị sub-ontology (ontology con) được đề xuất để trích xuất kết quả của framework. Kỹ thuật này dựa trên một phương pháp phân cụm hiện có nhưng cần cải tiến trong bước xử lý dữ liệu để có thể áp dụng vào việc phân cụm các đối tượng ontology con/đồ thị con. Cuối cùng, (3) chúng tôi thực nghiệm để xác minh hiệu quả và tính hiệu suất của các kỹ thuật cắt tỉa bằng cách sử dụng dữ liệu tổng hợp và so sánh các kỹ thuật với nhau.

Từ khóa: Ontology; Kết nối truy vấn; Tương đồng ngữ nghĩa; Kỹ thuật cắt tỉa.