A Folksonomy-Based Social Recommendation System for Scientific Workflow Reuse

Aravind Mohan, Mahdi Ebrahimi, Shiyong Lu
Wayne State University
Detroit, MI, USA
{amohan, mebrahimi, shiyong}@wayne.edu

Abstract— **In the past decade, scientific workflow systems have significantly improved scientists' ability to structure scientific processes, use computational resources, and analyze their data more efficiently. Such productivity can be further enhanced by sharing, reusing, and repurposing existing tasks and workflows across different users and institutes. However, existing scientific workflow systems are mainly single-user oriented with limited sharing and reusing functionalities. To overcome such limitations, we propose a folksonomy-based social workflow recommendation system to improve workflow design productivity. Our contributions are: i) We developed a web-based workflow design environment (called Webbench) to allow users to create workflows and collaboratively annotate and categorize them using social tags. The resulted folksonomy improves workflow searchability and shareability. ii) We proposed several workflow recommendation strategies to automatically or semi-automatically augment an in-progress workflow, leveraging both structural and semantic similarities between workflows and guiding information extracted from previously created workflows in the database. iii) We implemented the proposed environment and strategies in a prototype based on the DATAVIEW scientific workflow management system and validated our approach with numerous use cases.**

Keywords- Scientific workflows; folksonomy; DATAVIEW

I. INTRODUCTION

The past decade has witnessed the growing benefits of using scientific workflow systems to improve the productivity of designing scientific processes and data-driven scientific discoveries in various domains, such as bioinformatics [9], neuroinformatics [10], ecology [11], oceanography [12], astronomy [13], and high-energy physics [14]. However, the productivity of workflow design is still hampered in existing scientific workflow systems in two ways. Most existing scientific workflow management systems are single-user oriented and thus across-user sharing and reuse cannot be achieved within the system per se. Meanwhile, workflow design is largely still a tedious and error-prone process with little or no automation support. While social scientific workflow sharing environments, such as MyExperiment [15], greatly facilitate workflow sharing and reuse across tools, users, and institutes, such sharing is external to a workflow design environment and thus provides little help to the design of an in-progress workflow.

In the meanwhile, folksonomy, the practice and method of collaboratively creating and reusing tags to annotate and categorize digital contents, has become a key characteristic of Web 2.0 [3]. In contrast to a taxonomy, which has a fixed vocabulary, a folksonomy allows each author or user to create his or her own terms contributing to an evolving folksonomy. Such flexibility greatly improves the productivity of tagging and annotation and engagement of users. As a result, more digital contents are annotated and searchability is improved. Moreover, a folksonomy keeps track of emerging trends in tag usage and user interests. Therefore, it is natural to adopt folksonomies to annotate and categorize scientific workflows.

To overcome the above limitations of existing workflow design and sharing systems, we propose a folksonomy-based social workflow recommendation system to improve workflow design productivity. Our contributions are: i) We developed a web-based workflow design environment (called Webbench) to allow users to create workflows and collaboratively annotate and categorize them using social tags. The resulted folksonomy improves workflow searchability and shareability. ii) We proposed several workflow recommendation strategies to automatically or semi-automatically augment an in-progress workflow, leveraging both structural and semantic similarities between workflows and guiding information extracted from previously created workflows in the database. iii) We implemented the proposed environment and strategies in a prototype based on the DATAVIEW scientific workflow management system and validated our approach with a case study.

The rest of the paper is organized as follows. Section II provides an overview of the workflow recommendation framework. Section III introduces our folksonomy based workflow model. Section IV presents our workflow recommendation algorithms to recommend a suitable workflow to augment an in-progress workflow. Section V illustrates our implementation and experiments with a case study. Section VI and VII present related work and conclusions.

II. AN OVERVIEW OF WORKFLOW RECOMMENDATION FRAMEWORK

In this section, we propose a workflow recommendation framework to improve workflow design productivity by recommending a suitable workflow that is both syntactically and semantically compatible to any incomplete in-progress workflow. In accordance with the reference architecture for scientific workflows [7,16], we propose a workflow design inspector, a syntactic recommender and a semantic recommender as the core components of the workflow recommendation framework, which are positioned in the

978-1-4673-7281-7/15 $31.00 © 2015 IEEE
DOI 10.1109/SCC.2015.100

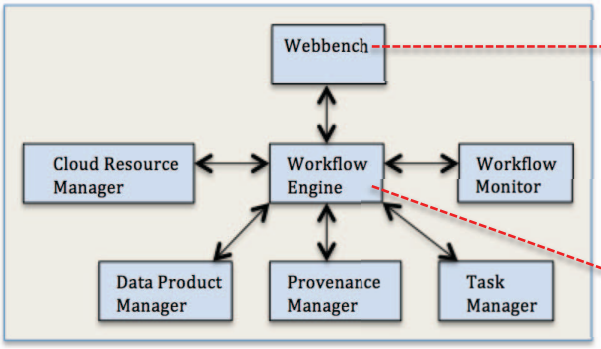

Fig. 1. (a) DATAVIEW architecture.

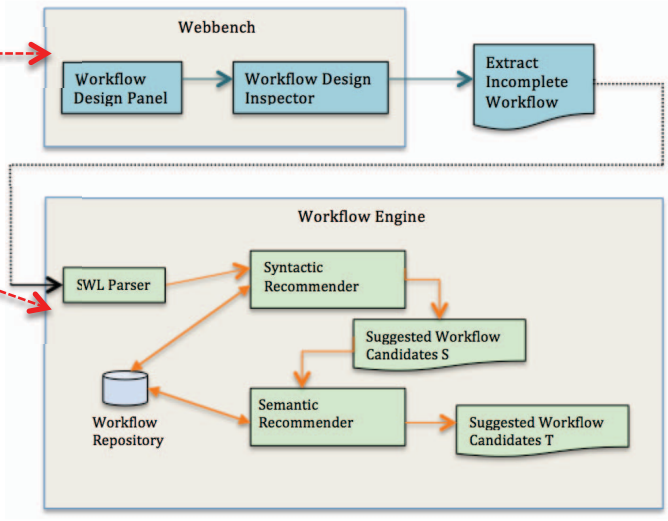

Fig. 1. (b) Workflow recommendation framework.

Webbench and the Workflow Engine, two subsystems in the reference architecture. In Fig. 1a, we show the system architecture for DATAVIEW [16], which is composed of seven loosely coupled subsystems, including the Webbench, the workflow engine, the workflow monitor, the cloud resource manager, the data product manager, the provenance manager, and the task manager. In Fig. 1b, we show an overview of the workflow recommendation framework, in which two recommenders, the syntactic recommender and the semantic recommender, are located in the workflow engine and the workflow design inspector in the Webbench, respectively.

The Webbench in Fig, 1a features an online scientific workflow system that allows data scientists to create, edit and run a visual scientific workflow online. In our DATAVIEW system, we use *mxGraph*, a visualization language program, for representing the workflow design. During workflow design, every time a new workflow is added to the workflow design panel, the workflow design inspector extracts the list of complete and incomplete workflows that exist in the workflow design panel. The workflow design inspector is the key component that drives the workflow recommendation framework. During the workflow design process, the workflow design inspector provides a clickable button called "Recommend Workflow" within the incomplete workflow and sends the specification of the incomplete workflow to the workflow engine.

The workflow engine in Fig. 1b, on the other hand, invokes the SWL Parser to extract the specification of the incomplete workflow that is provided by the workflow design inspector. Specification of the workflow contains logical details, mapping details, and physical details of the workflow. Logical details include the workflow name, the input, and the output ports of the workflow. Mapping details include the mapping information that illustrates how the data product is mapped to the input and the output port of the workflow. Physical details include information such as the location of the code that is embedded inside the workflow. The syntactic recommender component accepts the specification of the incomplete workflow and validates the connectivity on the ports to check whether incompleteness is on either the input side or the output side of the workflow. Incompleteness on an input requires a producer workflow whose output port shall be connected to the input port of the incomplete workflow. Incompleteness on an output port requires a consumer workflow whose input port shall be connected to the output port of the incomplete workflow.

The syntactic recommender then performs a look up in the workflow repository to find the list of workflows that contains an input port that matches the output port of the incomplete workflow. A list of workflows that match with the port information of the incomplete workflow is then added to the suggested workflow candidate list S that contains both the recommended producer and consumer workflows.

Although the list of workflows in the suggested workflow candidate S is syntactically compatible with the incomplete workflow, those workflows might not be semantically relevant to the incomplete workflow nor preferred by the user. Hence, the semantic recommender is used to filter and identify a sublist of workflow candidates from S that are also semantically compatible by leveraging the tag annotations. The semantic recommender component computes a workflow recommendation score based on both workflow similarity score and user interest matching score between the incomplete workflow and each of the workflows in the suggested workflow candidates S. with that of the tags associated with the incomplete workflow. After computing the workflow recommendation score, we rank the workflows based on the recommendation score and a new list of producer and consumer workflows is added to the suggested workflow candidate list T and finally recommended to the data scientist.

III. A FOLKSONOMY BASED WORKFLOW MODEL

Folksonomy [1] is a classification system derived from the practice and method of collaboratively creating and managing tags to annotate and categorize content. As shown in Fig. 2c, in folksonomy, a user tags a resource (e.g. workflow) and the ternary relationship between the user, the tag and the resource is collectively known as tag assignment (TAS).

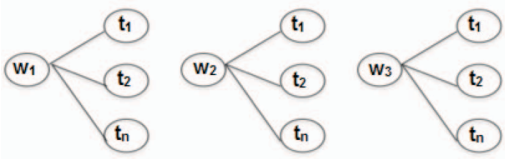

Fig. 2. (a) Workflow tag relevance.

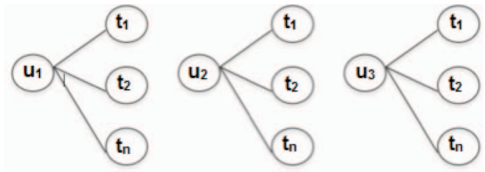

Fig. 2. (b) User tag preference.

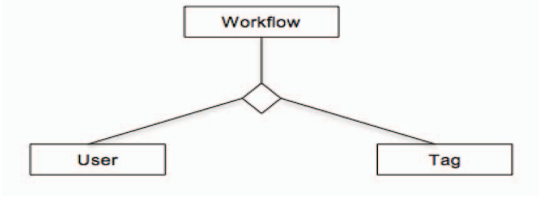

Fig. 2. (c) Tag assignment (ternary relationship).

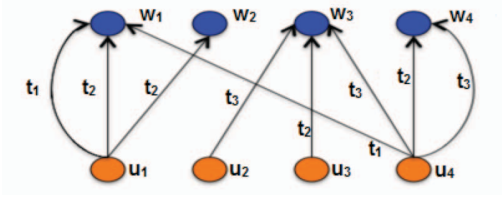

Fig. 2. (d) Folksonomy in scientific workflows.

As shown in Fig. 2d, TAS can be represented as a graph consisting of a set of users, a bag of tags and a set of workflows, thereby forming a folksonomy relationship. TAS emphasizes both user preference and workflow tag relevance factors. In Fig. 2b, we show the user preference, which is used to compute how much a particular user prefers the tag based on the user's previous tagging activities. In Fig. 2a, we show the workflow relevance, which is used to compute how much a particular tag is relevant to the workflow based on the tagging activities on the workflow.

Our previously proposed workflow model [8] contains the syntactic information of the workflow such as name, input port details, output port details and data mapping details. However, it does not include any semantic information about the workflow except the workflow name. In our new model, we support tagging the workflow during the workflow design process. Tags represent lightweight textual information that provides more insights about the workflow and more importantly it is user driven. Hence the semantic information driven by the tags is leveraged to provide support to the user by increasing the user's workflow design productivity.

Definition 1. *A* ***folksonomy*** *is a tuple F = (U, T, W, Y) where U, T, and W are finite sets, whose elements are called users, tags and workflows, respectively. Y is a ternary relation between them, i. e., $Y \subseteq U \times T \times W$, whose elements are called tag assignments. For workflow wf_i, we use tags(wf_i) to denote the bag of tags (duplicates are included) annotated by all users for workflow wf_i. For a user u_m, we use profile(u_m) to represent the bag of tags that a user u_m used to annotate all workflows in the workflow repository.* □

ASSUMPTION 1: given an incomplete workflow wf_i that is being created by a user u_m, the rank of a workflow wf_j in the recommended list of workflows is decided by two scores, $WS(wf_i, wf_j)$, the similarity matching between wf_i and wf_j, and $IM(u_m, wf_j)$, a user profile interest matching between user u_m and workflow wf_j.

A scientific workflow represents a multiple-step data analysis pipeline that chains several data analysis workflows (e.g. Web Services, Command line applications) together via data links, which connect the output of one workflow to the input of another workflow. We have two types of workflow namely, a primitive workflow, that has no subworkflows inside it, a composite workflow, that has at least one or more subworkflows inside it. More formally a scientific workflow is defined as:

Definition 2. *A* ***scientific workflow*** *W is a tuple (u, T, TS, IP, OP) where u is the unique identifier of the user who created W, T is a bag of tags that are assigned to W, TS is the set of constituent subworkflows inside W, IP is the set of input ports of W, and OP is the set of output ports of W. W is primitive if $|TS| = \Phi$ and composite otherwise.* □

The tf-idf score is widely used in the information retrieval community to identify how important a particular word is to a document in the collection. In our workflow model, we refer documents as workflows and terms as tags, and hence, the tf-idf score is used to compute how important a particular tag is in the context of a workflow. The tf-idf score is computed by multiplying term frequency (tf) with inverse document frequency (idf). The term frequency (tf) is used to compute the frequency of the particular tag t_k in a workflow wf_i and is computed by equation (1).

$$\text{tf}(t_k, wf_i) = 0.5 + \frac{0.5 \times f(t_k, wf_i)}{\max\{f(t, wf_i) : t \in wf_i.T\}} \quad (1)$$

where $f(t_k, wf_i)$ is the raw frequerncy of tag t_k in wf_i.T. The inverse document frequency (idf) is used to measure how much common a particular tag t_k is in the N number of workflows in the workflow repository W and is computed by the equation (2):

$$\text{idf}(t_k, W) = \log \frac{|W|}{|\{wf_i \in W : t_k \in wf_i.T\}|} \quad (2)$$

$$\text{tf-idf}(t_k, wf_i, W) = \text{tf}(t_k, wf_i) \times \text{idf}(t_k, W) \quad (3)$$

Given a user u_m and all the users U in the system, let $u_m.T$ be the bag of tags used by user u_m to annoate all the workflows in W, then the tf-idf of a tag t_k with respect to user u_m and U, *td-idf(t_k, u_m, U)*, can be defined similarly.

Definition 3. **Workflow Similarity** *WS* is used to compute the similarity between any two arbitrary workflows. Given two workflows wf_i and wf_j, we represent them as two vectors:

$$v(wf_i) = (w_{i1}, w_{i2}, w_{i3}, \dots, w_{in})$$
$$v(wf_j) = (w_{j1}, w_{j2}, w_{j3}, \dots, w_{jn})$$

where w_{ik} = tf-idf(t_k , wf_i , W), w_{jk} = tf-idf(t_k , wf_j , W) and n is the number of tags in T. The workflow similarity between wf_i and wf_j is computed by equation (4)

$$WS(wf_i, wf_j) = \frac{\sum_{k=1}^{n} w_{ik} \times w_{jk}}{\sqrt{\sum_{k=1}^{n}(w_{ik})^2} \times \sqrt{\sum_{k=1}^{n}(w_{jk})^2}} \quad (4)$$ □

Based on assumption 1, a recommended workflow should be not only similar to the incomplete workflow, but also matching to the interest of the user u_m, which is characerized by her profile to achive personalized recommendation. To this end, we introduce the notion of user interest matching score.

Definition 4. *The* **User Interest Matching Score** *IM* is used to compute the interest matching degree between a user u_m and a workflow wf_j, and is defined as follows:

$$IM(u_m, wf_j) = \frac{\sum_{k=1}^{n} w_{jk} \times u_{mk}}{\sqrt{\sum_{k=1}^{n}(w_{jk})^2} \times \sqrt{\sum_{k=1}^{n}(u_{mk})^2}} \quad (5)$$

where w_{jk} = tf-idf(t_k , wf_j , W), u_{mk} = tf-idf(t_k , u_m , U) and n is the number of tags in T. □

Definition 5. **Workflow Recommendation Score** *WR* is used to rank the workflows that are part of the suggested workflow candidates S provided by the syntactic recommender component by computing the workflow similarity and user profile similarity with respect to the incomplete workflow. (See section 2). Based on the workflow recommendation score computed as equation (6), the list of workflows are added to the suggested workflow candidates T and recommended to the user.

$$WR(u_m, wf_i, wf_j) = \gamma \cdot WS(wf_i, wf_j) + (1-\gamma) \cdot IM(u_m, wf_j) \quad (6)$$

where, γ is the recommendation weight factor (RWF) that is used to balance workflow similarity score and user interest matching score that satisfies $0 \leq \gamma \leq 1$. □

IV. WORKFLOW RECOMMENDATION ALGORITHMS

Our workflow recommendation framework considers both syntactic and semantic information that are part of the workflow in order to recommend a suitable workflow to the incomplete workflow. As part of the workflow design process, we provide the user with the option to annotate the workflow by clicking on the tagging button in the workflow design panel. As shown in Fig. 2d, the following tag assignments are created in our user profile table during the user annotation process.

{(u_1,t_1,w_1), (u_1,t_2,w_1), (u_1,t_2,w_2), (u_2,t_2,w_3), (u_3,t_2,w_3), (u_4,t_3,w_1), (u_4,t_3,w_3), (u_4,t_2,w_4), (u_4,t_3,w_4)}

As evident from the above tag assignments, in our system different users can use the same tags to annotate the same workflow. Also different users can use different tags to annotate the same workflow. Because of these constraints, we allow duplicates to be included and hence represent the tags as bag of words.

The syntactic workflow recommender component is mainly used to compare the input/output ports of the incomplete workflow with the workflows residing in our workflow repository. The incompleteness in the workflow occurs mainly due to the missing connection link from the input ports of the workflow to another producer workflow or output ports of the workflow to another consumer workflow. Producer workflows are those workflows in which one or more of the output ports of the workflow are connected to one or more input ports of the incomplete workflow. Consumer workflows are those workflows in which one or more of the input ports of the workflow are connected to one or more output ports of the incomplete workflow. As shown in Algorithm1, Syntactic workflow recommender component accepts two inputs as, incomplete workflow and the list of workflows in the repository.

In our DATAVIEW system, we use an XML based workflow specification language called *SWL* to represent the meta data information of the workflow. Each workflow in our repository contains a *SWL* associated with it. We implemented *SWL Parser* as part of the workflow engine to parse the specification file of the workflow and get all the input and output ports associated with the workflow. Output of the Algorithm1 generates two sets of recommended workflow lists for producer workflows and consumer workflows. For each logical port in the incomplete workflow, we compare the input ports of the incomplete workflow to find a suitable workflow in the workflow repository that contains at least one matching output port that is type compatible with that of the input port of the incomplete workflow. The workflow is then added to the list of recommended producer workflows. We compare the output ports of the incomplete workflow to find a suitable workflow in the workflow repository that contains at least one matching input port that is type compatible with that of the output port of the incomplete workflow. The identified workflow is then added to the list of recommended consumer workflows. For example, in Fig. 3a we show an in-progress

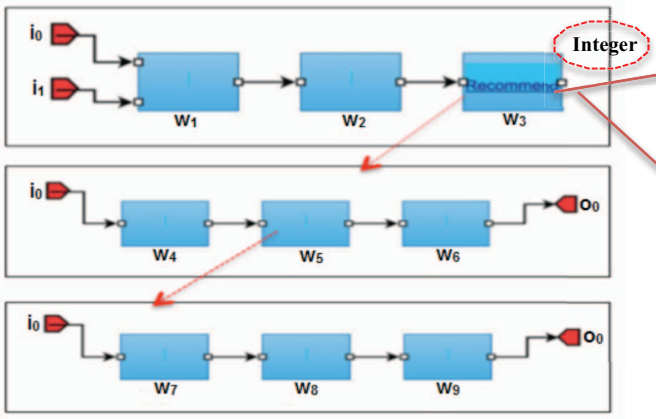

Fig. 3. (a) Sample incomplete composite workflow (W_3).

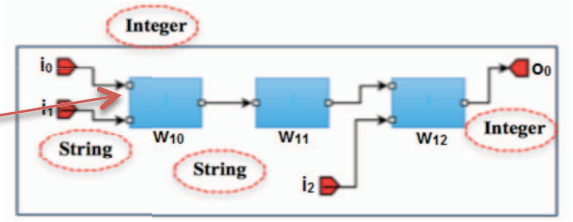

Fig. 3. (b) Sample recommended workflow (wf_1).

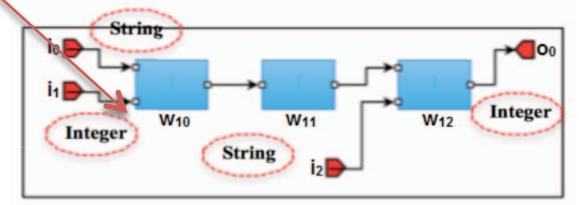

Fig. 3. (c) Sample recommended workflow (wf_2).

workflow that contains an incomplete composite workflow w_3, with the input port connected to another producer workflow w_2. But the output port of the workflow is not connected to any workflow and hence we provide the user with a workflow recommendation link. The output port of the incomplete workflow w_3 is integer type and hence we look at the workflows in the workflow repository and identify those workflows that contain at least one matching port. As shown in Fig. 3b and Fig. 3c, the sample recommended workflows wf_1 and wf_2 contains at least one port of type integer. Hence both wf_1 and wf_2 are added to the list of consumer workflows as part of the recommendation list generated by the syntactic workflow recommender.

Although the workflows wf_1 and wf_2 are syntactically compatible with w_3, the workflows might not be relevant or preferred by the user based on the user's profile. So, we identify the workflow similarity by using Algorithm2 to compute the similarity between the vector representation of the incomplete workflow and the vector representation of the recommended workflows generated as output of the syntactic workflow recommender component. In addition to the workflow similarity, we also compute the user interest matching score and the workflow recommendation score. Based on the recommendation score and a system defined threshold value, the workflow ranking is computed and the list of suggested workflow candidates for both the producer and consumer workflows is generated and provided to the user. We set a system defined threshold value, so that only those workflows that contain the recommendation score greater than the threshold value are added to the recommendation list.

By setting a threshold value, we avoid recommending any workflow that has a low recommendation score to the user. One challenge incurred during our workflow recommendation technique is for folksonomy, both the tags and the workflows repository are growing at a constant rate. To address this issue, we periodically (the first day of each month） use the snapshot of the workflow repository to calculate a new tag vocabulary T and workflow collection D in n dimensions with n = |T|. As shown in Fig. 3a, the workflow w_3 is a composite workflow that contains a subworkflow inside it with the workflows w_4, w_5 and w_6. Further the workflow w_5 is a composite workflow that contains another subworkflow inside it with the workflows w_7, w_8 and w_9. So, when finding a semantically similar workflow, we consider not only the tags associated with the

Algorithm1: Syntactic Recommender

```
1: function syntacticRecommender
2:  input: incomplete workflow wi, list of workflows
       in repository Lw
3: output: list of syntactic recommended workflows
4: listOfProducerWorklows (LPW) ← []
5: listOfConsumerWorklows (LCW) ← []
6: for each logical port p in wi
7   if (p ∈ IP ∈ wi)
8:    for each workflow w ∈ Lw
9:      if (p is type-compatible with at least one
        logical port p* ∈ OP ∈ w)
10:       LPW ← LPW + w
11:     end if
12:   end for
13:  end if
14:  if (p ∈ OP ∈ wi)
15:   for each workflow w ∈ Lw
16:     if (p is type-compatible with at least one
        logical port p* ∈ IP ∈ w)
17:       LCW ← LCW + w
18:     end if
19:   end for
20:  end if
21: end for
22: return LPW, LCW
23: end function
```

incomplete workflow, but also all the subworkflows inside the incomplete workflow (recursively). Let us suppose the following tag assignments are done to the workflows (w_3, w_5, w_7, w_9, wf_1, wf_2) as shown in Fig. 3a, 3b, and 3c.

TABLE 1
Tag Assignment (TAS) Table.

TAS	tf	idf	tf-idf
(u_1, w_3, t_1)	1	0.54	0.54
(u_1, w_3, t_5)	1	0.84	0.84
(u_2, w_5, t_2)	0.75	0.28	0.21
(u_2, w_5, t_1)	1	0.54	0.54
(u_3, w_7, t_3)	0.75	0.84	0.63
(u_4, w_9, t_5)	1	0.84	0.84
(u_2, wf_1, t_1)	1	0.54	0.54
(u_3, wf_1, t_2)	1	0.28	0.28
(u_4, wf_1, t_6)	1	1.14	1.14
(u_2, wf_1, t_6)	1	1.14	1.14
(u_1, wf_2, t_1)	1	0.54	0.54
(u_1, wf_2, t_2)	1	0.28	0.28
(u_3, wf_2, t_3)	1	0.84	0.84
(u_1, wf_2, t_4)	1	1.14	1.14

As shown in TABLE 1, we compute the tf, the idf and the tf-idf value for all the tag assignments. Then, we translate the incomplete workflow and the workflows in the suggested workflow candidates S into the corresponding vector representation. For example, the vector representation of the incomplete workflow w_3 and the recommended workflow candidates wf_1 and wf_2 are:

Vector (w_3) = (0.54,0.84,0.21,0.54,0.63,0.84)

Vector (wf_1) = (0.54, 0.28,1.14,1.14)

Vector (wf_2) = (0.54,0.28,0.84,1.14)

We collect all the user profiles that contain all the tags annotated by the user and translate the user profiles into the corresponding vector representations. For example, the vector representation of the user u_1 (w_3, wf_2), u_2 (w_5, wf_1), u_3 (w_7, wf_2), u_4 (w_9) are:

Vector (u_1) = (0.54,0.84,0.21,0.54,0.63,0.84,0.54, 0.28,0.84,1.14)

Vector (u_2) = (0.21,0.54,0.63,0.84,0.54,0.28,1.14,1.14)

Vector (u_3) = (0.63,0.54,0.28,0.84,1.14)

Vector (u_4) = (0.84)

The workflow similarity between the incomplete workflow w_3 and the workflow candidate's wf_1 and wf_2 is computed by using the corresponding vector as:

WS (w_3, wf_1) = 0.514

WS (w_3, wf_2) = 0.548

The user interest matching score between the user designing the workflow (u_1) and the workflow candidate's is computed by using the corresponding vector as:

IM (u_1, wf_1) = 0.366

IM (u_1, wf_2) = 0.390

Intuitively, it is evident that, the workflow wf_2 is relevant to the incomplete workflow w_3 and preferred by the user u_1 than the workflow wf_1. The final recommendation score validates that the consumer workflow candidate wf_2 (0.47) is higher than the workflow candidate wf_1 (0.44) and the threshold value is (0.45). So, we recommend the user, wf_2 as the consumer workflow to be connected to the output port of the incomplete workflow w_3.

Algorithm2: Semantic Recommender

```
1: function semanticRecommender
2: input: incomplete workflow wf_I, user u_m,
   list of recommended producer workflows Lrpw, list of
   recommended consumer workflows Lrcw,
   recommendation weight factor γ, threshold value T
3: output: list of semantic recommended workflows
4  listOfProducerWorklows (LPW) ← []
5: listOfConsumerWorklows (LCW) ← []
6: LPW_score=0
7: LCW_score=0
8: for each workflow wf_j ∈ Lrpw
9:   LPW_score = γ · WS(wf_i , wf_j ) + (1 − γ) ·
                 IM(u_m, wf_j)
10:  if (LPW_score > T)
11:    LPW ← LPW + w
12:  end if
13: end for
14: for each workflow wf_j ∈ Lrcw
15:  LCW_score = γ · WS(wf_i , wf_j ) + (1 − γ) ·
                 IM(u_m, wf_j)
16:  if (LCW_score > T)
17:    LCW ← LCW + w
18:  end if
19: end for
20: return LPW, LCW
21: end function
```

V. Implementation and Case Study

We implemented the proposed folksonomy based social recommendation framework as a Web-based application called DATAVIEW, written in Java. As part of our implementation, we deployed our DATAVIEW in Futuregrid's Openstack platform. We validated our proposed *Syntactic Recommender* and *Semantic Recommender* algorithms by designing a workflow downloaded from the myExperiment website.

A. myExperiment Data Set

The study focuses on workflows designed in Taverna, an open source popular workflow system. We downloaded the workflows, the input and output port details, the number of workflow instances, the user profile information and the tag assignments available in myExperiment workflow repository. The myExperiment website allows its users to share the workflows from several domains. Based on our analysis on the myExperiment dataset, as shown in Fig. 5a, we found that there are 9886 users, 3542 workflows, 2664 publicly available workflows and 9624 tag assignments in the myExperiment website.

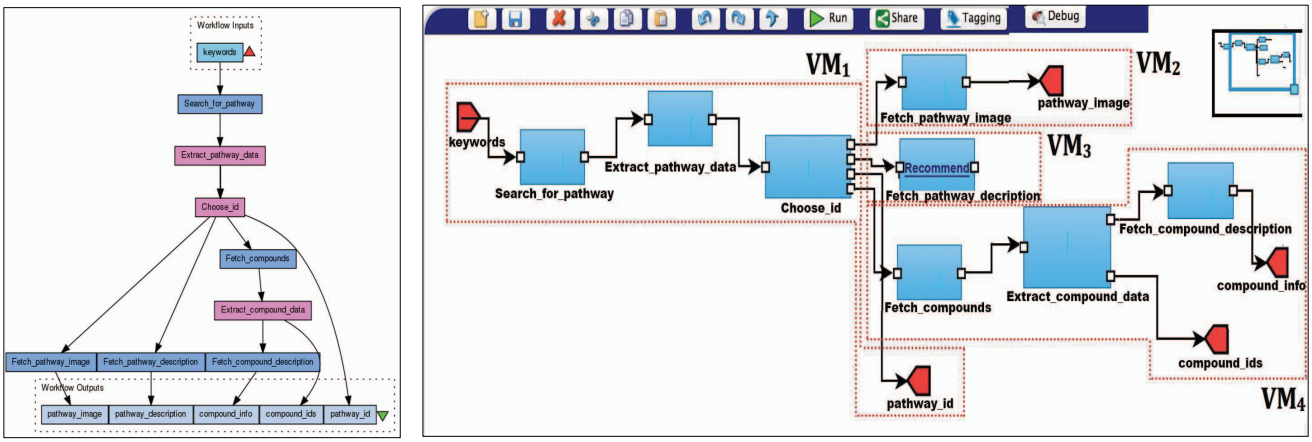

Fig. 4. (a) Taverna workflow in myExperiment; (b) Scientific workflow analyzing metabolite pathway.

B. Case Study: Analyzing Metabolite Pathway

We developed a scientific workflow analyzing metabolite pathway. As shown in Fig. 4b, we designed the workflow based on the taverna workflow downloaded from the myExperiment website (see Fig. 4a) using our data collection technique. Our workflow takes as input, the search keyword and then searches for metabolomic pathways that match the entered keywords and returns information about the chosen pathway. Although there are different ways of parallelizing scientific workflow [17], in our system, we parallelized the execution of the scientific workflow based on the number of workflow fragments in the workflow specification. A workflow fragment is a sequence of workflows that contains a source, a destination and a data channel representing the connectivity (data flow) between the workflows. We identified all the independent workflow fragments in the workflow specification and captured the workflows that are part of those workflow fragments. Then, we executed each workflow fragment separately by running them in different virtual machines.

As shown in Fig. 4b, our workflow contains eight primitive workflows and is deployed using four virtual machines. The input data set, keyword of data type String is sent to the virtual machine VM_1, data is processed using the Search_for_pathway, Extract_pathway_data and Choose_id workflows and the output data set generated is sent to VM_2, VM_3 and VM_4 respectively. In virtual machine VM_2, the input data is processed by Fetch_pathway_image workflow and generates an image output of type file. In virtual machine VM_3, the input data is processed by an incomplete Fetch_pathway_description workflow, whose output port is of type string and is not connected to any consumer workflow or output data stub. In virtual machine VM_4, the input data is processed by Fetch_compounds, Extract_compound_data and Fetch_compound_description workflows and generates two outputs, compound_ids of type list<Integer>, compound_infos of type list<String>. The in-progress workflow contains one incomplete workflow and we provided our users with a recommendation link. When our users, request for recommendation by clicking on the link, we recommended a list of suitable consumer workflows that shall be connected to the output port of the *Fetch_pathway_description* workflow. First, we identified the list of syntactically compatible workflows by executing our syntactic workflow recommender to get a list of workflows that contains at least one input port of type string and formulate the list of syntactically compatible workflows. Second, we identified the list of semantically compatible workflows by filtering the list generated in the previous step and executed our semantic workflow recommender to formulate the list of workflows that are both relevant to the incomplete workflow and also preferred by the user who designed the in-progress workflow. In Fig. 5b, we show the recommendation scores of the top 10 recommended workflows that are suitable to be connected to the output port of the incomplete *Fetch_pathway_description* workflow. In our experiment setting, we set the threshold value to be 0.5 and the recommendation weight factor γ = 0.5.

VI. RELATED WORK

The notion of artifact reuse and recommendation is well studied in the software engineering field. The folksonomy based approach, in which the system provides the users with the ability to publish and categorize various resources such as (web pages, photos, videos, documents, etc.) online with "social annotations" or "tags". Xu et al. [1] explore three properties of folksonomy, namely the categorization, the

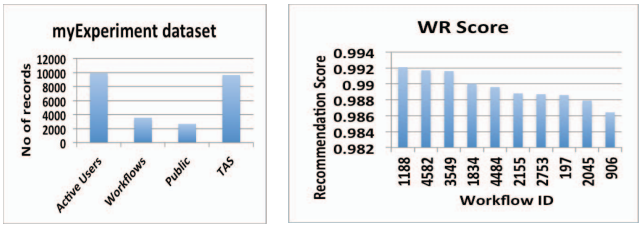

Fig. 5. (a) myExperiment dataset; (b) Workflow recommendation. score for top 10 recommended workflows.

keyword and the structure property and propose a personalized search framework utilizing the folksonomy. In the information retrieval community, Hotho et al. [2] propose a formal model and a new search algorithm for folksonomies. Bao et al. [3] explore the social annotations to optimize web search. In the paper, the author propose two novel algorithms called SocialSimRank and SocialPageRank to measure the similarity and popularity of web pages from web users' perspective.

In the scientific workflow community, building visualization and workflow pipelines is a large hurdle from the users' perspective. It is time-consuming to identify a reusable workflow by manually scanning through more than hundreds of workflows in the workflow repository. Koop et al. [4] propose VisComplete, an auto-complete suggestion technique to help users construct pipelines in the VisTrails system. In the paper, the authors propose a technique to find the syntactic similarity between the incomplete workflow and the workflows in the workflow repository by sub graph matching. Chinthaka et al. [5] propose a case-based reasoning approach to assist composition of workflows using the Lesk algorithm to perform the keyword matching between the input and output of the incomplete workflow and the workflows in the workflow repository. Zhang et al. [6] propose an approach to recommend services in the workflow composition process. In the paper, the author models existing scientific artifacts, services and workflows as a PSW network and recommends services based on the service usage history.

None of the above techniques address the workflow reuse problem using the syntactic and semantic information available in the workflow specification file. Further the above techniques do not address the scientific workflow reuse in the granularity of recommending the producer and consumer workflows based on the input and output port type matching. Our technique validates the port types to recommend syntactically compatible workflows. Next, we leverage the user profile and the tags associated with the incomplete workflow and the workflows residing in the workflow repository to recommend a suitable workflow that is both relevant and preferred by the user to be a producer or consumer of the incomplete workflow.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we first developed a workflow recommendation framework to recommend the list of syntactically and semantically compatible workflow candidates and thereby improve the scientific workflow design productivity of the user. Second, we presented a folksonomy based workflow model to extend our previously proposed primitive workflow model that emphasizes on the semantic information in a workflow. Third, we proposed two workflow recommendation algorithms, to capture the social annotations' capability on syntactic and semantic workflow recommendation respectively. Finally, we implemented the proposed environment and strategies in our DATAVIEW system and validated our approach with a case study and experimental results. Ongoing work includes extending our recommendation framework to support a proactive, system driven recommendation approach to provide recommendations for all the incomplete workflows in an in-progress workflow.

ACKNOWLEDGMENT

This work is supported by U.S. National Science Foundation under ACI-1443069 and is based upon work supported in part by the National Science Foundation under Grant No. 0910812.

REFERENCES

[1] S. Xu, et al., "Exploring folksonomy for personalized search," in Proc. of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'08), pp. 155-162.

[2] A. Hotho, et al., "Information retrieval in folksonomies: search and ranking," in Proc. of the 3rd European conference on The Semantic Web: research and applications (ESWC'06), Springer-Verla. pp. 411-426.

[3] S. Bao, et al., "Optimizing web search using social annotations," in Proc. of WWW '07, pp. 501–510.

[4] D. Koop, et al., "VisComplete: Automating Suggestions for Visualization Pipelines," IEEE Transactions on Visualization and Computer Graphics, vol.14, no. 6, pp. 1691, 2008.

[5] E. Chinthaka, et al., "CBR Based Workflow Composition Assistant," in Proc. of 2009 World Congress on Services (SERVICES-I), IEEE Computer Society. pp. 352-355.

[6] J. Zhang, et al., "Recommend-As-You-Go: A Novel Approach Supporting Services-Oriented Scientific Workflow Reuse," in Proc. of the 2011 IEEE International Conference on Services Computing (SCC'11), pp. 48-55.

[7] C. Lin, et al., "A Reference Architecture for Scientific Workflow Management Systems and the VIEW SOA Solution," IEEE Transactions on Services Computing, vol.2, no. 1, pp. 77-92, 2009.

[8] A. Mohan, et al., "Addressing the Shimming Problem in Big Data Scientific Workflows," in Proc. of the 2014 IEEE International Conference on Services Computing (SCC'14), pp. 347-354.

[9] J. Li, et al., "A bioinformatics workflow for variant peptide detection in shotgun proteomics," Molecular & Cellular Proteomics 10.5 (2011): M110-006536.

[10] K. Fissell. "Workflow-based approaches to neuroimaging analysis," Neuroinformatics. Humana Press, pp. 235-266, 2007.

[11] W. Michener, et al., "Data integration and workflow solutions for ecology," Data integration in the life sciences. Springer Berlin Heidelberg, pp. 321-324, 2005.

[12] R.S. Barga, et al., "Trident: Scientific Workflow Workbench for Oceanography," SERVICES I, pp. 465-466, 2008.

[13] G. Singh, et al., "Workflow task clustering for best effort systems with Pegasus," in Proc.of the 15th ACM Mardi Gras conferences (MG'08), pp. 235-266.

[14] A. Dolgert, et al., "Provenance in high-energy physics workflows," Computing in Science & Engineering, vol.10, no. 3, pp. 22-29, 2008.

[15] C. A. Goble, et al., "myExperiment: a repository and social network for the sharing of bioinformatics workflows," Nucleic acids research, 38(suppl 2), W667-W682.

[16] A. Kashlev, et al., "A System Architecture for Running Big Data Workflows in the Cloud," in Proc. of the 2014 IEEE International Conference on Services Computing (SCC'14), pp. 51-58.

[17] E. Deelman, et al., "Pegasus: A framework for mapping complex scientific workflows onto distributed systems," Scientific Programming Journal, vol.13, no. 3, pp. 219-237, 2005.