Diversified and Verbalized Result Summarization for Semantic Association Search

Yu Gu, Yue Liang, Gong Cheng, Daxin Liu, Ruidi Wei, and Yuzhong Qu

National Key Laboratory for Novel Software Technology, Nanjing University, China
{ygu,yliang,dxliu}@smail.nju.edu.cn,{gcheng,yzqu}@nju.edu.cn,
ruidiwei@gmail.com

Abstract. The widespread adoption of graph data models has enhanced many Web applications. Among others, semantic association search is to search an entity-relation graph for subgraphs called semantic associations that connect a set of entities specified in a user's query; it has proved useful in various domains. Recent research on this topic has concentrated on summarizing numerous search results by mining their important patterns to form an abstractive overview. However, those top-ranked patterns may have redundancy, and their graph structure may not be comprehensible to non-expert users. To reduce redundancy, we present a novel framework featuring a combinatorial optimization model to select top-k diversified patterns. In particular, we devise a new similarity measure which jointly considers structural and semantic similarity to assess the overlap between patterns. To facilitate non-expert users' comprehension of a pattern, we verbalize its graph structure, transforming it into compact and coherent English text based on a novel method for discourse planning. Extensive experiments demonstrate the effectiveness of our approach compared with existing methods.

Keywords: Semantic association search · Summary generation · Diversity · Graph edit distance · Graph verbalization.

1 Introduction

A common type of information needs enabled by the Semantic Web is to search an entity-relation graph (e.g., Fig. 1) for *semantic associations* (SAs) which represent complex relationships between a set of entities specified in a user's query (called query entities). Its application exists in various domains such as national security [20] and bioinformatics [16], e.g., to find notable SAs that connect a number of suspect airline passengers. Typically, a SA is defined to be a path connecting two query entities [2, 8, 14, 18], or more generally, is a compact subgraph connecting a set of query entities [5–7, 15, 22]. For example, given a query comprising the three people in Fig. 1, two SAs are shown in Fig. 2.

On a large entity-relation graph, numerous SAs can be found and overload users, which can be alleviated by properly summarizing the search results. One established approach is to aggregate SAs using their high-level abstraction called

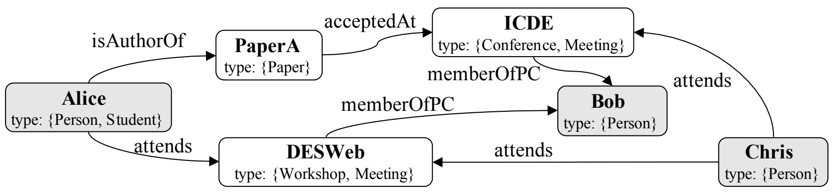


Fig. 1: A running example of entity-relation graph, including the types of each entity and a set of three query entities: `Alice`, `Bob`, and `Chris`.

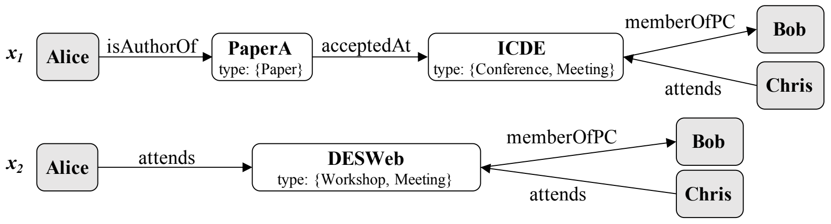


Fig. 2: Two SAs connecting `Alice`, `Bob`, and `Chris` in Fig. 1.

SA pattern (SAP), which is obtained by substituting each non-query entity in a SA with one of its types [6, 8, 18, 24]. For example, the SA x_1 in Fig. 2 matches two SAPs z_1 and z_2 in Fig. 3. Nonetheless, *two problems remain and influence the usability of SAP*. First, in a schema-rich entity-relation graph like DBpedia for the encyclopedic domain, the number of aggregated SAPs can still be excessive. Previous research ranks SAPs by their frequency [6] or query relevance [24], but simply choosing top-ranked SAPs may suffer from redundancy; e.g., z_1 and z_2 in Fig. 3 are similar. Second, as users may not have expertise in reading or accessing (Semantic Web) graphs, rather than intuitively using node-link diagram, a more comprehensible way of presenting SAP is needed for average users.

To meet the above two challenges, our research contributions are threefold.

- We devise a new similarity measure for assessing the overlap between two SAPs based on a variant of graph edit distance, which jointly considers the graph structure and the semantics of graph labels (Section 3.2).
- To select a set of top-ranked SAPs that are also diversified (i.e., with limited overlap), we model a combinatorial optimization problem and solve it using a greedy algorithm (Section 3.1).
- To make selected SAPs more comprehensible to non-expert users, we present a verbalization approach that transforms a SAP into compact and coherent English text based on a novel method for discourse planning (Section 4).

We conduct extensive experiments including a user study to evaluate our proposed contributions (Section 5).

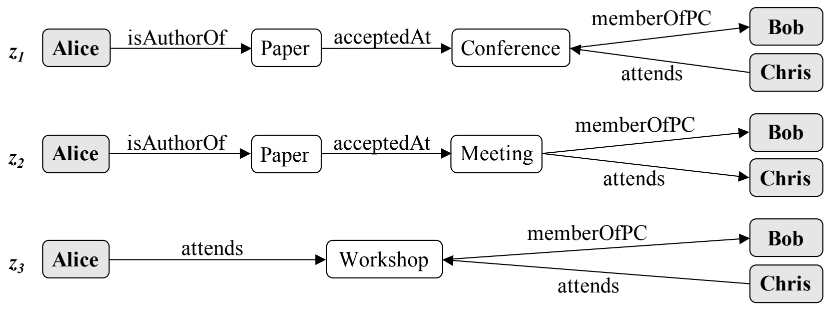


Fig. 3: Three SAPs.

2 Preliminaries

We deal with a finite directed labeled *entity-relation graph* denoted by G, where each vertex is labeled with a unique entity, and each arc is labeled with a binary relation on entities. An entity has at least one type (i.e., class). Classes are organized into a subsumption hierarchy. For example, the entity-relation graph shown in Fig. 1 will be used as a running example in this paper. However, real graphs can be much larger and more heterogeneous.

A *query* Q consists of n *query entities* ($n > 1$). All the other entities in G are called *non-query entities*. A result of Q is called a *semantic association* (SA) denoted by x, which is a subgraph of G satisfying [6, 7]: (i) x is connected, (ii) x contains all the query entities, and (iii) x is minimal, i.e., none of its proper subgraphs satisfy (i) and (ii). The minimality of x indicates that its underlying (undirected) graph is a tree. For example, Fig. 2 illustrates two SAs connecting `Alice`, `Bob`, and `Chris`. The *diameter* of x, denoted by $diam(x)$, is the largest distance between pairs of entities in x, e.g., $diam(x_1) = 3$ in Fig. 2.

A *SA pattern* (SAP) is a directed labeled graph where each vertex is labeled with a query entity or a class, and each arc is labeled with a relation [6, 8, 18, 24]. A SA x *matches* a SAP z if there is a bijection between their vertices such that (i) each query entity in x corresponds to itself in z, (ii) each non-query entity in x corresponds to one of its types in z, and (iii) the bijection is arc-preserving and label-preserving. For example, x_1 in Fig. 2 matches z_1 and z_2 in Fig. 3; x_2 matches z_3 as well as another SAP not shown in the figure.

3 Diversified SAP Selection

To form a compact summary of SAs, we can only select a handful of SAPs from possibly many candidates. Rather than simply choosing top-ranked SAPs [6, 24], we present a novel framework featuring a combinatorial optimization model to select top-k SAPs that are diversified, which uses a new similarity measure.

3.1 Framework

We aim to select k top-ranked SAPs that are not similar to each other. For ranking, we follow [6] to consider frequency, which can be replaced by any other ranking criterion. Specifically, for a set of SAs X, the *frequency* of a SAP z, denoted by $f(z, X)$, is the number of SAs in X that match z. Given a threshold $\tau > 0$, z is *frequent* if $f(z, X) \geq \tau$. Let $Z = \{z_1, \ldots, z_m\}$ be the set of all frequent SAPs for X; m is the size of Z. The k SAPs will be selected from Z.

Given $z_i, z_j \in Z$, let $sim(z_i, z_j) \in [0, 1]$ be the similarity between z_i and z_j, which will be detailed in Section 3.2; z_i and z_j are *similar* if $sim(z_i, z_j) \geq \phi$ where ϕ is a predefined threshold. For $i = 1$ to m, we use $y_i \in \{0, 1\}$ to represent whether $z_i \in Z$ is selected ($y_i = 1$) or not ($y_i = 0$), and then formulate a combinatorial optimization problem as follows:

$$\begin{aligned} \text{maximize } & \sum_{i=1}^{m} y_i \cdot f(z_i, X) \\ \text{subject to } & \sum_{i=1}^{m} y_i \leq k \\ & y_i + y_j \leq 1 \text{ for every } i \neq j \text{ and } sim(z_i, z_j) \geq \phi\,. \end{aligned} \tag{1}$$

Not more than k SAPs will be selected. The goal is to maximize the frequency of selected SAPs, but any two similar SAPs cannot be selected together.

The formulated problem is an instance of the *multidimensional 0-1 knapsack problem* (MKP), which is NP-hard and is usually solved by using a greedy algorithm [12]. The idea is to iteratively select a SAP that satisfies all the constraints and maximizes the following heuristic function:

$$\text{heuristic score of } z_i = \frac{f(z_i, X)}{|\{z_j \in (Z \setminus \{z_i\}) : sim(z_i, z_j) \geq \phi\}| + 1}\,, \tag{2}$$

until k is reached or Z is exhausted. Here, priority is given to SAPs that are more important (i.e., more frequent) and are similar to fewer other SAPs. The algorithm calculates sim for $O(m^2)$ pairs of SAPs, sorts the SAPs in Z in $O(m \log m)$ time, and selects k SAPs in $O(mk)$ time.

3.2 Similarity Measurement

We devise a new measure of similarity (sim) between SAPs which *jointly considers the graph structure and the semantics of graph labels*. Specifically, we define the distance (or dissimilarity) between two SAPs as the amount of transformation that is needed to turn one of them to the other, which is related to their *graph edit distance* (GED) [19]. We propose a new variant of GED, called pGED, which better suits our application and is also computationally less expensive.

GED Given a source graph and a target graph, the idea of GED is to transform the source into the target by a sequence of edit operations including insertions, deletions, and substitutions (i.e., relabeling) of vertices and arcs [19]. Each edit operation has a cost. A sequence of edit operations comprises an edit path. The cost of an edit path is the sum of the costs of its edit operations. An edit path that transforms the source into the target is a complete edit path. The GED from the source to the target is the minimum cost of any complete edit path. GED is normally defined to be symmetric.

pGED For SAPs, we consider a variant of GED called pGED, which *disallows any edit operation that involves query entities* because they are the focus of a SAP. Therefore, only classes and relations can be inserted, deleted, or substituted. This also helps to reduce the search space for calculating GED. For example, for $Q = \{\texttt{Alice}, \texttt{Bob}, \texttt{Chris}\}$, the pGED between z_1 and z_3 in Fig. 3 is the sum of the costs of the following edit operations:

1. Relabel `Conference` with `Workshop`.
2. Delete `Paper` along with the arcs incident from/to it.
3. Insert an arc labeled with `attends` from `Alice` to `Workshop`.

In pGED, we define the cost of an edit operation to be in the range of $[0, 1]$. The cost of insertion and deletion is fixed to 1. For a substitution which relabels a class or a relation with another, we calculate its cost by one minus the similarity between the old and the new labels, which is defined as follows.

– For two classes, we use *wpath* [25] to measure their *semantic similarity* which is based on a combination of ontological distance and statistical information. We refer the reader to [25] for details.
– For two relations, we simply calculate their *discrete similarity*; i.e., the similarity between different relations is 0.

By this way of exploiting semantics in the cost of edit operation, we readily integrate the semantics and the graph structure of SAPs, which is an important motivation for our proposed similarity measure underpinned by GED.

Calculation of pGED An A*-based algorithm has been presented for calculating GED [19]. We adapt it to suit the calculation of pGED by prohibiting edit operations on query entities. The algorithm seeks a mapping between vertices of two graphs, from which the optimum complete edit path is derived. We refer the reader to [19] for details. Here, the adaptation is: instead of starting from an empty mapping, we initially map all the query entities in one SAP to their counterparts in the other. This adaptation also reduces the search space.

Similarity between SAPs Now we are ready to define $sim(z_i, z_j)$. For z_i with ν_{z_i} vertices and ϵ_{z_i} arcs, and z_j with ν_{z_j} vertices and ϵ_{z_j} arcs, let $pged(z_i, z_j)$

be the pGED between z_i and z_j defined as above. We divide it by the following trivial upper bound such that the result is in the range of $[0, 1]$:

$$\overline{pged}(z_i, z_j) = \max\{\nu_{z_i}, \nu_{z_j}\} + \epsilon_{z_i} + \epsilon_{z_j} \, . \tag{3}$$

As *pged* is often much smaller than $\overline{pged}$, the ratio of *pged* to $\overline{pged}$ clusters close to 0. To facilitate tuning the threshold ϕ in Eqs. (1) and (2), we define *sim* as

$$sim(z_i, z_j) = 1 - \sqrt{pged(z_i, z_j) \ / \ \overline{pged}(z_i, z_j)} \, . \tag{4}$$

4 SAP Verbalization

To present selected SAPs in a way that is more comprehensible than node-link diagrams [8, 14, 18], we propose a rule-based, domain-independent approach to transform SAP into English text. Basically, each arc is expressed by a subject-predicate-object clause consisting of lexicalized relations, classes, or query entities. Our novel method for discourse planning properly aggregates and orders clauses to improve the compactness and coherence of the generated text.

4.1 Discourse Planning

Discourse planning usually comprises two levels: *sentence level* which organizes the clauses within a sentence, and *document level* which optimizes the relationship between sentences. Recall that the underlying (undirected) graph of a SAP is a tree. In our proposed method, at the document level, when the tree is large, we decompose it into small subtrees having a diameter of at most 2 ($diam \leq 2$). At the sentence level, from each small subtree we generate a simple, a compound, or a complex sentence.

Sentence-level Planning We generate a sentence from each small (sub-)tree. We assume that some vertex v_{subj} in the tree has been specified as the subject of the sentence by document-level planning which we will introduce later; document-level planning may also require another vertex v_{end} to appear at the end of the sentence. As the tree is small ($diam \leq 2$), we exhaustively plan for all possible cases in the following. *Our goal is to generate a compact sentence.*

For $diam = 1$, a tree consists of exactly one arc, as illustrated in Fig. 4(a). We *generate a simple sentence* consisting of one clause whose subject is v_{subj}.

For $diam = 2$, a tree consists of an internal vertex v_i and at least two leaf vertices v_l, v_{ll}, etc., as illustrated in Fig. 4(b). We distinguish between two cases, depending on the position of v_{subj}.

- When v_{subj} is the internal vertex v_i, we *generate a compound sentence* consisting of clauses joined by commas and the conjunction *and*. These clauses have a common subject v_i, so the subject is omitted from the second clause. Further, we order these clauses such that (i) those having a common predicate are compactly aggregated into one clause, and (ii) the last clause will be the one whose object is v_{end} if it is specified by document-level planning. For example, for z_3 in Fig. 3, if v_{subj} = `Workshop` and v_{end} = `Chris`, we will generate the following unlexicalized sentence after aggregation and ordering: ⟨`Workshop`⟩ ⟨`memberOfPC`⟩ ⟨`Bob`⟩, *and* ⟨`attends`⟩$^-$ ⟨`Alice`⟩ *and* ⟨`Chris`⟩. We use $^-$ to represent a relation to be reversely lexicalized.

– When v_{subj} is a leaf vertex, say v_l WOLOG, we *generate a complex sentence* where the main clause expresses the arc between v_l and the internal vertex v_i, and the other arcs form relative clauses with their antecedents referring to v_i. These relative clauses are aggregated and ordered also in the above-mentioned manner. For example, for z_3 in Fig. 3, if v_{subj} = `Alice` and v_{end} = `Chris`, we will generate the following unlexicalized sentence: ⟨`Alice`⟩ ⟨`attends`⟩ ⟨`Workshop`⟩, *which* ⟨`memberOfPC`⟩ ⟨`Bob`⟩ *and* ⟨`attends`⟩$^-$ ⟨`Chris`⟩.

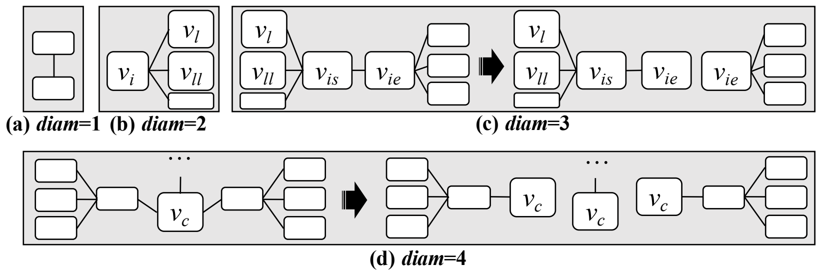


Fig. 4: An illustration of discourse planning.

Document-level Planning To verbalize a SAP, our method allows an arbitrary query entity (denoted by v_Q) to be the subject (i.e., v_{subj}) of the first sentence; in our implementation, v_Q is set to the first query entity. When $diam \leq 2$, we directly generate a sentence using sentence-level planning with $v_{subj} = v_Q$; document-level planning is not needed. When $diam > 2$, we decompose the tree into small subtrees of $diam \leq 2$, and generate one sentence for each subtree; then *the goal of our document-level planning is to improve the coherence of these sentences* by properly specifying v_{subj} and v_{end} for their sentence-level planning. Here we exhaustively plan for two cases: $diam = 3$ and $diam = 4$, as in practice larger ones still cannot be efficiently generated by existing algorithms [6]. However, our method has the potential to be generalized.

For $diam = 3$, a tree consists of two internal vertices v_{is} and v_{ie}, and at least two leaf vertices v_l, v_{ll}, etc., as illustrated in Fig. 4(c). There are two cases.

– When v_Q is a leaf vertex, say v_l WOLOG, we decompose the tree into two subtrees as shown on the right side of Fig. 4(c). The first sentence is generated using sentence-level planning with $v_{subj} = v_l$ and $v_{end} = v_{ie}$. The second sentence is generated under $v_{subj} = v_{ie}$, so *its subject verbatim repeats the end* *of the first sentence to improve coherence* [4]. For example, for z_1 in Fig. 3, if $v_Q = \texttt{Alice}$, we will generate the following unlexicalized document: $\langle\texttt{Alice}\rangle$ $\langle\texttt{isAuthorOf}\rangle$ $\langle\texttt{Paper}\rangle$, *which* $\langle\texttt{acceptedAt}\rangle$ $\langle\texttt{Conference}\rangle$. $\langle\texttt{Conference}\rangle$ $\langle\texttt{memberOfPC}\rangle$ $\langle\texttt{Bob}\rangle$ *and* $\langle\texttt{attends}\rangle^-$ $\langle\texttt{Chris}\rangle$.

– When v_Q is an internal vertex, say v_{is} WOLOG, we decompose the tree in the same way as above and generate two sentences. The only difference is that the first sentence is generated under $v_{subj} = v_Q = v_{is}$.

For $diam = 4$, a tree contains a unique central vertex v_c that is not more than two hops away from other vertices. As illustrated in Fig. 4(d), we decompose the tree at v_c into at least two subtrees. We distinguish between two classes.

– When $v_Q \neq v_c$, the first sentence is generated from the subtree containing v_Q under $v_{subj} = v_Q$ and $v_{end} = v_c$. After that, one sentence is generated from each remaining subtree under $v_{subj} = v_c$, so *the subjects of these sentences verbatim repeat the end of the first sentence to improve coherence.*
– When $v_Q = v_c$, we generate one sentence from each subtree under $v_{subj} = v_c$. *These sentences form coherent text as they have a common subject.*

4.2 Lexicalization

Lexicalization transforms $\langle\cdots\rangle$ and $\langle\cdots\rangle^-$ into natural language, where $\langle\cdots\rangle^-$ represents a relation to be reversely lexicalized. Our rules of lexicalization build on [17, 21], without using any domain-specific lexicon.

To lexicalize a relation r, we categorize it by its part of speech.

– When r is a noun phrase (e.g., *member of PC*), we prefix it by *has* (e.g., *has member of PC*). To reversely lexicalize it, we enclose it with *is...of* (e.g., *is member of PC of*).
– When r is a noun phrase enclosed with *is...of* (e.g., *is author of*), we express it as it is. To reversely lexicalize it, we express by replacing *is...of* with *has* (e.g., *has author*).
– When r is the past participle form of a verb phrase (e.g., *accepted at*), we prefix it by *is* (e.g., *is accepted at*). To reversely lexicalize it, we express by its third person singular present form (e.g., *accepts*).
– When r is any form of a verb phrase other than past participle (e.g., *attends*), we express it as it is. To reversely lexicalize it, we express by its past participle form (e.g., *is attended by*).

This categorization is general enough to cover most relations in practice.

To lexicalize a class, we use indefinite article (i.e., *a* or *an*) if it is mentioned for the first time, otherwise use definite article (i.e., *the*).

To lexicalize a query entity, we express it as it is.

For example, z_1 in Fig. 3 is finally verbalized as follows when $v_Q = \texttt{Alice}$: *Alice is author of a Paper, which is accepted at a Conference. The Conference has member of PC Bob and is attended by Chris.*

5 Experiments

Our approach has an open-source implementation[1]. We conducted a blind user study involving twenty students majoring in computer science, to test the following three hypotheses that evaluate our technical contributions.

1. Our proposed top-ranked diversified SAPs form a better summary for SAs than top-ranked undiversified SAPs considered in [6].
2. In particular, our proposed pGED is more effective than a popular similarity measure considered in previous research [9, 10, 18].
3. Our proposed discourse planning generates better document structure than the planning considered in a state-of-the-art system [1].

5.1 Datasets and Queries

Following previous research [6–8, 14, 18], our experiments were performed on DBpedia[2]. We used DBpedia's *Mappingbased Objects* for entity-relation graph, *Instance Types* and *Instance Types Transitive* for entity types, and the *DBpedia ontology* for measuring semantic similarity according to [25].

We followed [7] to construct queries. We used DBpedia entities mentioned in the training set of QALD-5 [23] as seeds. For each *seed entity* (e.g., `Anthony Kiedis`), we submitted its name to Google, and tried to collect four *related entities* that (i) "people also search for" and (ii) existed in DBpedia (e.g., `Flea`). Consequently, we could construct a query consisting of n entities by randomly choosing a seed entity and $n-1$ of its related entities.

We implemented the algorithm presented in [6] to search for SAs and mine frequent SAPs. The algorithm required configuring a diameter constraint λ and a frequency threshold τ. Given a query, it would search the entity-relation graph for all the SAs of $diam \leq \lambda$, and mine their frequent SAPs satisfying $f \geq \tau$. Under different configurations of n, λ, and τ, we constructed a total of 80 queries[3] in the above-mentioned manner, including

– 10 queries for each $n \in \{2, 3, 4, 5\}$ under $\lambda = 3$ and $\tau = 5$ such that at least 10 frequent SAPs could be mined from the search results of a query, and
– 10 queries for each $n \in \{2, 3, 4, 5\}$ under $\lambda = 4$ and $\tau = 100$ such that at least 50 frequent SAPs could be mined from the search results of a query.

Larger values of λ were not used due to the limited performance of the search algorithm [6]; τ was empirically determined.

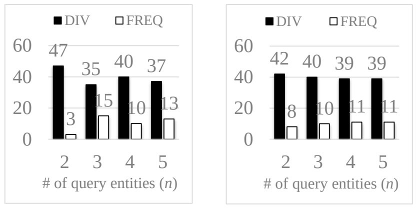
[1] `http://ws.nju.edu.cn/association/summ2018`
[2] `http://wiki.dbpedia.org/dbpedia-dataset-version-2015-10`
[3] `http://ws.nju.edu.cn/association/summ2018/query.zip`

5.2 Experiment on Diversification

Experiment Design We designed two experiments to separately test the first and the second hypotheses. In these experiments, SAPs presented to users were not verbalized but straightforwardly visualized as node-link diagrams.

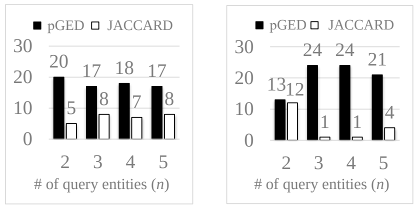
In the first experiment, we compared our top-ranked diversified SAPs (denoted by DIV) with top-ranked undiversified SAPs selected by [6] (denoted by FREQ). Both DIV and FREQ preferred more frequent SAPs, but DIV also considered diversity. In DIV, the similarity threshold was fixed to $\phi = 0.5$. For each of the 80 queries, DIV and FREQ separately selected five SAPs ($k = 5$) as two summaries of search results, to be compared by five users who were invited to decide which summary was preferred and then explain the decision.

In the second experiment, we tested the effectiveness of our similarity measure pGED, compared with a popular measure used in [9, 10, 18] which, denoted by JACCARD, measures the Jaccard similarity between sets of graph labels. The second experiment used half of the 80 queries due to the availability of users. For each query, three frequent SAPs z, z_i, z_j were selected such that pGED and JACCARD disagreed on their relative similarity; i.e., z_i and z_j were considered more similar to z by pGED and JACCARD, respectively. Five users were invited to decide whether z_i or z_j was more similar to z.



(a) $\lambda = 3$ (b) $\lambda = 4$

Fig. 5: Number of user decisions that prefer DIV and FREQ.



(a) $\lambda = 3$ (b) $\lambda = 4$

Fig. 6: Number of user decisions that agree with pGED and JACCARD.

Table 1: Average pGED between SAPs Selected by DIV and FREQ

	$\lambda = 3$				$\lambda = 4$			
	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
DIV	0.684	0.596	0.678	0.594	0.565	0.619	0.626	0.588
FREQ	0.229	0.183	0.205	0.171	0.228	0.185	0.204	0.195

Experiment Results In the first experiment, 319 user decisions (80%) preferred the SAPs selected by DIV, whereas only 81 (20%) favored FREQ. One-sample t-test showed that DIV significantly outperformed FREQ under $p < 0.01$, *suggesting that our first hypothesis could be accepted.* As detailed in Fig. 5, the superiority of DIV was observed under all the eight configurations of λ and n.

To characterize the effectiveness of DIV, as shown in Table 1, the average pGED between SAPs selected by FREQ was very small, and much smaller than the one for DIV, *showing the necessity of diversification and the effectiveness of DIV.* We also analyzed the explanations given by the users for their decisions, and found that the diversity of DIV's selection was recommended in 239 decisions (60%); i.e., *diversity was considered in most decisions.* The effectiveness of DIV was illustrated by the real example in Fig. 7, where DIV selected relatively diverse SAPs but the SAPs selected by FREQ were much redundant.

In the second experiment, 154 user decisions (77%) agreed with pGED, whereas only 46 (23%) agreed with JACCARD. One-sample t-test showed that pGED significantly outperformed JACCARD under $p < 0.01$, *suggesting that our second hypothesis could be accepted.* As detailed in Fig. 6, the superiority of pGED was observed under all the eight configurations of λ and n.

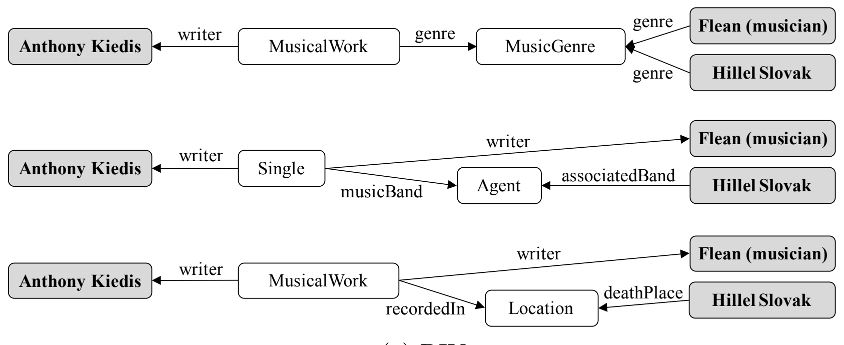
Running Time We also tested the running time of DIV on a Xeon E3-1226v3 (3.30 GHz). For each $n \in \{2, 3, 4, 5\}$, we selected a query that could result in more than 1,000 frequent SAPs under $\lambda = 4$ and $\tau = 100$. For each query, we used DIV to select SAPs under $k = 5$ from a controlled number of frequent SAPs varying from 20 to 1,000 in 20 increments.

Most time was spent calculating pGED between pairs of SAPs to establish similarity constraints, so the running time exhibited quadratic growth as shown in Fig. 8. When the number of query entities was small ($n \leq 3$) and hence SAP was small, DIV took less than 6 seconds to process 1,000 SAPs. When $n = 5$, more than 1 minute was used, which might be unacceptable for an interactive system. However, for practical use, pGEDs could be calculated in parallel.

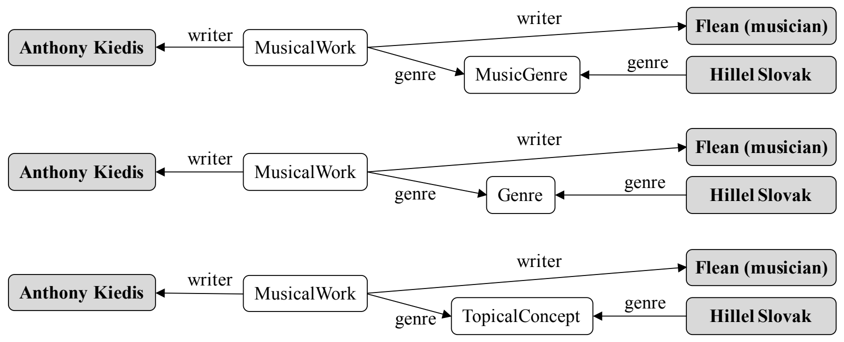
5.3 Experiment on Verbalization

Experiment Design To test the third hypothesis, we compared our verbalization approach (denoted by PaVer) with NaturalOWL [1], which was among the best-performing open-source systems that could process our SAP. We configured NaturalOWL and PaVer to start verbalization from the same query entity. NaturalOWL relied on external templates for lexicalizing relations, and we set it to use ours. Therefore, the main difference between NaturalOWL and PaVer was discourse planning, which was exactly one of our technical contributions.

For each of the 80 queries, we randomly selected a frequent SAP and used NaturalOWL and PaVer to separately generate English text to be evaluated by five users, who also had access to the original SAP visualized as a node-link diagram for reference. For each text, each user was asked to complete a questionnaire consisting of four statements about different aspects of the quality

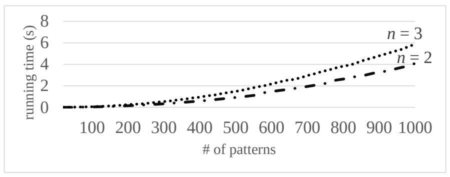


(a) DIV

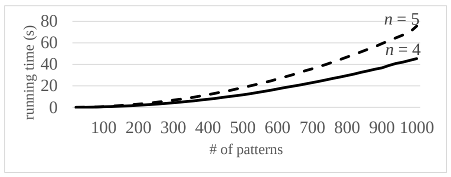


(b) FREQ

Fig. 7: Three SAPs selected by DVI and FREQ for the results of the query {`Anthony Kiedis`, `Flean`, `Hillel Slovak`}.



(a) $n = 2$ and 3.



(b) $n = 4$ and 5.

Fig. 8: Running time of DIV.

of verbalization as shown in Table 2, which extended the questionnaire used in [21]. The user's level of agreement/disagreement on each statement was to be responded on a five-level Likert item: 1 (strongly disagree), 2 (disagree), 3 (neither agree nor disagree), 4 (agree), 5 (strongly agree).

Experiment Results As shown in Table 3, the mean of all the 400 paired questionnaire results showed that PaVer outperformed NaturalOWL in all the four aspects of quality. Paired t-tests suggested that the differences were statistically significant under $p < 0.01$ in correctness, comprehensibility, and conciseness, whereas the difference in accuracy was not significant. That is, NaturalOWL and PaVer generated equally accurate descriptions for SAPs, but the text generated by PaVer was syntactically more correct, easier to comprehend, and more concise and compact, *suggesting that our third hypothesis could be accepted.*

We attributed the superiority of PaVer in correctness to its *proper use of indefinite and definite articles when lexicalizing classes*, and attributed its superiority in conciseness and comprehensibility to its *use of relative clauses in sentence-level planning* and its *coherence-oriented ordering in document-level planning*. All these features were not considered in NaturalOWL.

Besides, in Table 3, by breaking down the results by λ, comprehensibility and conciseness dropped notably on both systems when increasing λ from 3 to 4, indicating room for improvement in future work. The differences between NaturalOWL and PaVer also became larger when $\lambda = 4$, mainly because NaturalOWL generated many boring simple sentences under that setting; in fact, NaturalOWL was specifically optimized for verbalizing only a small neighborhood (one or two hops) surrounding the start entity.

Table 2: Statements in the Questionnaire about Quality of Verbalization

	Statement
(Correctness)	The text is syntactically correct.
(Comprehensibility)	The text is easy to comprehend.
(Conciseness)	The text is concise and compact.
(Accuracy)	The text precisely describes the information contained in the original graph.

6 Related Work

To avoid overloading users with numerous results in SA search, [5, 7, 15, 22] present top-ranked results which can be treated as an extractive summary. A complementary paradigm which we follow in this paper is to generate an abstractive summary, by mining and selecting a few frequent SAPs [6, 8, 18, 24]. In [6, 24], top-ranked SAPs are selected, which are ranked by their frequency [6] or by a

Table 3: Mean Response to Each Statement in the Questionnaire

Statement	Overall			$\lambda = 3$		$\lambda = 4$	
	NaturalOWL	PaVer	p-value	NaturalOWL	PaVer	NaturalOWL	PaVer
Correctness	3.79	3.95	2.0e−3	3.86	3.97	3.72	3.94
Comprehensibility	3.58	3.80	1.2e−4	3.81	3.91	3.35	3.68
Conciseness	3.36	3.68	6.2e−8	3.60	3.84	3.12	3.53
Accuracy	3.95	4.03	4.8e−2	3.96	4.08	3.95	3.98

combination of their size, graph centrality, and query relevance [24]. Considering there may be redundancy in the information provided by top-ranked SAPs, [8] proposes to select dissimilar SAPs to improve diversity, but its similarity measure can only handle path-structured SAPs. Other measures [9, 10, 18] process more general SAPs in a heuristic way. By comparison, *our pGED measure systematically considers the cost of transforming one SAP into another, which captures both the graph structure and the semantics of graph labels*, and empirically outperforms the Jaccard measure used in [9, 10, 18].

Verbalization is another contribution of our work. Compared with [1, 3, 21], whereas our lexicalization and aggregation build on common practice in this research field [3, 21], *our domain-independent discourse planning is novel and is inspired by empirical language research* [4]; in [3], the ordering and structuring of clauses are domain-dependent, and manual configuration is required. Compared with [1] which mainly generates simple and compound sentences, *our sentence-level planning also supports complex sentences for compactness, and our document-level planning properly orders sentences and clauses for coherence*, thereby showing superiority over [1] in the experiments. A related line of research verbalizes graph queries (e.g., SPARQL queries) [11, 13, 17]. Their discourse planning is task-specific and not suitable for SAP in our problem.

7 Conclusion and Future Work

Towards more usable SA search, we improve its result summarization by (i) diversifying selected SAPs based on a new GED measure which jointly considers structural and semantic similarity, and (ii) verbalizing SAPs based on a novel method for discourse planning to generate compact and coherent English text. Potential applications of our approach are not restricted to SA search. Our framework for diversification can be adapted to diversify the results of keyword queries on graph data [9, 10]. Our verbalization approach can be used to verbalize graph-structured query answers, graph representation of ontologies, etc.

Future work consists of several directions. As found in the experiments, the performance of SAP selection is not satisfying when the number of query entities or the number of SAPs is large. We will experiment with approximate algorithms and analyze trade-offs between quality and performance. As to the verbalization approach, we will extend it to support larger trees and more general graph structures. However, considering the notable fall in the comprehensibility and conciseness of text for larger SAPs in the experiment, it would be desirable to discuss the applicability of verbalization to more complex cases, and to examine other presentation techniques.

Acknowledgement

This work is supported in part by the NSFC under Grants 61572247 and 61772264, and in part by the Qing Lan and Six Talent Peaks Programs of Jiangsu Province.

References

1. Androutsopoulos, I., Lampouras, G., Galanis, D.: Generating natural language descriptions from OWL ontologies: the naturalowl system. J. Artif. Intell. Res. **48**, 671–715 (2013). https://doi.org/10.1613/jair.4017, `https://doi.org/10.1613/jair.4017`
2. Anyanwu, K., Maduko, A., Sheth, A.P.: Semrank: ranking complex relationship search results on the semantic web. In: Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005. pp. 117–127 (2005). https://doi.org/10.1145/1060745.1060766, `http://doi.acm.org/10.1145/1060745.1060766`
3. Bontcheva, K., Wilks, Y.: Automatic report generation from ontologies: The MIAKT approach. In: Natural Language Processing and Information Systems, 9th International Conference on Applications of Natural Languages to Information Systems, NLDB 2004, Salford, UK, June 23-25, 2004, Proceedings. pp. 324–335 (2004). https://doi.org/10.1007/978-3-540-27779-8_28, `https://doi.org/10.1007/978-3-540-27779-8\_28`
4. Brennan, S.E.: Centering attention in discourse. Language and Cognitive Processes **10**(2), 137–167 (1995). https://doi.org/10.1080/01690969508407091, `https://doi.org/10.1080/01690969508407091`
5. Chen, C., Wang, G., Liu, H., Xin, J., Yuan, Y.: SISP: a new framework for searching the informative subgraph based on PSO. In: Proceedings of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011, Glasgow, United Kingdom, October 24-28, 2011. pp. 453–462 (2011). https://doi.org/10.1145/2063576.2063645, `http://doi.acm.org/10.1145/2063576.2063645`
6. Cheng, G., Liu, D., Qu, Y.: Efficient algorithms for association finding and frequent association pattern mining. In: The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I. pp. 119–134 (2016). https://doi.org/10.1007/978-3-319-46523-4_8, `https://doi.org/10.1007/978-3-319-46523-4\_8`
7. Cheng, G., Shao, F., Qu, Y.: An empirical evaluation of techniques for ranking semantic associations. IEEE Trans. Knowl. Data Eng. **29**(11), 2388–2401 (2017). https://doi.org/10.1109/TKDE.2017.2735970, `https://doi.org/10.1109/TKDE.2017.2735970`
8. Cheng, G., Zhang, Y., Qu, Y.: Explass: Exploring associations between entities via top-k ontological patterns and facets. In: The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II. pp. 422–437 (2014). https://doi.org/10.1007/978-3-319-11915-1_27, `https://doi.org/10.1007/978-3-319-11915-1\_27`

9. Dass, A., Aksoy, C., Dimitriou, A., Theodoratos, D., Wu, X.: Diversifying the results of keyword queries on linked data. In: Web Information Systems Engineering - WISE 2016 - 17th International Conference, Shanghai, China, November 8-10, 2016, Proceedings, Part I. pp. 199–207 (2016). https://doi.org/10.1007/978-3-319-48740-3_14, `https://doi.org/10.1007/978-3-319-48740-3\_14`
10. Dass, A., Theodoratos, D.: Trading off popularity for diversity in the results sets of keyword queries on linked data. In: Web Engineering - 17th International Conference, ICWE 2017, Rome, Italy, June 5-8, 2017, Proceedings. pp. 151–170 (2017). https://doi.org/10.1007/978-3-319-60131-1_9, `https://doi.org/10.1007/978-3-319-60131-1\_9`
11. Ell, B., Harth, A., Simperl, E.: SPARQL query verbalization for explaining semantic search engine queries. In: The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings. pp. 426–441 (2014). https://doi.org/10.1007/978-3-319-07443-6_29, `https://doi.org/10.1007/978-3-319-07443-6\_29`
12. Fréville, A.: The multidimensional 0-1 knapsack problem: An overview. European Journal of Operational Research **155**(1), 1–21 (2004). https://doi.org/10.1016/S0377-2217(03)00274-1, `https://doi.org/10.1016/S0377-2217(03)00274-1`
13. Gardent, C., Perez-Beltrachini, L.: A statistical, grammar-based approach to microplanning. Computational Linguistics **43**(1), 1–30 (2017). https://doi.org/10.1162/COLI_a_00273, `https://doi.org/10.1162/COLI\_a\_00273`
14. Heim, P., Lohmann, S., Stegemann, T.: Interactive relationship discovery via the semantic web. In: The Semantic Web: Research and Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010, Proceedings, Part I. pp. 303–317 (2010). https://doi.org/10.1007/978-3-642-13486-9_21, `https://doi.org/10.1007/978-3-642-13486-9\_21`
15. Kasneci, G., Elbassuoni, S., Weikum, G.: MING: mining informative entity relationship subgraphs. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009. pp. 1653–1656 (2009). https://doi.org/10.1145/1645953.1646196, `http://doi.acm.org/10.1145/1645953.1646196`
16. Makita, Y., Kobayashi, N., Yoshida, Y., Doi, K., Mochizuki, Y., Nishikata, K., Matsushima, A., Takahashi, S., Ishii, M., Takatsuki, T., Bhatia, R., Khadbaatar, Z., Watabe, H., Masuya, H., Toyoda, T.: Posmed: ranking genes and bioresources based on semantic web association study. Nucleic Acids Research **41**(Webserver-Issue), 109–114 (2013). https://doi.org/10.1093/nar/gkt474, `https://doi.org/10.1093/nar/gkt474`
17. Ngomo, A.N., Bühmann, L., Unger, C., Lehmann, J., Gerber, D.: Sorry, i don't speak SPARQL: translating SPARQL queries into natural language. In: 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013. pp. 977–988 (2013). https://doi.org/10.1145/2488388.2488473, `http://doi.acm.org/10.1145/2488388.2488473`
18. Pirrò, G.: Explaining and suggesting relatedness in knowledge graphs. In: The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I. pp. 622–639 (2015). https://doi.org/10.1007/978-3-319-25007-6_36, `https://doi.org/10.1007/978-3-319-25007-6\_36`

19. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. Image Vision Comput. **27**(7), 950–959 (2009). https://doi.org/10.1016/j.imavis.2008.04.004, https://doi.org/10.1016/j.imavis.2008.04.004
20. Sheth, A.P., Aleman-Meza, B., Arpinar, I.B., Bertram, C., Warke, Y.S., Ramakrishnan, C., Halaschek, C., Anyanwu, K., Avant, D., Arpinar, F.S., Kochut, K.: Semantic association identification and knowledge discovery for national security applications. J. Database Manag. **16**(1), 33–53 (2005). https://doi.org/10.4018/jdm.2005010103, https://doi.org/10.4018/jdm.2005010103
21. Sun, X., Mellish, C.: An experiment on "free generation" from single RDF triples. In: Proceedings of the Eleventh European Workshop on Natural Language Generation, ENLG 2007, Schloss Dagstuhl, Germany, June 17-20, 2007 (2007), https://aclanthology.info/papers/W07-2316/w07-2316
22. Tong, H., Faloutsos, C.: Center-piece subgraphs: problem definition and fast solutions. In: Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006. pp. 404–413 (2006). https://doi.org/10.1145/1150402.1150448, http://doi.acm.org/10.1145/1150402.1150448
23. Unger, C., Forascu, C., López, V., Ngomo, A.N., Cabrio, E., Cimiano, P., Walter, S.: Question answering over linked data (QALD-5). In: Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015. (2015), http://ceur-ws.org/Vol-1391/173-CR.pdf
24. Yang, M., Ding, B., Chaudhuri, S., Chakrabarti, K.: Finding patterns in a knowledge base using keywords to compose table answers. PVLDB **7**(14), 1809–1820 (2014), http://www.vldb.org/pvldb/vol7/p1809-yang.pdf
25. Zhu, G., Iglesias, C.A.: Computing semantic similarity of concepts in knowledge graphs. IEEE Trans. Knowl. Data Eng. **29**(1), 72–85 (2017). https://doi.org/10.1109/TKDE.2016.2610428, https://doi.org/10.1109/TKDE.2016.2610428