Topic-based crossing-workflow fragment discovery

Zhangbing Zhou, Jinfeng Wen, Yasha Wang, Xiao Xue, Patrick C.K. Hung, Long Nguyen

Version of Record: https://www.sciencedirect.com/science/article/pii/S0167739X20301084
Manuscript_00ba10f2b4eff1c42a521a3ee7a66baf

Topic-Based Crossing-Workflow Fragment Discovery

Zhangbing Zhou[a,f,*], Jinfeng Wen[a], Yasha Wang[b], Xiao Xue[c], Patrick C. K. Hung[d], Long D. Nguyen[e]

[a]*School of Information Engineering, China University of Geosciences (Beijing), China*
[b]*National Research & Engineering Center of Software Engineering, Peking University, China*
[c]*School of Software Engineering, Tianjin University, China*
[d]*Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, Canada*
[e]*Duy Tan University, Da Nang, Vietnam*
[f]*Computer Science Department, TELECOM SudParis, Evry, France*

Abstract

Along with the large and increasing number of scientific workflows publicly accessible on Web repositories, the discovery of workflow fragments is significant to promote the reuse or repurposing of *best-practices* evidenced in legacy workflows, when novel scientific experiments are to be conducted. This paper proposes a novel crossing-workflow fragment discovery mechanism, where an activity knowledge graph is constructed to capture *flat* invocation relations between activities, and *hierarchical* parent-child relations specified upon sub-workflows and their corresponding activities. Semantic relevance of activities and sub-workflows is calculated based on their representative topics, and these topics are generated by applying the biterm topic model. Given a requirement specified in terms of a workflow fragment template, individual candidate activities or sub-workflows are discovered when considering their semantic relevance and short-document descriptions. Candidate fragments are constructed through discovering the relations in activity knowledge graph specified upon candidate activities or sub-workflows. These fragments are evaluated by balancing their structural and semantic similarities. Evaluation results show that our approach is accurate in discovering appropriate crossing-workflow fragments in comparison with the state of art's techniques.

Keywords: Crossing-Workflow Fragments; Activity Knowledge Graph; Topic Discovery; Scientific Workflows.

1. Introduction

Scientific workflows prescribe partial-ordering relations upon activities for the conduction of recurrent scientific experiments. Along with the wide-adoption of Web 2.0 technology, activities are usually encapsulated and represented in terms of Web/REST services, or mashup APIs nowadays, and scientific workflows can be publicly accessible through the Internet, in order to promote their sharing with colleagues when possible. For instance, there are more than 2,830 scientific workflows contained in the *myExperiment* repository [1], and these workflows are provided by workflow systems including *Taverna 2*, *Taverna 1*, *RapidMiner*, *Galaxy*, etc. Considering the fact that developing a novel scientific workflow from scratch is typically a knowledge- and effort-intensive, and error-prone mission, reusing and repurposing *best-practices* that have been evidenced by legacy scientific workflows is considered as a cost-effective and error-avoiding strategy [2, 3].

Current techniques have been developed to support the reuse and repurposing of workflows [4]. Generally, the similarity of workflows (or business processes) can be effectively calculated to evaluate whether a workflow can be replaced (or reused) by another when necessary. These techniques can be categorized into annotation-based [5], structure-based [6], and data-driven [7, 8] mechanisms. We have proposed a layer-hierarchial technique for computing the similarity of scientific workflows in the *myExperiment* repository, where a layer hierarchy represents the relations between a workflow, its sub-workflow and their activities [9]. In general, these techniques aim to calculate the similarity for pairs of workflows, and discover and rank candidate workflows concerning certain requirements according to their similarity values, in order to promote the reuse or repurposing of single workflows. Note that when the similarity is no larger in value than a pre-specified threshold, a conclusion can be safely drawn that no single workflow could be recommended for satisfying certain requirements.

It is worth emphasizing that a novel requirement may be partially relevant to multiple scientific workflows. This is due to the fact that certain experimental scenarios may *not* or *partially* occur, and thus, no single workflow exists in the repository that can satisfy respective requirements. For instance, Fig. 4 shows a sample requirement that should be satisfied through the composition of crossing-workflow fragments derived from legacy scientific work-

* *Email address:* zbzhou@cugb.edu.cn (Zhangbing Zhou)
URL: wenjinfeng@cugb.edu.cn (Jinfeng Wen), wangyasha@pku.edu.cn (Yasha Wang), jzxuexiao@tju.edu.cn (Xiao Xue), patrick.hung@uoit.ca (Patrick C. K. Hung), dinhlonghcmut@gmail.com (Long D. Nguyen)

flows, rather than by any single workflow. In this setting, this requirement should be achieved by discovering appropriate fragments contained in various workflows, and assembling these crossing-workflow fragments according to certain principles. To mitigate this problem, typical fragments can be discovered from single workflows [10, 11], and a fragment-index mechanism has been developed to promote the search and reuse of these fragments with various granularities [12]. Besides, we have proposed a crossing-workflow fragments discovery mechanism [13], where an activity network is constructed to capture invocation relations between pairs of activities in scientific workflows. A graph isomorphism algorithm is applied to discover activity fragments in this activity network with respect to the workflow model representing certain requirements. A fragment is examined as appropriate when activities in this fragment and the workflow model are pair-wisely matching. Generally, current techniques can discover fragments whose activities may be contained in various workflows. The constraint lies in the fact that except for *flat* invocation relations, other relations like *hierarchical* parent-child relations may exist between workflows, their sub-workflows, and activities, whereas these relations have not been included in the activity network. However, a certain requirement may be satisfied by composing fragments with coarse or fine granularities with respect to the granular level of requirement specification, and this happens especially when scientists are initially not familiar with legacy workflows in the repository, and thus, has to refine the solution through discovering coarse to fine fragments in an iterative fashion. Consequently, the discovery of crossing-workflow fragments containing activities with coarse or fine granularities is challenging.

To remedy this issue, this paper proposes a novel crossing-workflow fragments discovery mechanism through the query answering based on knowledge graphs [14]. The contribution of this paper are summarized as follows:

- We construct an activity knowledge graph, whose edges capture (i) *flat* invocation relations between activities in scientific workflows in the *myExperiment* repository, and (ii) *hierarchical* parent-child relations specified upon sub-workflows and their corresponding activities.
- We calculate the semantic relevance of activities and sub-workflows leveraging their *representative* topics, where these topics are generated by applying the biterm topic model based on the description in short-texts for activities and sub-workflows.
- We design a novel and efficient crossing-workflow fragments discovery algorithm when the requirement is specified in terms of a workflow fragment.
- We conduct extensive experiments, and evaluation results demonstrate the advantage of our approach in comparison with the state of art's techniques.

The rest of this paper is organized as follows. Section 2 presents the technique to construct an activity-based knowledge graph. Section 3 explores the topic relevance of activities and sub-workflows. Section 4 discovers and recommends fragments that are crossing scientific workflows. Section 5 introduces our experimental settings, and Section 6 evaluates this approach and compares it with the state of art's techniques. Section 7 reviews relevant techniques, and Section 8 concludes this work.

2. Activity Knowledge Graph Construction

As presented in our previous work [9], a scientific workflow represents invocation and parent-child relations upon sub-workflows and activities. Specifically,

Definition 1 (*Scientific Workflow*). *A scientific workflow swf is a tuple $(tl, dsc, SWF_{sub}, ACT, LNK)$, where:*

- *tl is the title of swf.*
- *dsc is the description in the short-text of swf.*
- *SWF_{sub} is a set of sub-workflows contained in swf.*
- *ACT is a set of activities contained in swf.*
- *$LNK = \{LNK_{inv}, LNK_{pch}\}$ is a set of links, where LNK_{inv} specifies* flat *invocation relations upon SWF_{sub} and ACT, and LNK_{pch} specifies* hierarchical *parent-child relations upon sub-workflows in SWF_{sub} and their corresponding activities in ACT.*

Note that a sub-workflow can be regarded as an activity with relatively *coarse* granularity. The invocation relations between activities prescribe the execution sequences upon functional services. An activity in the *myExperiment* repository can be REST/Web service or a mashup API, and it is usually represented by (i) a *name* in string with several keywords, and (ii) a *description* in short-text. As presented in our previous work [9], a scientific workflow can be converted into a layer hierarchy, when hierarchical parent-child relations are explicitly specified for activities between contiguous layers. Through folding sub-workflows into single activities, each layer represents (part of) the workflow specification. A sample scientific workflow and its corresponding layer hierarchy are shown in Fig. 1 and 2, respectively.

To facilitate the discovery of fragments that may contain activities from various workflows, we construct an activity knowledge graph (AKG) to capture various kinds of structural relations among workflows, sub-workflows, and activities in a semantic fashion. Given a set of scientific workflows $\{swf\}$, AKG is defined as follows:

Definition 2 (*Activity Knowledge Graph*). *An activity knowledge graph is a tuple (E, R, S), where*

- *E is a set of entities, which includes workflows, their sub-workflows, and activities in $\{swf\}$.*
- *R is a set of relation types including:*
  - Invok *to specify a flat invocation relation between a pair of sub-workflows or activities.*

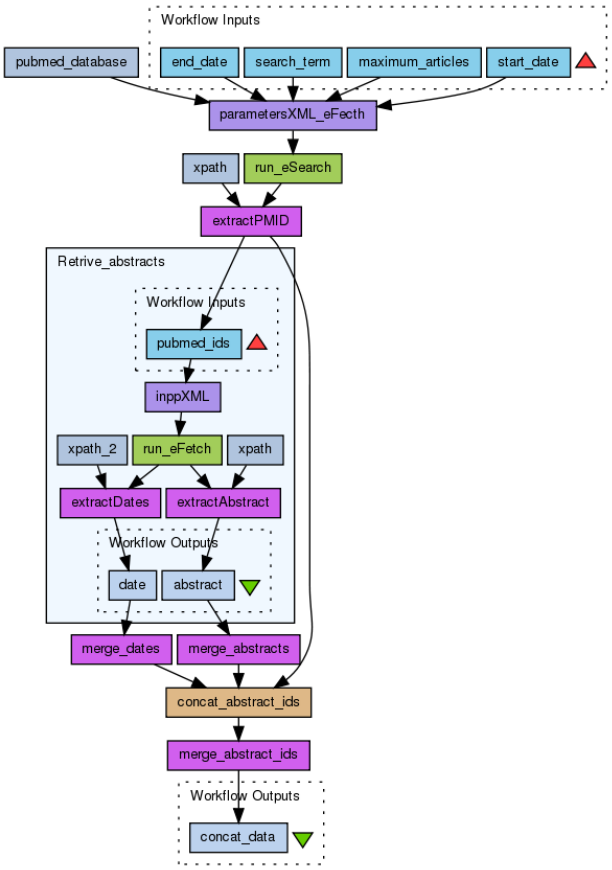

Figure 1: A scientific workflow originally from *Taverna 2* of the *myExperiment* repository with the title "*PubMed Search*" publicly accessible at `https://www.myexperiment.org/workflows/1833.html`.

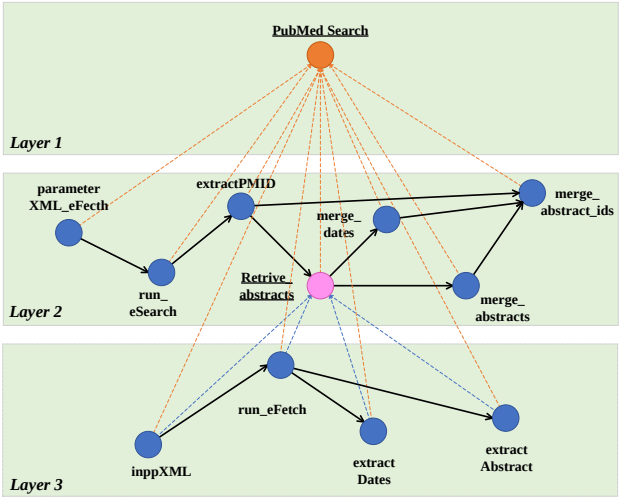

Figure 2: The layer hierarchy for scientific workflow as shown in Fig. 1.

  - PrtOf *to specify the relation for a sub-workflow or activity as part of a workflow.*
  - PrtCld *to specify a hierarchical parent-child relation for a sub-workflow and the corresponding set of activities.*

- $S \subset E \times R \times E$ *is a set of triples, which specify the relations prescribed upon entities.*

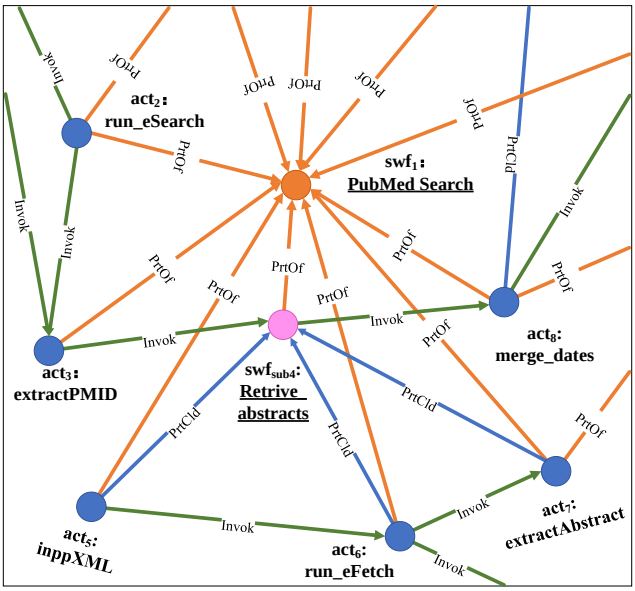

Figure 3: A snippet of *AKG* for layer hierarchy as shown in Fig. 2, where legacy scientific workflows are interconnected when they share certain activities or workflow fragments.

An *AKG* is constructed by adopting relations prescribed in the layer hierarchy of scientific workflows. A snippet of *AKG* is shown in Fig. 3, where (i) entities for workflows are denoted by nodes in orange, and "swf_1*: PubMed Search*" is an example, (ii) entities for sub-workflows are denoted by nodes in pink, and "*Retrive_abstracts*" is an example, and (iii) entities for activities are denoted in blue, and act_3 is an example. Relations are denoted as triples, and sample relations are given as follows:

- $s_1 = (swf_{sub4}, PrtOf, swf_1)$
- $s_2 = (act_5, PrtOf, swf_1)$
- $s_3 = (act_5, Invok, act_6)$
- $s_4 = (act_5, PrtCld, swf_{sub4})$

Leveraging this *AKG*, we propose in the following sections to discover crossing-workflow fragments, whose definition is presented as follows:

Definition 3 (*Crossing-workflow Fragment*). *A crossing-workflow fragment cwf is a tuple (ACT, LNK, SWF), where ACT, LNK, and SWF are the same as those in scientific workflows.*

A sample crossing-workflow fragment is shown in Fig. 4, and it is composed of (i) activities act_4, act_5, act_6, act_7, and sub-workflows swf_{sub3} originated from workflow swf_1, and (ii) activities act_5, act_6, act_7, act_8, and sub-workflows swf_{sub9} originated from workflow swf_2.

3. Topic-Based Activity Relevance

Note that activities and sub-workflows may be semantically equivalent or similar although they are different in their names and descriptions, this section explores the semantic relevance of activities and sub-workflows to facilitate the discovery of crossing-workflow fragments. Considering the fact that the name is typically the concatenation of several keywords or their abbreviations, and their description contains one or several sentences, we adopt the topic model [15], such that activities or sub-workflows are assumed as semantically relevant when they are highly correlated in their representative topics.

3.1. Activities and Sub-Workflows Representation as Short Documents

This section converts the description for activities and sub-workflows into a chain of keywords, and appends them upon the name, to generate short documents in order to derive their topics. We argue that the name, which is usually represented as the concatenation of few carefully-selected keywords or their abbreviations, should contain more precise information than the words in the relatively long text description. On the other hand, due to the briefness of the name which may hardly convey complete information, words in text descriptions are evaluated and combined as the complement of the name to generate short documents as the representation of activities or sub-workflows.

A word in text description should be considered as relevant when this word (i) is semantically similar to a word in the name (i.e., *semantic* relevance), and (ii) it co-occurs frequently with a word which is semantically equivalent or similar with a word in the name (i.e., *co-occurrence* relevance). Algorithm 1 presents the evaluation and selection procedure for words in text descriptions, where ACT is a set of activities or sub-workflows in terms of their names and text descriptions, thd_{sim} is the pre-specified threshold to determine semantic relevance, and thd_{rel} is the pre-specified threshold to determine co-occurrence relevance. Note that thd_{sim} (or thd_{rel}) should be set to an appropriate value with respect to certain requirements of domain applications.

As presented by Algorithm 1, given an activity or sub-workflow act contained in ACT with the name act_{nm} and text description act_{ds}, a word splitting function Spt is adopted to convert act_{nm} and act_{ds} to sets of words, and these words are saved in word document sets WD_{nm} and WD_{ds}, respectively (line 2). The semantic similarity between words in WD_{nm} and words in WD_{ds} is calculated pair-wisely through the calculation function $SimCal$ based on *WordNet* [16] (line 4). Generally, *WordNet* is a lexical database of the English language, and it has been widely used for supporting the semantic computation of similarity for paragraphs. If the similarity between wd_{nm} and wd_{ds} is no smaller than the pre-specified threshold thd_{sim}, wd_{ds} is assumed semantically similar to wd_{nm}, and thus, wd_{ds} in WD_{ds} can be selected and inserted into the keyword set KW (line 5), which is considered semantically similar to words in the name. The co-occurrence situation in the

Algorithm 1 Representative Words Selection

Require:
- ACT: a set of activities and sub-workflows
- thd_{sim}: the threshold for semantic relevance
- thd_{rel}: the threshold for co-occurrence relevance
- $wdSz$: the size of co-occurring text window

Ensure:
- WD: a set of representative words for ACT

1: **for all** $act \in ACT$ **do**
2: $WD_{nm} \leftarrow Spt(act_{nm})$; $WD_{ds} \leftarrow Spt(act_{ds})$
3: **for** $\forall\ wd_{nm} \in WD_{nm}$ **and** $\forall\ wd_{ds} \in WD_{ds}$ **do**
4: **if** $SimCal(wd_{nm}, wd_{ds}) \geq thd_{sim}$ **then**
5: $KW \leftarrow KW \cup \{wd_{ds}\}$
6: **end if**
7: **end for**
8: $FR \leftarrow FrCl(act_{ds})$
9: $CPN \leftarrow InPr(WD_{ds})$
10: $WI \leftarrow WdSp(act_{ds}, wdSz)$
11: **for all** $wi \in WI$ **do**
12: $CPN \leftarrow CPN \cup StaOcc(wi)$
13: **end for**
14: $CPR \leftarrow ClRl(CPN, FR)$
15: **for all** $kw \in KW$ **do**
16: **if** $QyRl(kw, CPR) \geq thd_{rel}$ **then**
17: $WD \leftarrow WD \cup \{pr_{kw}\}$
18: **end if**
19: **end for**
20: **end for**

text description is explored as follows. Word frequencies are counted and saved into the *FR*equency set FR for each word in WD_{ds} based on act_{ds} and the frequency calculation function $FrCl$ (line 8). The numbers of co-occurrence for the pairs of words are initialized to 0 and saved into CPN by the initialization function $InPr$ (line 9), where the sets of words in different pairs are usually different. Since text descriptions are mostly not short in terms of the number of words, the size of co-occurring text window $wdSz$ is set to a certain value, and thus, act_{ds} can be transferred into multiple sub-vectors wi leveraging the window splitting function $WdSp$ (line 10). We iterate each wi and update the values of co-occurrence pairs in CPN through the statistical function $StaOcc$ (lines 11-13). Consequently, the co-occurrence relevance cr for each pair is calculated based on CPN and FR as follows:

$$CPR(wd^i_{ds}, wd^j_{ds}, cr) = CPN(wd^i_{ds}, wd^j_{ds}, cn)/FR(wd^i_{ds}, fr), \quad (1)$$

where FR represents the frequency of occurrences fr of the word wd^i_{ds}, and CPN refers to the number of co-occurrences cn where words wd^i_{ds} and wd^j_{ds} appear in a certain text description. CPR shows that the larger the ratio of (i) the times the word wd^i_{ds} co-occurs with the word wd^j_{ds}, with respect to (ii) the frequency where the word wd^i_{ds} appears, the more relevant the words wd^i_{ds} and

wd^j_{ds} are. Consequently, relevant pairs pr_{kw} are discovered for each keyword kw in KW through the query function $QyRl$. When the value of co-occurrence relevance is relatively large in comparison with the pre-specified threshold thd_{rel}, words in pr_{kw} are inserted into the set WD (lines 15-19). Consequently, representative words for each activity and sub-workflow are selected and appended to enrich their names. In this setting, the representations of short documents for activities and sub-workflows are generated for the topic discovery task in the following section.

The time complexity of Algorithm 1 is $O(n_1 * s_1 * d_1)$, where n_1 is the maximum number of activities and sub-workflows in the repository, s_1 is the upper number of words in the word document set WD_{nm} for names, and d_1 is the upper bound number of words in the word document set WD_{ds} for text descriptions. Note that line 11 and line 15 should iterate fewer times than line 3, and thus the time complexity of Algorithm 1 is captured by line 1 and line 3, which iterate $n_1 * (s_1 * d_1)$ times for the procedure of lines 4-6 in the worst case. Therefore, the time complexity of this algorithm is $O(n_1 * s_1 * d_1)$.

3.2. Topic Discovery Using BTM

Based on short documents generated for activities and sub-workflows, this section aims to discover their topics, where biterm topic model (BTM) [17, 18] is assumed as the most appropriate algorithm for processing short documents. Generally, topics are represented as groups of correlated words in topic models like BTM. Topics are learned by modeling the generation of word co-occurrence patterns (i.e. *biterms*) in the short document-based corpus, where the problem of sparse word co-occurrence patterns can be addressed at the document-level. The procedure of topic discovery includes the following steps:

3.2.1. Content Digitization

Short documents are digitized and transformed into the format according to the requirement of BTM. This procedure is presented by Algorithm 2, where DOC represents the set of short documents for activities and sub-workflows. Given a document $doc_i \in DOC$, all words contained in doc_i (denoted $word(doc_i)$) are examined for the statement that whether a word $wd^j_{doc_i}$ is included in the word conversion table $WdLt$ through the query function $QyIs$. If this is not a fact, $wd^j_{doc_i}$ is inserted into $WdLt$ (lines 3-4). The mapping relation MP between the word wd_i and the serial number id_{tmp} is constructed in order to re-represent DOC (lines 9-17). Especially, the size of serial numbers is determined by the word number in $WdLt$ starting at 0 and corresponding to words in an increasing order (line 10). When doc_i is required to be transferred into the digital document doc^{wds}_i, the corresponding mapping relations are retrieved through the query function $QyId$, whose numbers are used to replace words that are located through the replacement function $RpLc$ (line 14). Consequently, doc^{wds}_i is updated and inserted into the digital corpus SN_{wd} (line 16).

Algorithm 2 Content Digitization

Require:
- DOC: a set of short documents for activities and sub-workflows

Ensure:
- SN_{wd}: a set of serial numbers that represent short documents contained in DOC

```
1: for all doc_i ∈ DOC do
2:   for all wd^j_{doc_i} ∈ word(doc_i) do
3:     if QyIs(WdLt, wd^j_{doc_i}) = false then
4:       WdLt ← WdLt ∪ {wd^j_{doc_i}}
5:     end if
6:   end for
7: end for
8: id_tmp ← 0
9: for all wd_i ∈ WdLt do
10:   MP ← MP ∪ {(wd_i, id_tmp++)}
11: end for
12: for all doc_i ∈ DOC do
13:   for all wd^j_{doc_i} ∈ word(doc_i) do
14:     doc^{wds}_i ← RpLc(doc_i, QyId(MP, wd^j_{doc_i}))
15:   end for
16:   SN_wd ← SN_wd ∪ {doc^{wds}_i}
17: end for
```

The time complexity of Algorithm 2 is $O(n_1*(s_1+d_1))$, where n_1 is the maximum number of short documents for activities and sub-workflows, and $(s_1 + d_1)$ is the upper bound number of words in short documents including its name and text descriptions of a certain activity. Generally, the iteration times of lines 1-2 and lines 12-13 are the same, and there are $n_1 * (s_1 + d_1)$ times of iteration in the worst case for the procedure of lines 3-5 or line 14. Consequently, the time complexity of this algorithm is $O(n_1 * (s_1 + d_1))$.

3.2.2. Biterm Extraction

According to the principle of BTM, every short text in short documents can be treated as a separate text fragment. Any pair of distinct words is extracted as a biterm, and these biterms are treated as the training data set of topic probability distribution in BTM. For instance, three biterms are extracted from the short document "*pathway description gene*" as "*pathway description*", "*pathway gene*", and "*description gene*".

3.2.3. BTM Training

The corpus of short documents can be regarded as a mixture of topics, where each biterm is drawn from a specific topic independently. Generally, the probability that a biterm drawn from a specific topic is further captured by the chance such that both words in the biterm are drawn from the topic. This concrete generative procedure, where α and β are supposed to be *Dirichlet* priors, can be illustrated as follows:

- ***Step 1***: For each topic z, draw a topic-specific word distribution $\phi_z \sim \text{Dir}(\beta)$;
- ***Step 2***: Draw a topic distribution $\theta \sim \text{Dir}(\alpha)$ for the whole corpus of short documents;
- ***Step 3***: For each biterm b in the set of biterms B, draw a topic assignment $z \sim \text{Multi}(\theta)$, and draw two words: w_i, $w_j \sim \text{Multi}(\phi_z)$.

The joint probability of a biterm $b = (w_i, w_j)$ according to the above procedure can be calculated as follows:

$$P(b) = \sum_z P(z)P(w_i|z)P(w_j|z) = \sum_z \theta_z \phi_{i|z}\phi_{j|z}. \quad (2)$$

Therefore, the likelihood for the corpus of short documents is calculated as follows:

$$P(B) = \prod_{(i,j)} \sum_z \theta_z \phi_{i|z}\phi_{j|z}. \quad (3)$$

Generally, latent topics are discovered when considering word co-occurrence patterns, where these patterns are exhibited in the corpus of short documents.

3.3. Inferring Topics of Activities and Sub-Workflows

Since *BTM* does not model the document generation process, topic proportions for activities and sub-workflows are not discovered directly by applying the topic learning technique. To infer topics for each short document d, the expectation of topic proportions of biterms generated from d is calculated as follows:

$$P(z|d) = \sum_b P(z|b)P(b|d). \quad (4)$$

According to the parameters estimated by *BTM*, the factor $P(z|b)$ can be obtained through the Bayes' formula as presented by Formula 5:

$$P(z|b) = \frac{P(z)P(w_i|z)P(w_j|z)}{\sum_z P(z)P(w_i|z)P(w_j|z)}, \quad (5)$$

where $P(z) = \theta_z$, $P(w_i|z) = \phi_{i|z}$ and $P(w_j|z) = \phi_{j|z}$. Note that parameters θ and ϕ are determined through adopting *Gibbs* sampling [19] for the approximate inference, which is a widely applicable *Markov chain Monte Carlo* algorithm. Generally, *Gibbs* sampling is in principal more accurate since it asymptotically approaches the correct distribution. Therefore, $P(b|d)$ is calculated by taking the empirical distribution of biterms in d as the estimation:

$$P(b|d) = \frac{n_d(b)}{\sum_b n_d(b)}, \quad (6)$$

where $n_d(b)$ refers to the frequency of the biterm b in d. In short texts, $P(b|d)$ follows a uniform distribution over all biterms in d mostly.

3.4. Determination of an Optional Topic Number K

Note that the result of the topic extraction procedure is sensitive to the pre-specified number of topics K. Therefore, determining an optimal K is fundamental leveraging certain metrics. The perplexity is a factor that is usually adopted for evaluating the quality of a language model. However, K is usually large in number and it may cause the problems of low topic identification and high similarity between extracted topics. In fact, the topic identification and topic similarity are closely related. The lower the topic similarity and the higher the topic identification, and hence, K should be determined considering these two factors. Consequently, this paper adopts the perplexity and topic similarity in order to balance the generalization ability and improve the effect of topic extraction in *BTM*.

3.4.1. Perplexity

Generally, the perplexity decreases along with the increase of the number of topics. A low value of perplexity is supposed to generate a better predictive effect on testing text corpus. In a topic model, the perplexity is calculated as follows:

$$Per = \exp\{-\frac{\sum_{d=1}^{M} \log p(w_d)}{\sum_{d=1}^{M} N_d}\}, \quad (7)$$

where M represents the number of short documents, and N_d is the number of words contained in a certain document d. The parameter w_d refers to the words in d, while $p(w_d)$ is the probability produced by w_d leveraging the document-topic and topic-word distributions. When Per approaches to 0, a better generalization ability is assumed to be achieved.

3.4.2. Topic Similarity

The topic similarity is usually calculated using methods like *Kullback-Leibler* (*KL*) divergence, or *Jensen-Shannon* (*JS*) divergence, etc. Considering the fact that *KL* divergence can hardly satisfy symmetry and triangle inequality, but *JS* divergence can, although they are similar regarding the order of topic comparison. In this paper, *JS* divergence is adopted for the measurement of topic similarity. On the basis of *JS* divergence, we introduce the variance of random variables into the potential topic space, which can measure the overall difference of the topic space. The variance for topics, denoted Var, is calculated as the average for the sum of squares of the distances between (i) the word probability distribution of each topic, and (ii) the mean probability distribution of words, as follows:

$$Var = \sum_{i=1}^{K} [D_{JS}(T_i, \bar{\xi})]^2 \div K, \quad (8)$$

where $\bar{\xi}$ is the mean probability distribution of different words obtained by the topic-word probability distribution T. K represents the number of topics, and D_{JS} denotes the *JS* divergence. When Var is larger, the difference between topics is greater, and thus, the distinction between topics is higher and the topic structure is more stable.

3.4.3. Topic Number K Determination

Based on afore-determined perplexity and topic similarity, the factor Per_Var, which is adopted to evaluate the performance of topic models, is calculated as follows:

$$Per\_Var = Per \div Var. \tag{9}$$

When Per_Var is relatively small in value, the corresponding topic model is assumed to be optimal. The range for optimal K can be determined by testing several K values according to experiments. As an example of scientific workflows in this paper, the values for Per, Var, and Per_Var are calculated and presented at Table 1. An optimal K as 370 is determined when Per_Var is the smallest in value as 35.801, while Per is the smallest as 11.288, and Var is the largest as 0.315.

3.5. Representative Topics Determination

After generating the topic distribution of short documents, we intend to use *representative* topics to quantify the semantic relevance of activities or sub-workflows to facilitate their reuse and repurposing. In most scenarios, a small number of topics may have a large probability score for each activity or sub-workflow, and these topics are assumed as *representative* in this paper to represent the latent semantic information for activities or sub-workflows.

The procedure of determining representative topics is presented as follows. For a certain topic, an average probability is calculated with respect to all activities and sub-workflows. A threshold thd_{tp}, which is usually pre-set to several times of this probability average, is adopted to specify the significance of this topic. When the value is larger than thd_{tp}, this topic is assumed as significant. Consequently, the probability of this topic is reserved when the value is no smaller than thd_{tp}, and this topic is assumed as *representative* for a certain activity or sub-workflow. Since the probability for non-representative topics is set to zero, a probability normalization procedure is conducted for the topics of activities and sub-workflows afterwards.

For instance, given an activity with the name "*Kegg_gene_ids*" in the workflow entitled "*Pathways and Gene annotations for QTL region*", the short document is generated as "*kyoto genome encyclopedia gene*". Sample topics are generated as ($topic_{332}$, 0.6117) and ($topic_{73}$, 1.3951E-9), where the topic $topic_{332}$ contains the top 10 keywords including "{*gene, encyclopedia, kyoto, genome, provide, database, description, get, rest, national*}". The average probability is calculated as 0.0018 for $topic_{332}$, and the threshold is set to five times of this average as 0.0090. Consequently, $topic_{332}$ is reserved as a representative topic. On the other hand, the average probability for $topic_{73}$ is calculated as 0.0026, and the threshold is set to 0.0130 accordingly. Therefore, $topic_{73}$ is not a representative topic, and its probability is set to 0 afterwards.

4. Crossing-Workflow Fragments Discovery

This section introduces our crossing-workflow fragments discovery mechanism leveraging the semantic relevance of representative topics for activities and sub-workflows. Specifically, candidate activities or sub-workflows are discovered according to the specification of activity stubs in a certain requirement represented in terms of a workflow fragment. Crossing-workflow fragments are discovered through the query processing upon *AKG* constructed in Section 2, and these fragments are evaluated and recommended when balancing their structural and semantic similarities with respect to the requirement.

4.1. Candidate Activity and Sub-workflow Discovery

A novel scientific workflow fragment to be constructed with respect to an un-conducted scientific experiment may be similar to an existing one. In this setting, a certain scientific workflow should be reused or repurposed partially or completely. On the other hand, when a requirement is relevant with several scientific workflows in the repository, activities or sub-workflows should be discovered from appropriate workflows and thus optimally composed accordingly. For instance, a sample crossing-workflow fragment is presented in Fig. 4 for satisfying a certain requirement.

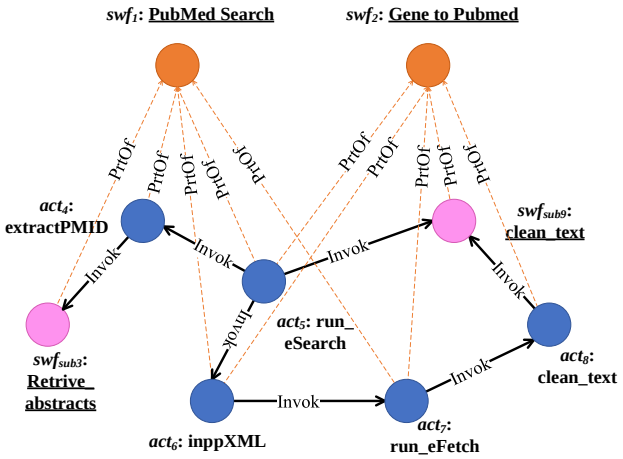

Figure 4: Based on the *AKG* snippet as shown in Fig. 3, a sample requirement is generated that can be satisfied by the composition of crossing-workflow fragments, where "swf_1: PubMed Search", "swf_2: Gene to Pubmed", "swf_{sub3}: Retrive_abstracts", and "swf_{sub9}: clean_text" are legacy scientific workflows in the repository.

Candidate activities or sub-workflows are discovered when they are highly relevant in their representative topics respecting those of activity stubs in the requirement. It is worth emphasizing that topics can be regarded as a high-level categorization. When two activities (or sub-workflows) are highly relevant in their topics, this fact can hardly specify that they are equivalent or highly similar to each other. Taking this fact into concern, the semantic similarity of short documents is adopted to improve the accuracy of the discovery for activities or sub-workflows.

Table 1: Per, Var and Per_Var Settings When K is Set to Various Values.

K	80	120	160	260	350	360	**370**	380	400
Per	23.840	19.179	16.809	13.145	11.765	11.384	**11.288**	11.555	11.776
Var	0.264	0.289	0.296	0.311	0.314	0.313	**0.315**	0.312	0.310
Per_Var	90.007	66.268	56.657	42.225	37.380	36.302	**35.801**	36.993	37.012

Consequently, the relevance of a candidate activity or sub-workflow act_{cnd} and an activity stub in the requirement act_{rq} is calculated as follows:

$$relevance(act_{cnd}, act_{rq}) = \alpha \times sim_T(act_{cnd}, act_{rq}) + (1-\alpha) \times sim_S(act_{cnd}, act_{rq}), \quad (10)$$

where sim_T represents the similarity for representative topics, and it is reflected by distances between representative topics. Intuitively, the smaller the distance is, the larger the sim_T is between a certain pair of topics. It returns a value between 0 and 1, where 0 means totally different and 1 means the equivalence. sim_S represents the semantic similarity between short documents, and it is calculated by adopting the minimum cost and maximum flow algorithm, where *WordNet* is used to calculate the semantic similarity for words. The calculation method of sim_S refers to the part of semantic similarity for activity name in our previous work [13]. sim_S returns a value between 0 and 1, where 0 means totally different and 1 means the equivalence. The factor $\alpha \in [0, 1]$ in Formula 10 reflects the relative importance of sim_T with respect to sim_S. When candidate activities and sub-workflows are determined according to the value of their relevance, top $K1$ candidates are recommended for a certain activity stub. For instance, α is set to 0.7, and $K1$ is set to 7, in our experiments.

4.2. Crossing-Workflow Fragments Discovery

This section proposes to discover crossing-workflow fragments, where relations prescribed by AKG are obtained to connect activities or sub-workflows for respective activity stubs in the requirement. Relevant fragments from various workflows are discovered and composed. Leveraging candidate activities and sub-workflows, invocation relations are instantiated and examined the statement that whether they are contained in AKG. Suitable relations are applied to generate crossing-workflow fragments, and these fragments are examined and ranked through balancing their structural and semantic similarities.

Step 1: Instantiated relations examination based on AKG. This procedure is presented by Algorithm 3, where $RQ = \{ACT_{rq}, LNK_{rq}\}$ gives a certain requirement, ACT_{rq} refers to a set of activity stubs, and LNK_{rq} represents links connecting activity stubs in RQ. In particular, the symbol $|ACT_{rq}|$ denotes the number of activity stubs contained in ACT_{rq} while $|LNK_{rq}|$ represents the number of links contained in LNK_{rq}. Without loss of generality, RQ is assumed to have a single initial activity stub act_{int}. To facilitate the fragment discovery procedure, activity stubs are numbered in an incremental manner by adopting the breadth-first search method upon RQ starting at act_{int}. For a link specified in the requirement, the sets of candidate activities and sub-workflows are obtained respecting its source and sink activity stubs (lines 2-3). For an activity stub act^i_{rq} contained in ACT_{rq}, $ACT^i_{cnd} \in ACT_{cnd}$ represents the set of candidate activities and sub-workflows generated in Section 4.1. Candidate activities or sub-workflows in ACT^{src}_{cnd} compose as a series of relations rl with those in ACT^{snk}_{cnd}. rl is denoted as $(act^{src}_{cnd}, r, act^{snk}_{cnd})$, where r refers to the invocation relation upon activities or sub-workflows evidenced in current scientific workflows. rl is verified through a query function $QryAKG$ conducted upon AKG (line 4), and this query statement is presented as follows:

MATCH $(e_1 : E\{act^{src}_{cnd}\}) - [r_1 : R\{r\}] -> (e_2 : E\{act^{snk}_{cnd}\})$
RETURN e_1 as $Source$, r_1 as Rel, e_2 as $Sink$

Algorithm 3 Instantiated Relations Examination

Require:
- $RQ = \{ACT_{rq}, LNK_{rq}\}$: a requirement specification
- $ACT_{cnd} = \{ACT^i_{cnd}\}$: a set of candidate activities or sub-workflows for each $act^i_{rq} \in ACT_{rq}$ where $i \in [1, |ACT_{rq}|]$
- AKG: an activity knowledge graph in Section 2

Ensure:
- $NACT_{cnd}$: a new candidate set for activity stubs contained in ACT_{rq}

1: **for all** $lnk_i \in LNK_{rq}$ **do**
2: $ACT^{src}_{cnd} \in ACT_{cnd}$, $ACT^{snk}_{cnd} \in ACT_{cnd} \leftarrow$ get the sets of candidate activities and sub-workflows for source $lnk_i.act^{src}_{rq} \in ACT_{rq}$ or sink activity stub $lnk_i.act^{snk}_{rq} \in ACT_{rq}$
3: **for** $\forall\ act^{src}_{cnd} \in ACT^{src}_{cnd}$ **and** $\forall\ act^{snk}_{cnd} \in ACT^{snk}_{cnd}$ **do**
4: **if** $QryAKG(rl \leftarrow (act^{src}_{cnd}, r, act^{snk}_{cnd}), AKG) = null$ **then**
5: **continue**
6: **end if**
7: $lnk_i.RL \leftarrow lnk_i.RL \cup \{rl\}$
8: $NACT^{src}_{cnd} \leftarrow NACT^{src}_{cnd} \cup \{act^{src}_{cnd}\}$
9: $NACT^{snk}_{cnd} \leftarrow NACT^{snk}_{cnd} \cup \{act^{snk}_{cnd}\}$
10: **end for**
11: **end for**

where the **MATCH** clause specifies the relation as constraints. Entities are represented with "()" brackets (e.g., (e_1: $E\{act^{src}_{cnd}\}$) and edges with "[]" brackets (e.g., [r_1: $R\{r\}$]). Filters for labels are denoted following the node separated with the symbol ":", and (e_1: E) represents a node e_1 that must match to a certain entity label in E. Certain values for properties can be specified within "{}" (e.g., $\{act^{src}_{cnd}\}$). The **RETURN** clause aims to project output variables. The triple ($Source$, Rel, $Sink$) is (act^{src}_{cnd}, r, act^{snk}_{cnd}) when rl exits in relations prescribed by AKG, or is set to $null$ otherwise. rl is inserted into the candidate relation set as the corresponding link (line 7). To avoid unnecessary combinations, unrelated candidate activities or sub-workflows are excluded. Besides, source and sink activity of rl, i.e., act^{src}_{cnd} and act^{snk}_{cnd}, are inserted into new candidate sets $NACT^{src}_{cnd}$ and $NACT^{snk}_{cnd}$ for corresponding activity stubs in ACT_{rq} (lines 8-9).

The time complexity of Algorithm 3 is $O(l * (K1)^2)$, where l is the maximum number of links connecting activity stubs in a certain requirement. For an activity stub, $K1$ candidates are mostly recommended. In this algorithm, the time complexity is reflected by *for* loops of line 1, and line 3. In fact, there are $l * (K1 * K1)$ times of iteration in the worst case for the procedure of lines 4-9, and thus the time complexity of this algorithm is $O(l * (K1)^2)$.

Step 2: Crossing-workflow fragments generation. Based on candidate instantiation relations, the crossing-workflow fragment generation strategy is presented by Algorithm 4, where fragments are initially constructed through the inclusion of a certain activity or sub-workflow (lines 1-7). Given a fragment frg in FRG, we examine each candidate activity or sub-workflow act^k_{cnd} respecting the activity stub act^i_{rq}, and matched relations, to evaluate whether frg can be extended to generate new fragments. A tag is adopted to specify the fact that whether act^k_{cnd} has extended certain relations to produce new fragments or not (line 10). The set of indexes for links in LNK_{rq} are obtained associated with act^i_{rq} being processed and those activity stubs that have already been processed, where each link index lnk_{idx} should be handled independently (lines 11-13). Involved candidate relations are decomposed into the respective source and sink activities, which allows the indexes of their source and sink activity stubs, i.e., idx_1 and idx_2, to be determined for promoting the selection of extended relations (line 14).

When source and sink activities of a candidate relation are the same as (i) the activity for the corresponding index idx_1 of frg and the candidate activity being processed act^k_{cnd}, respectively, or (ii) act^k_{cnd} and the activity at the corresponding index idx_2 of frg (line 15), this fact indicates that a matched relation is found, and the tag can be set to *true* (line 16). In this setting, the growth procedure for new fragments is executed upon frg. In particular, act^k_{cnd} is inserted into the activity set of this new fragment while the matched relation is supplemented to its relation set (lines 17-18). Since act^k_{cnd} and the associated activity in frg are deterministic, there is at most one suitable relation in the corresponding candidate relation set. Once this relation is discovered, the iteration for its candidate relation set terminates (line 20). On the other hand, when no suitable relations are discovered, i.e., the tag is set to *false*, we need to simply insert act^k_{cnd} into a new fragment without corresponding relations (lines 24-27), which contributes to the fragment discovery for the following phase act^{i+1}_{rq}. When the extension generation task of frg is complete, frg is removed from the set FRG (line 29).

The time complexity of Algorithm 4 is $O(n * (K1)^2 * s * d)$, where n is the maximum number of activities in a

Algorithm 4 Crossing-Workflow Fragments Generation

Require:
- $RQ = \{ACT_{rq}, LNK_{rq}\}$: a requirement specification
- $NACT_{cnd}$ = a new candidate set for corresponding activity stubs in ACT_{rq} provided by Algorithm 3

Ensure:
- $FRG = \{frg\}$: a set of crossing-workflow fragments

1: **for all** $act^i_{rq} \in ACT_{rq}$ where $NACT^i_{cnd} \neq \emptyset$ **do**
2: **if** $FRG = \emptyset$ **then**
3: **for all** $act^k_{cnd} \in NACT^i_{cnd}$ **do**
4: $frg.ACT \leftarrow \{act^j_{cnd}\}$; $FRG \leftarrow FRG \cup \{frg\}$
5: **end for**
6: **continue**
7: **end if**
8: **for all** $frg \in FRG$ **do**
9: **for all** $act^k_{cnd} \in NACT^i_{cnd}$ **do**
10: $tag^k_{cnd} \leftarrow$ **false**
11: $IDX \leftarrow$ get indexes for links in LNK_{rq} associated with act^i_{rq} and processed activity stubs
12: **for all** $idx \in IDX$ **do**
13: **for all** $rl \in lnk_{idx}.RL$ **do**
14: idx_1, $idx_2 \leftarrow$ get the indexes of activities in frg related to $rl.act^{src}_{cnd}$ or $rl.act^{snk}_{cnd}$
15: **if** ($rl.act^{src}_{cnd} = act^{idx_1}_{frg}$ **and** $rl.act^{snk}_{cnd} = act^k_{cnd}$) **or** ($rl.act^{src}_{cnd} = act^k_{cnd}$ **and** $rl.act^{snk}_{cnd} = act^{idx_2}_{frg}$) **then**
16: $frg_n \leftarrow frg$; $tag^k_{cnd} \leftarrow$ **true**
17: $frg_n.ACT \leftarrow frg_n.ACT \cup \{act^k_{cnd}\}$
18: $frg_n.RL \leftarrow frg_n.RL \cup \{rl\}$
19: $FRG \leftarrow FRG \cup \{frg_n\}$
20: **break**
21: **end if**
22: **end for**
23: **end for**
24: **if** $tag^k_{cnd} =$ **false then**
25: $frg_n \leftarrow frg$
26: $frg_n.ACT \leftarrow frg_n.ACT \cup \{act^k_{cnd}\}$
27: $FRG \leftarrow FRG \cup \{frg_n\}$
28: **end if**
29: **end for**
30: $FRG \leftarrow FRG$ - $\{frg\}$
31: **end for**
32: **end for**

certain requirement, s is the upper number of connected activity stubs for each activity stub in the requirement, and d is the upper bound number of instantiation relations between two activity stubs. Generally, for an activity stub, $K1$ candidates are mostly recommended. In Algorithm 4, the time complexity is reflected by *for* loops of line 1, line 8, line 9, line 12, and line 13. Especially, line 9 represents the iteration of initialization fragments, which are initially constructed through the inclusion of the first activity or sub-workflow stub. In this setting, $K1$ fragments are generated. Therefore, there are $(n-1)*K1*K1*s*d$ times of iteration in the worst case for the procedure of lines 14-19, and hence, the time complexity of this algorithm is $O(n * (K1)^2 * s * d)$.

Algorithm 5 Candidate Fragments Recommendation

Require:
- $RQ = \{ACT_{rq}, LNK_{rq}\}$: a requirement specification
- $FRG = \{frg\}$: a set of generated crossing-workflow fragments provided by Algorithm 4

Ensure:
- $FRG_{rec} = \{frg\}$: a set of top $K2$ crossing-workflow fragments where $frg = \{ACT, RL, sim\}$

1: **for all** $frg \in FRG$ **do**
2: $frg.ACT \leftarrow$ remove activities irrelevant with $\forall\ rl \in frg.RL$
3: **for all** $act \in frg.ACT$ **do**
4: $act_{rq} \leftarrow$ get the activity stub in ACT_{rq} *w.r.t.* act
5: $sim_{sm} \leftarrow sim_{sm} + SimCal(act_{rq}, act)$
6: **end for**
7: $sim_{st} \leftarrow |frg.RL|/|LNK_{rq}|$
8: $sim_{sm} \leftarrow sim_{sm}/|frg.ACT|$
9: $frg.sim \leftarrow \beta * sim_{st} + (1 - \beta) * sim_{sm}$
10: **end for**
11: $FRG \leftarrow \{frq\}$ where $frq.sim$ is among the top $K2$

Step 3: Candidate fragments recommendation. This procedure is presented by Algorithm 5. The activity set of afore-generated fragments may contain irrelevant activities. Therefore, the activity set needs to be updated according to activities related to the relation set (line 2). Meanwhile, the fragment similarity $frg.sim$ is calculated as the following two parts, e.g., structural similarity sim_{st} and semantic similarity sim_{sm} (line 8), where $\beta \in [0, 1]$ reflects the relative importance of sim_{st} against sim_{sm}. Note that sim_{st} corresponds to the structural similarity between frg and RQ, which is the relation ratio for the number of relations in $frg.RL$ to the number of total relations in LNK_{rq} (line 7). Besides, sim_{sm} represents the average semantic similarity between frg and RQ, which is calculated by leveraging similarities between each activity in frg and the corresponding activity stub, and the number of activities in $frg.ACT$. Similarities are calculated by adopting the function $SimCal$ (line 5).

Generally, fragments with the larger structural similarity should be more appropriate to be recommended. In this setting, sim_{st} is assigned a larger weight than sim_{sm} (i.e., β is set to 0.7 in our experiments). The similarity of each fragment returns a value between 0 and 1, where 0 means totally different and 1 means the equivalent. Consequently, top $K2$ (like 10) fragments $\{frg\}$ are selected and recommended according to the values of their fragment similarities (line 10).

The time complexity of Algorithm 5 is $O(f * n)$, where f is the upper number of crossing-workflow fragments generated by Algorithm 4, and n is the maximum number of activities in a fragment. In fact, the time complexity is reflected by *for* loops of line 1, and line 3, and the procedure of lines 4-5 is to be iterated $f * n$ times at most. Therefore, the time complexity of this algorithm is $O(f * n)$.

5. Experimental Setting

A prototype has been implemented by Java programs, and scientific workflows in the category of *Taverna 2* of the *myExperiment* repository are collected and cleansed, where there are 1573 scientific workflows in this category till to June 12, 2018. Experiments are conducted upon a desktop with an Intel(R) Core(TM) i5-6500 processor, a 6.00GB memory, and a 64-bit Windows 7 system, and topic models are generated by a tool executed on Linux.

5.1. Data Cleansing

Before conducting the experimental evaluation, scientific workflows are cleansed as follows:

- The titles or text descriptions of scientific workflows lacks specification. In this case, similarity computation for activities cannot be conducted, and the topic discovery is inaccurate. Therefore, these workflows are removed from the dataset.
- Activities corresponding to slim services are mostly *glue* codes, and they do not represent invocation relations between functionalities. Therefore, activities with the slim service type are filtered out.

As aforementioned, the semantic similarity between words is computed by *WordNet*. However, improper words that are not recognized by *WordNet* are contained in names or text descriptions of activities and sub-workflows. We have to manually handle them case-by-case as follows:

- 334 abbreviations are used in the bioinformatics domain, but they cannot be recognized by *WordNet*. An example is "*snp*", which means "*single nucleotide polymorphism*" after searching on the web. A conversion table is manually established, and it aims to transform these abbreviations into their full descriptions. If a full description cannot be found through browsing the web like the term "*martijn*", it is assumed to be dissimilar to any other words, and such an abbreviation is removed.
- As to the abbreviation that is just a part of a word, we can convert it to the full description. An example is "*bio*" which corresponds to "*biology*".

- Words, whose meaning is not clear and information is incomplete, are complemented according to the context and web network. An example is "*cpath*" rewriting as "*Canadian Professional Association for Transgender Health*" in the life science domain.
- When two or several words are joint without delimiters, they are separated manually. An example is "*documentfind*", which is transferred into two words "*document*" and "*find*".

After handling the names and descriptions of activities and sub-workflows, stopwords words such as *can*, *of*, and *and*, are removed. A table including 883 stopwords is constructed according to the dataset, and it helps the dataset cleansing procedure. Besides, words like *accessed*, *accesses*, and *accessing* have a common word root *access*. Affixes are removed to keep the root only. This is important for topic model algorithms. Otherwise, they are treated as different word entities, and thus, their topic relevance is inaccurate. To remedy this issue, the lemmatization technology is applied by establishing common reduction rules, which are divided into general deformation rules for nouns and verbs, deformation rules for irregular verbs or nouns, and so on.

5.2. Experimental Setup

Without loss of generality, scientists are assumed to be unambiguous about functional components required to support their experiments, although they can hardly identify reusable functional fragments from legacy workflows in the repository, due to the large number and relatively complex in the structure of workflows. As presented by Fig. 3 in our previous work [9], scientific workflows in the *Taverna 1* category of the *myExperiment* repository are mostly fallen into 6 clusters. This observation is also supported by AKG we have constructed where a workflow may connect with at most 5 to 6 other workflows. Consequently, based on AKG, we carefully make 20 crossing-workflow fragments as testing fragments, which contains activities or sub-workflows originally from 1 to 6 workflows, and the majority (12 out of 20 in our experiments) spans 2 to 4 workflows.

Scientific workflows are typically small in the number of activities. A statistic has been made for the number of activities contained in workflows in the *Taverna 2* category of the *myExperiment* repository, and the result shows that roughly 86.046% of workflows contain no more than 11 activities. Based on this observation, we set 3 out of 20 testing fragments containing over 11, while the others containing less than 11 activities and sub-workflows.

Parameters of BTM models are presented in Table 2, where *alpha*, *beta*, *niter*, and *save_step* are set to default values as $50/K$, 0.01, 1000 and 100, respectively, and W is determined by the word number contained in the corpus. The number of topics K is determined by examining the training corpus, where experimental results show that an optimal K does not change when most of the corpus remains invariant contents, although a small amount of data is appended. Based on this result, K is set to 370 where the criteria including perplexity and topic similarity are optimal as shown in Table 1.

Table 2: BTM Model Parameter Settings.

Symbol	Description
K	the topic number set by Per_Var
W	the word number determined by the corpus
alpha	the parameter $\alpha = 50/K$ in BTM
beta	the parameter $\beta = 0.01$ in BTM
niter	the number of iterations: $niter = 1000$
save_step	the number of iterations when saving a temporary result, and $save\_step = 100$
data_dir	the path to the input document of BTM training
res_dir	the path to the output document of BTM training

5.3. Crossing-workflow Fragments Discovery When No Changes Are Applied to Sample Fragments

Experiments are conducted when sample fragments remain as they are retrieved from the dataset. An expected result should be exactly matched fragments compared with the fragment of requirement. Experiments for 20 sample crossing-workflow fragments return the *right* recommendations, which contain original sample fragments, and these fragments are ranked as the first in terms of the fragment similarity value. For instance, the sample requirement as shown in Fig. 4 is represented as follows:

- $E = \{swf_{sub3}, act_4, act_5, act_6, act_7, act_8, swf_{sub9}\}$
- $R = \{Invok\}$
- $S = \{(act_5, Invok, act_4), (act_4, Invok, swf_{sub3}), (act_5, Invok, swf_{sub9}), (act_5, Invok, act_6), (act_6, Invok, act_7), (act_7, Invok, act_8), (act_8, Invok, swf_{sub9})\}$

where 7 relation triples are included, and crossing-workflow fragments are discovered and ranked as follows:

- *Candidate 1* $= \{frg.ACT = (swf_{sub3}, act_4, act_5, act_6, act_7, act_8, swf_{sub9}), frg.RL = \big((act_5, Invok, act_4), (act_4, Invok, swf_{sub3}), (act_5, Invok, swf_{sub9}), (act_5, Invok, act_6), (act_6, Invok, act_7), (act_7, Invok, act_8), (act_8, Invok, swf_{sub9})\big), frg.sim = 1.0\}$
- *Candidate 2* $= \{frg.ACT = (act_{330}, act_4, act_5, act_6, act_7, act_8, swf_{sub9}), frg.RL = \big((act_5, Invok, act_4), (act_4, Invok, act_{330}), (act_5, Invok, swf_{sub9}), (act_5, Invok, act_6), (act_6, Invok, act_7), (act_7, Invok, act_8), (act_8, Invok, swf_{sub9})\big), frg.sim = 0.985\}$
- .....
- *Candidate 5* $= \{frg.ACT = (swf_{sub3}, act_4, act_5, act_{324}, act_7, act_8, swf_{sub9}), frg.RL = \big((act_5, Invok, act_4), (act_4, Invok, swf_{sub3}), (act_5, Invok, swf_{sub9}), (act_5, Invok, act_{324}), (act_{324}, Invok, act_7), (act_7, Invok, act_8), (act_8, Invok, swf_{sub9})\big), frg.sim = 0.961\}$

This experiment demonstrates that most of the same or similar relations are discovered, and most suitable candidate fragments can be recommended.

To compare the effectiveness of our approach against the strategy that recommends single workflows, we conduct experiments where the requirement as shown in Fig. 4 is chosen. The similarity for this requirement and workflows are calculated as the semantic transition-labeled graph edit distance [20]. Top three legacy workflows are recommended as follows:

- *Candidate 1* = "*Phenotype to pubmed*", *sim* = 0.598
- *Candidate 2* = "*Pathway and Gene to Pubmed*", *sim* = 0.592
- *Candidate 3* = "*Gene to Pubmed*", *sim* = 0.560

For example, candidate 1 represents the first workflow to be recommended, whose title is *Phenotype to pubmed*. Workflows of the sample requirement and candidate *1* are partially relevant with *pubmed*, which represents a free database of the biomedical paper search. This result demonstrates that this requirement can hardly be achieved by any single legacy workflow, and the discovery of crossing-workflow fragments is more appropriate.

5.4. *Crossing-workflow Fragments Discovery When Changes Are Applied to Sample Fragments*

Novel requirements may not be equivalent or similar to crossing-workflow fragments in the repository. In this case, sample fragments are changed to generate sample requirements. Changing operations include *insertion*, *deletion*, and *replacement* as follows:

- *Insertion.* Sub-workflows and activities not included in a sample workflow fragment swf_{smpl} are inserted as a component of a testing fragment swf_{tst}. If some sub-workflows or activities are specified in another fragment swf_{ano} or newly-made fragment swf_{new}, and they express similar topics as swf_{smpl}, swf_{tst} is usually more appropriate to reuse these similar sub-workflows or activities. For instance, activity act_{i1} named "*extractDates*" is inserted into swf_{tst}, and another activity act_{i2} named "*concat_abstract_ids*" is contained as shown in Fig. 5.
- *Deletion.* Sub-workflows or activities are deleted from a sample fragment swf_{smpl}. Examples are the sub-workflow swf_{sub3} named "*Retrive_abstracts*" and activity act_8 named "*clean_text*", which have been deleted as shown in Fig. 5.
- *Replacement.* Sub-workflows and activities in a sample fragment swf_{smpl} are replaced by the others that are not specified in swf_{smpl}. Similar to the scenario of *insertion*, if activities or sub-workflows have similar topics to the replaced activities or sub-workflows, the testing fragment is usually more inclined to reuse them in swf_{smpl}. An example is the activity act_6 = "*inppXML*" that is to be replaced by a newly constructed activity act_{r6} = "*inpp*", and it has similar descriptive information with act_6 as shown in Fig. 5.

Experiments are conducted for changes applied upon sample fragments. Sample fragments swf_{smpl} are changed

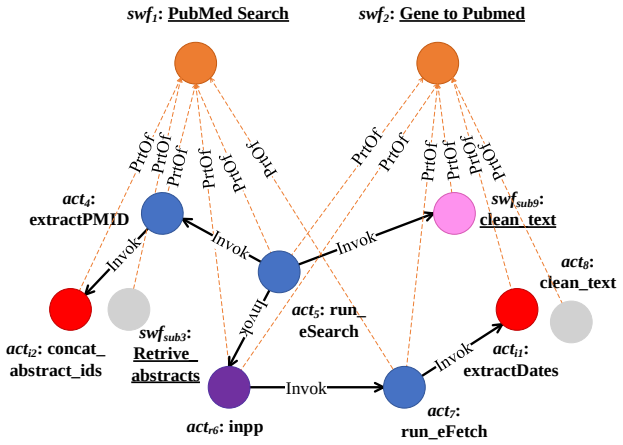

Figure 5: The testing requirement is created from Fig. 4, and changes are made through inserting 2 activities, deleting 1 sub-workflow and 1 activity, and replacing 1 activity.

through *insertion*, *deletion*, and *replacement* operations. Besides, some novel activities or sub-workflows are constructed in sample fragments, and they are more in line with new features of novel experiments such that certain requirements put forward by scientists may be different, but similar to existing ones. Consequently, 10 fragments are generated to conduct experiments of special requirements. An example is shown in Fig. 5, where 2 activities are inserted, 1 sub-workflow and 1 activity are deleted, and 1 activity is replaced. Specifically, activity act_{i1} named "*extractDates*" is inserted into act_7 to build a new relation triple (act_7, *Invok*, act_{i1}). Besides, the new relation triple (act_4, *Invok*, act_{i2}) is generated, where an inserted activity act_{i2} named "*concat_abstract_ids*" is reconnected to act_4. Sub-workflow swf_{sub3} and activity act_8 are deleted, and activity act_6 is replaced by a newly constructed activity act_{r6} that has similar topics as act_6.

Regarding these 10 testing workflow fragments, results for 4 testing workflow fragments return the *right* recommendations that contain sample fragments, and they are ranked in the first place regarding the fragment similarity. An example is swf_{tst27}, which spans 4 workflows (e.g., swf_{1661}, swf_{1662}, swf_{1687}, and swf_{1598}) and replaces 1 activity. The representation based on *AKG* for swf_{tst27} is presented as follows:

- $E = \{act_2, act_7, act_{69}, act_{70}, act_{201}, act_{124}\}$
- $R = \{Invok\}$
- $S = \{(act_{124}, Invok, act_{70}), (act_{69}, Invok, act_{124}), (act_{69}, Invok, act_2), (act_2, Invok, act_7), (act_{201}, Invok, act_7)\}$

The recommended workflow fragments with similarity values are listed as follows:

- *Candidate 1* = $\{frg.ACT = (act_2, act_7, act_{69}, act_{70}, act_{201}, act_{124}), frg.RL = ((act_{124}, Invok, act_{70}), (act_{69}, Invok, act_{124}), (act_{69}, Invok, act_2), (act_2, Invok, act_7), (act_{201}, Invok, act_7)), frg.sim = 1.0\}$

- .....
- *Candidate 5* = {$frg.ACT = (act_2, act_7, act_{69}, act_{70}, act_{201}, act_{123})$, $frg.RL = ((act_{123}, Invok, act_{70}), (act_{69}, Invok, act_{123}), (act_{69}, Invok, act_2), (act_2, Invok, act_7), (act_{201}, Invok, act_7))$, $frg.sim = 0.960$}

This result shows that, when changes are applied upon sample workflow fragments, our technique can discover appropriate crossing-workflow fragments in most scenarios.

6. Experimental Evaluation

6.1. Performance Metrics

The metrics *precision*, *recall*, and *F1* are adopted for our evaluation purpose. Given a testing fragment swf_{tst}, a reusable fragment swf_{ept} contained in the repository is assumed to be included in the *expected* list of recommendations (denoted SWF_{ept}) when the similarity between swf_{tst} and swf_{ept} is no smaller than a pre-specified threshold $thrd_{ept}$. In our experiments, an activity stub in swf_{tst} is replaced by relevant activities and sub-workflows in the dataset, to obtain a series of crossing-workflow fragments, where links on activities and sub-workflows are checked whether they are retained based on the relations specified upon scientific workflows in the repository. Importantly, similarities of fragments are calculated by line 8 of Algorithm 5 through balancing their structural and semantic similarities with respect to swf_{tst}. We adopt the notation SWF_{rec} to denote the set of workflow fragments, which are actually recommended by our technique, and *precision*, *recall*, and *F1* are computed as follows:

$$precision = (|SWF_{ept} \cap SWF_{rec}|) \div |SWF_{rec}| \quad (11)$$

$$recall = (|SWF_{ept} \cap SWF_{rec}|) \div |SWF_{ept}| \quad (12)$$

$$F1 = \frac{(precision * recall)}{(precision + recall)/2} \quad (13)$$

where the symbol "$|SWF_{rec}|$" refers to the number of fragments in the set SWF_{rec} while the symbol "$|SWF_{ept}|$" denotes the number of fragments in the set SWF_{ept}. The symbol "$|SWF_{ept} \cap SWF_{rec}|$" represents the number of fragments contained in both SWF_{ept} and SWF_{rec}.

6.2. Baseline Techniques

The following two methods are chosen as the baselines:

- *ClusteringRec* [21]: In this technique, *abstract* activities are discovered and represented the functional relevance of activities. Following this manner, a modularity-based clustering algorithm is developed to generate activity clusters. Specifically, *core* activities [9] are determined to represent certain clusters, and they correspond to the most *representative* activities in these clusters. When a requirement is satisfied, the target cluster is determined with respect to each activity stub, where their activities or sub-workflows are discovered and ranked as candidate activities or sub-workflows. These candidate activities and sub-workflows construct a series of crossing-workflow fragments according to invocation relations among them. Fragments are identified, ranked and recommended according to their similarity values.
- *Sub-graphRec* [13]: This sub-graph matching algorithm discovers and recommends fragments according to certain requirements. Given a requirement specified in terms of a workflow fragment, this technique discovers candidate fragments in a constructed activity network model, where edges reflect invocation relations between activities. Similarities of candidate fragments are calculated through considering their structural and semantic similarities (like line 8 of Algorithm 5) with respect to the workflow fragment of requirements. Top $K2$ fragments are recommended according to fragment similarity values.

6.3. Evaluation Results

This section presents and discusses the evaluation results of our technique (denoted *AKGRec*), *Sub-graphRec*, and *ClusteringRec*, when the following three parameters are considered as influential factors:

- $thrd_{ept}$: the pre-specified threshold of the similarity for generating SWF_{ept} as presented in Section 6.1.
- $K2$: the number of recommended fragments as referred to line 11 of Algorithm 5.
- β: the relative importance of the structural similarity with respect to the semantic similarity for the calculation of fragment similarity as referred to line 9 of Algorithm 5.

6.3.1. Impact of $thrd_{ept}$

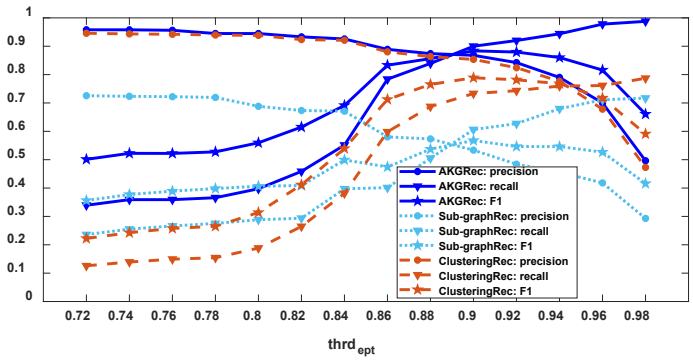

Figure 6: *Precision*, *recall* and *F1* for *AKGRec*, *Sub-graphRec*, and *ClusteringRec*, when $K2$ is set to 10, β is set to 0.7, and $thrd_{ept}$ is set from 0.72 to 0.98 with an increment of 0.02.

To investigate the impact of $thrd_{ept}$ to *precision*, *recall*, and *F1*, we set $K2$ to 10, β to 0.7, and $thrd_{ept}$ to a

value from 0.72 to 0.98 with an increment of 0.02. Fig. 6 shows that precision drops and recall increases along with the increase of $thrd_{ept}$, and *AKGRec* performs better than *Sub-graphRec* and *ClusteringRec* in precision and recall. Specifically, the majority of recommended fragments of *Sub-graphRec* are not expected. In fact, *Sub-graphRec* emphasizes the *structural* similarity of recommended fragments with respect to the requirement, but the semantically matching of activities is not the focus. Therefore, the precision and recall are relatively lower for *Sub-graphRec* than that of *AKGRec* and *ClusteringRec*. The precision of *ClusteringRec* is quite high, since recommended fragments are mostly included in SWF_{ept}. On the contrary, the recall of *ClusteringRec* is relatively low. After carefully analyzing the experiments, it is observed that activities are unevenly assigned to various clusters, and some clusters may contain quite a few candidate activities or sub-workflows. Consequently, there may have no enough fragments generated by *ClusteringRec* for the recommendation.

Fig. 6 shows that precision decreases, and recall increases, along with the increasing of $thrd_{ept}$ for *AKGRec*, *Sub-graphRec*, and *ClusteringRec*, and the variation of *F1* values shows that *AKGRec* performs better than *Sub-graphRec* and *ClusteringRec*. In fact, SWF_{rec} does not change when $K2$ is set to a certain value. Fragments in SWF_{ept} may be fewer when $thrd_{ept}$ is set to a relatively large value. Therefore, more fragments in SWF_{rec} may be missed in SWF_{ept}, which makes the decrease of the precision according to Formula 11. Fig. 6 shows that precision for *AKGRec* and *ClusteringRec* is relatively stable when $thrd_{ept}$ changes from 0.72 to 0.84, since the similarity value for most expected fragments is within these two ranges. The number of fragments in SWF_{ept} decreases to an extent when $thrd_{ept}$ is set from 0.86 to 0.98, since expected workflows whose similarity value is within this range is quite few. Besides, the difference between the sets of $SWF_{ept} \cap SWF_{rec}$ and SWF_{ept} is decreasing, and thus, recall increases for *AKGRec*, *Sub-graphRec* and *ClusteringRec* according to Formula 12.

6.3.2. Impact of $K2$

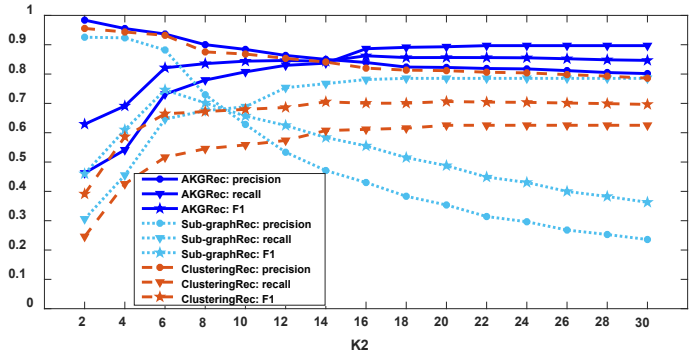

Figure 7: *Precision*, *recall* and *F1* for *AKGRec*, *Sub-graphRec* and *ClusteringRec* when $thrd_{ept}$ is set to 0.86, β is set to 0.7 and $K2$ is set from 2 to 30 with an increment of 2.

To investigate the impact of $K2$ to *precision*, *recall*, and *F1*, we set $thrd_{ept}$ to 0.86, β to 0.7, and $K2$ to a value from 2 to 30 with an increment of 2. As shown in Fig. 7, the precision and recall are larger for *AKGRec* than *Sub-graphRec* and *ClusteringRec*, due to the similar reason as discussed at the paragraph for Fig. 6. In particular, when $K2$ is set to relatively large values, Fig. 7 shows that the precision of all three technique begins to decrease, since more fragments should be discovered and recommended, and they are actually not that relevant and may not exist in SWF_{ept}. Note that the precision for *Sub-graphRec* does not perform well, since a large number of fragments are obtained by the sub-graph matching method. When $K2$ changes from 2 to 16, recall of all three technique increases to a large extent, since more expected fragments are recommended. Besides, recall becomes relatively stable when $K2$ changes from 16 to 30, since most expected workflows have been discovered and included in SWF_{ept}.

This figure also shows that *F1* values of *AKGRec* and *ClusteringRec* rise steadily, but those of *Sub-graphRec* appear a decreased trend when $K2$ is set to the value starting at 6. This means that *AKGRec* and *ClusteringRec* perform better than *Sub-graphRec* in this situation.

6.3.3. Impact of β

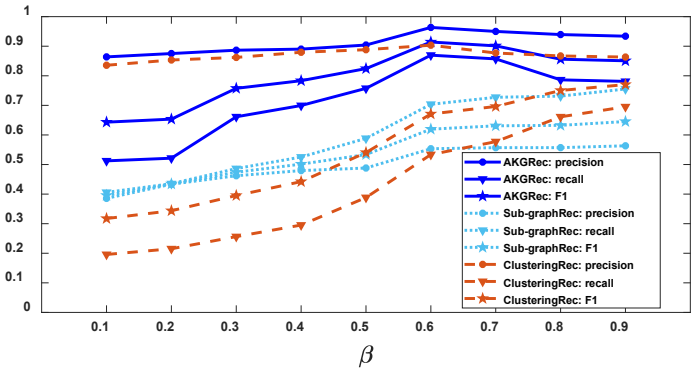

Figure 8: *Precision*, *recall* and *F1* for *AKGRec*, *Sub-graphRec* and *ClusteringRec*, when $thrd_{ept}$ is set to 0.86, $K2$ is set to 10, and β is set from 0.1 to 0.9 with an increment of 0.1.

As discussed in Section 4.2, the similarities of recommended fragments with a certain requirement are impacted by β. To investigate this impact to *precision*, *recall*, and *F1*, we set $thrd_{ept}$ to 0.86, $K2$ to 10, and β to a value from 0.1 to 0.9 with an increment of 0.1. Fig. 8 shows experimental results of β for three techniques. Due to the similar reason as discussed for Fig. 6 and 7, *AKGRec* performs better in precision and recall than *Sub-graphRec* and *ClusteringRec*. Generally, precision and recall increase along with the increasing of β, which shows the strong significance of structural relevance to recommended fragments. Concretely, SWF_{rec} does not change when $K1$ is set to a certain value, and precision increases when β changes from 0.1 to 0.6, since more recommended fragments are contained in SWF_{ept}. As to *Sub-graphRec*, semantically

matching is not considered in the sub-graph matching algorithm. Therefore, most discovered fragments are not contained in SWF_{ept} and the precision is relatively low. Fig. 8 also shows that when the importance of structural information is enhanced, expected fragments in $thrd_{ept}$ are more consistent with fragments recommended by *AKGRec* and *ClusteringRec*. However, the importance of semantics is gradually decreased from the fragment similarity when β is set from 0.6 to 0.9, and results show that precision begins to decrease, since more recommended fragments are not contained in SWF_{ept}, in case the fragment similarity relies on structure while less regard to semantics.

This figure also shows that, when $|SWF_{ept} \cap SWF_{rec}|$ increases, recall increases as well for three techniques. Specifically, recall for *AKGRec* has the similar trend as its precision. Especially, a high recall occurs when β is set to 0.6. However, the recall of *ClusteringRec* is lower than *AKGRec* and *Sub-graphRec*. In fact, the number of recommended fragments is not enough in comparison with recommended fragments as explained for $K2$, which is affected by the clustering effect and selection strategy of candidate activities or sub-workflows. Generally, the factors of structure and semantic similarity should be balanced to facilitate candidate fragment recommendation. *AKGRec* performs better than *Sub-graphRec* and *ClusteringRec* with current parameter settings, and the structure relevance performs an important role than activity semantics in the fragment similarity calculation. Also shown by the variation trend of *F1* values in Fig. 8, fragments with a higher structural similarity should be more appropriate to be recommended.

6.3.4. Evaluation when $thrd_{ept}$, $K2$ and β are optimum

Table 3: Comparison of *AKGRec* with *Sub-graphRec* and *ClusteringRec* when $thrd_{ept}$, $K2$, and β are set to optimal values.

	Precision	*Recall*	*F1*
AKGRec	0.959	0.862	0.908
Sub-graphRec	0.852	0.422	0.564
ClusteringRec	0.454	0.517	0.483

According to experimental results presented in previous sections, the optimum of $thrd_{ept}$, $K2$, and β are determined as 0.86, 10, and 0.6, respectively. With this parameter setting, as shown in Table 3, experimental results show that *AKGRec* performs better than *Sub-graphRec* and *ClusteringRec*. Similar to the discussion presented previously, *AKGRec* considers both structure-based and semantic-based characteristics, which can complement each other to achieve an optimal crossing-workflow fragment discovery performance.

Generally, the precision and recall should be balanced somehow to facilitate the discovery and recommendation of crossing-workflow fragments, such that a more number of optimal workflow fragments should be recommended. Results illustrated by Fig. 6 and 7 shows that precision decreases, while recall increases, when $thrd_{ept}$ and $K2$ increase. This result specifies that $thrd_{ept}$ and $K2$ should not be set to relatively large values. Experimental results in Fig. 8 show that the structure relevance should be considered as more important when discovering crossing-workflow fragments to achieve a promising result.

7. Related Work and Comparison

7.1. Crossing-Workflow Fragments Discovery

An efficient discovery of crossing-workflow fragments is of paramount importance to promote the reuse or repurposing of legacy scientific workflows, especially when the requirement of novel scientific experiments is relevant with multiple workflows in the repository. In [13], a crossing-workflow fragments discovery mechanism is proposed, where semantically relevant activities are identified and combined as abstract ones. A network model is constructed to capture invocation relations specified upon abstract activities in legacy workflows. Sub-graph matching algorithm is adopted to discover fragments from the network model according to graph-style requirement specifications, and these fragments are composed of the parts that originated from various workflows. Note that a network model captures *flat* invocation relations, whereas *not hierarchical* relations like parent-child ones, between activities. This suggests that fragments containing activities in various granularities cannot be discovered. This observation drives us to construct a knowledge graph for activities in order to capture both flat invocation and hierarchical parent-child relations between activities, and these relations are represented in a semantic fashion [22]. The discovery of fragments is conducted by a graph query operation upon the knowledge graph, and this is typically achieved by the path query technique [23], such that a path captures a sequence of invocation relations between activities and sub-workflows, and fragments may be assembled partially by multiple workflows with different levels of granularities.

Typical fragments mined from workflows can improve their sharing and reuse when possible. As presented in [10], a two-objective evolutionary algorithm is adopted for the ranking and recommendation of relevant fragments according to certain requirements. Common fragments are extracted through dependency graph calculation in a reconfigurable business process model [11], in order to facilitate the composition of business variations from main business processes. Authors propose an approach to automatically obtain abstractions from low-level provenance data [24], where workflow fragments with a larger occurrence frequency are discovered on workflow execution provenance and associated with certain templates. An efficient index-based mechanism is developed to promote the search and reuse of fragments with various levels of granularities [12]. To summarize, current approaches are to generate structural-relevant fragments from single workflows, whereas the discovery of crossing-workflow fragments has not been explored extensively.

7.2. Workflow Similarity Assessment

Similarity assessment promotes the reuse and repurposing of workflows [4]. Traditional methods take annotations into account [5], where experiments show that semantics represented in the ontology can promote the workflow similarity examination. Annotations and text descriptions have also been adopted in our technique to facilitate the computation of semantic similarity between activities or sub-workflows. Besides, structure-based approaches are developed [6], where a graph-based workflow recommendation is proposed to improve business process modeling, and graph mining method and graph edit distance are used to extract and calculate process patterns from the repository. Generally, these techniques are efficient when rich annotations are provided to describe workflows, and users can be provided by several workflow recommendation strategies to automatically or semi-automatically augment their developing-in-progress workflows, leveraging both structural and semantic similarities between workflows and guiding information extracted from current workflows [25].

Recent attempts to consider data in business process management and the support of data modeling in business process standards have led to the creation of multiple business models with data access [7, 26]. A systematic data-driven approach is proposed to assist situational application development and extract certain information from multiple sources to abstract service capabilities and represent in terms of tags [8]. Different from this data-driven mechanism, we adopt invocation data information to complement the similarity assessment of discovered fragments. Specifically, we have developed a workflow similarity computation technique [9], where workflows in the *myExperiment* repository are transformed into layer hierarchies that reflect hierarchical relations between workflow, its sub-workflow and activities. This technique is to promote the reuse or repurposing of workflows as the whole, and it fails to work when crossing workflow fragments are to be discovered to satisfy novel requirements.

7.3. Service Discovery

The discovery of workflow fragments is closely related to service discovery, whose approaches mostly focus on WSDL-based keyword search [27], and semantic matching based on domain knowledge or ontologies [28], context-awareness [29, 30, 31], or QoS-based discovery [32, 33]. Machine learning techniques are usually adopted to examine service relevance [34]. Authors present an activity clustering method to promote activity reuse, and this technique can improve the performance of the retrieval and recommendation of relevant services. Taking text descriptions of services into consideration, topic models have been used to promote service discovery. As presented in [35], a novel Web service discovery mechanism is proposed, where common topic groups are mined from the service-topic distribution matrix generated by topic modeling methods, and these topic groups promote to match user queries with relevant Web services. A theoretical approach is developed to extract latent service co-occurrence topics, which facilitate the discovery about the trend of service compositions, and promote the precision and recall of the service recommendation [36]. In this paper, latent topics are chosen as an important metric when discovering candidate activities or sub-workflows. Particularly, the structure of Web services is converted into *W*eighted *D*irected *A*cyclic *G*raphs (*WDAG*), and *BTM* is adopted to generate topics [37]. The similarity for the pairs of *WDAGs* takes the topic similarity into computation. Different from current techniques, we adopt representative topics to represent topic distribution, and the relevance of activities and sub-workflows depends on the similarity of their topics and text descriptions. Experimental results demonstrate the efficiency of this mechanism.

8. Conclusion

This paper proposes a novel crossing-workflow fragment discovery mechanism to promote the reuse or repurposing of legacy scientific workflows. Specifically, an *AKG* is constructed to represent flat invocation relations between activities, and hierarchical parent-child relations specified upon sub-workflows and their corresponding activities. The semantic relevance of activities and sub-workflows is quantified by their representative topics generated by *BTM*. Given a requirement specified in terms of a workflow template, individual candidate activities or sub-workflows are discovered leveraging their semantic relevance and short documents, and they are composed into fragments according to relations specified in *AKG*. Candidate fragments are evaluated through balancing their structural and semantic similarities against the requirement specification. Extensive experiments have been conducted, and evaluation results demonstrate that our approach is accurate in discovering appropriate crossing-workflow fragments in comparison with the state of art's techniques.

References

[1] C. A. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, myexperiment: a repository and social network for the sharing of bioinformatics workflows, Nucleic Acids Research 38 (suppl_2) (2010) W677–W682.

[2] A. Bolt, M. de Leoni, W. M. van der Aalst, Scientific workflows for process mining: Building blocks, scenarios, and implementation, International Journal on Software Tools for Technology Transfer 18 (6) (2016) 607–628.

[3] R. Eshuis, F. Lecue, N. Mehandjiev, Flexible construction of executable service compositions from reusable semantic knowledge, ACM Transactions on the Web 10 (1) (2016) 5.

[4] A. Goderis, U. Sattler, P. Lord, C. Goble, Seven bottlenecks to workflow reuse and repurposing, in: International Semantic Web Conference, 2005, pp. 323–337.

[5] M. Krzywucki, S. Polak, Workflow similarity analysis, Computing and Informatics 30 (4) (2011) 773–791.

[6] B. Cao, J. Yin, S. Deng, D. Wang, Z. Wu, Graph-based workflow recommendation: on improving business process modeling, in: International Conference on Information and Knowledge Management, 2012, pp. 1527–1531.

[7] M. J. Amiri, M. Koupaee, Data-driven business process similarity, IET Software 11 (6) (2017) 309–318.

[8] X. Liu, Y. Ma, G. Huang, J. Zhao, H. Mei, Y. Liu, Data-driven composition for service-oriented situational web applications, IEEE Transactions on Services Computing 8 (1) (2015) 2–16.

[9] Z. Zhou, Z. Cheng, L.-J. Zhang, W. Gaaloul, K. Ning, Scientific workflow clustering and recommendation leveraging layer hierarchical analysis, IEEE Transactions on Services Computing 11 (1) (2018) 169–183.

[10] R. Lapeña, J. Font, C. Cetina, O. Pastor, Model fragment reuse driven by requirements, in: International Conference on Advanced Information Systems Engineering, 2017, pp. 73–80.

[11] R. Sarno, E. W. Pamungkas, D. Sunaryono, et al., Workflow common fragments extraction based on wsdl similarity and graph dependency, in: International Seminar on Intelligent Technology and Its Applications, 2015, pp. 309–314.

[12] C. Zeng, T. Zhang, P. C. Hung, Fast service process fragment indexing and ranking, IEEE Transactions on Services Computing 9 (5) (2016) 672–685.

[13] J. Wen, Z. Zhou, Z. Shi, J. Wang, Y. Duan, Y. Zhang, Crossing scientific workflow fragments discovery through activity abstraction in smart campus, IEEE Access 6 (2018) 40530–40546.

[14] S. Hu, L. Zou, J. X. Yu, H. Wang, D. Zhao, Answering natural language questions by subgraph matching over knowledge graphs, IEEE Transactions on Knowledge and Data Engineering 30 (5) (2018) 824–837.

[15] D. M. Blei, Probabilistic topic models, Communications of the ACM 55 (4) (2012) 77–84.

[16] C. Fellbaum, Wordnet and wordnets, in: Encyclopedia of Language and Linguistics, 2005, pp. 2–665.

[17] X. Yan, J. Guo, Y. Lan, X. Cheng, A biterm topic model for short texts, in: International Conference on World Wide Web, 2013, pp. 1445–1456.

[18] X. Cheng, X. Yan, Y. Lan, J. Guo, Btm: Topic modeling over short texts, IEEE Transactions on Knowledge and Data Engineering 26 (12) (2014) 2928–2941.

[19] A. Asuncion, M. Welling, P. Smyth, Y. W. Teh, On smoothing and inference for topic models, in: Conference on Uncertainty in Artificial Intelligence, 2009, pp. 27–34.

[20] Z. Wang, L. Wen, S. Wang, J. Wang, Similarity measurement for process models based on transition-labeled graph edit distance, Computer Integrated Manufacturing Systems 22 (2) (2016) 343–352.

[21] R. Conforti, M. Dumas, L. García-Bañuelos, M. La Rosa, BPMN miner: automated discovery of BPMN process models with hierarchical structure, Information Systems 56 (2016) 284–303.

[22] H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods, Semantic Web 8 (3) (2017) 489–508.

[23] S. Hong, N. Park, T. Chakraborty, H. Kang, S. Kwon, PAGE: answering graph pattern queries via knowledge graph embedding, in: International Conference on Big Data, 2018, pp. 87–99.

[24] D. Garijo, O. Corcho, Y. Gil, Detecting common scientific workflow fragments using templates and execution provenance, in: International Conference on Knowledge Capture, 2013, pp. 33–40.

[25] A. Mohan, M. Ebrahimi, S. Lu, A folksonomy-based social recommendation system for scientific workflow reuse, in: International Conference on Services Computing, 2015, pp. 704–711.

[26] T. T. Aye, G. K. K. Lee, Y. Su, T. Zhang, C. Lee, H. Kasim, I. Hoe, B. S. Lee, T. G. G. Hung, Layman analytics system: A cloud-enabled system for data analytics workflow recommendation, IEEE Transactions on Automation Science and Engineering 14 (1) (2017) 160–170.

[27] O. Hatzi, G. Batistatos, M. Nikolaidou, D. Anagnostopoulos, A specialized search engine for web service discovery, in: International Conference on Web Services, 2012, pp. 448–455.

[28] T. Louge, M. H. Karray, B. Archimede, Z. Maamar, M. Mrissa, Semantic web services composition in the astrophysics domain: Issues and solutions, Future Generation Computer Systems 90 (2019) 185–197.

[29] X. Xue, H. Hongfang, S. Wang, C. Qin, Computational experiment-based evaluation on context-aware o2o service recommendation, IEEE Transactions on Services Computing 12 (6) (2019) 910–924.

[30] Y. Zhong, Y. Fan, W. Tan, J. Zhang, Web service recommendation with reconstructed profile from mashup descriptions, IEEE Transactions on Automation Science and Engineering 15 (2) (2016) 468–478.

[31] Y. Zhang, G. Tian, H. Chen, Exploring the cognitive process for service task in smart home: A robot service mechanism, Future Generation Computer Systems 102 (2020) 588–602.

[32] A. Soltanian, F. Belqasmi, S. Yangui, M. A. Salahuddin, R. Glitho, H. Elbiaze, A Cloud-Based Architecture for Multimedia Conferencing Service Provisioning, IEEE Access 6 (2018) 9792–9806.

[33] A. Hussain, J. Chun, M. Khan, A novel customer-centric Methodology for Optimal Service Selection (MOSS) in a cloud environment, Future Generation Computer Systems 105 (2020) 562–580.

[34] Y. Wu, C. Yan, Z. Ding, P. Wang, C. Jiang, M. Zhou, A relational taxonomy of services for large scale service repositories, in: International Conference on Web Services, 2012, pp. 644–645.

[35] J. Wang, P. Gao, Y. Ma, K. He, P. C. K. Hung, A web service discovery approach based on common topic groups extraction, IEEE Access 5 (2017) 10193–10208.

[36] Z. Gao, Y. Fan, C. Wu, W. Tan, J. Zhang, Y. Ni, B. Bai, S. Chen, Seco-lda: Mining service co-occurrence topics for composition recommendation, IEEE Transactions on Services Computing 12 (3) (2019) 446–459.

[37] A. R. Baskara, R. Sarno, Web service discovery using combined bi-term topic model and WDAG similarity, in: International Conference on Information & Communication Technology and System, 2017, pp. 235–240.