SCIENTIFIC WORKFLOW CLUSTERING BASED ON MOTIF DISCOVERY

Tahereh Koohi-Var[1] and Morteza Zahedi[2]

[1]International Campus of Kharazmi, Shahrood University of Technology,
Shahrood, Iran
[2]CE Department, Shahrood University of Technology,
Shahrood, Iran

ABSTRACT

In this paper, clustering of scientific workflows is investigated. It proposes a work to encode workflows through workflow representations as sets of embedded workflows. Then, it embeds extracted workflow motifs in sets of workflows. By motifs, common patterns of workflow steps and relationships are replaced with indices. Motifs are defined as small functional units that occur much more frequently than expected. They can show hidden relationships, and they keep as much underlying information as possible. In order to have a good estimate on distances between observed workflows, this work proposes the scientific workflow clustering problem with exploiting set descriptors, instead of vector based descriptors. It uses k-means algorithm as a popular clustering algorithm for workflow clustering. However, one of the biggest limitations of the k-means algorithm is the requirement of the number of clusters, K, to be specified before the algorithm is applied. To address this problem it proposes a method based on the SFLA. The simulation results show that the proposed method is better than PSO and GA algorithms in the K selection.

KEYWORDS

Scientific Workflow Clustering, K-Means, Workflow Motif, SFLA

1. INTRODUCTION

With ever increasing number of scientific workflow repositories, organizing and categorizing these workflows to diverse need of user by manual means is a complicated job. Hence a machine learning technique named clustering is very useful. The ability to group similar workflows together has many important applications. For example, clustering algorithms can be applied for regrouping similar workflows for their simultaneous execution.

Scientific workflows are complex objects that allow users to specify multi-step computational tasks. A scientific workflow describes dependencies between tasks as can be seen in fig. 1. It models structural flow of a process and manages data flow. Most processes can be modeled with workflows. A process can be consists of repetitive patterns of sub-processes in a special network. The structure of this network has control and data dependencies [11]. Because of the importance of workflows as sequences of multiple related tasks, improving efficiency of them is important. The improvement requires testing and validating new statistical methods and software. One of the most important work about scientific workflows is clustering of similar sub-workflows. It is crucial to find generic, recurring workflows, automatically. Finding generic, recurring workflows is central to many tasks scheduling problems [1]. Workflow clustering can also be used for providing better organization for search results [2].

DOI : 10.5121/ijcseit.2017.7401

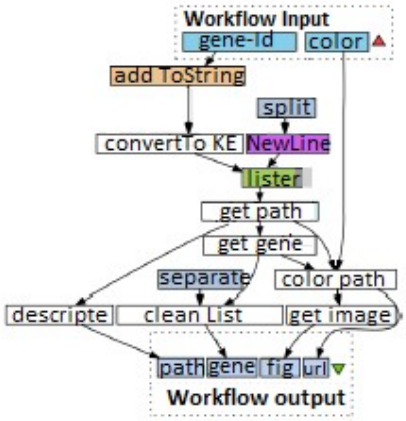

Fig. 1. Scientific workflow sample

This work is concerned with the problem of clustering multiple scientific workflows. A number of recent studies have addressed the problem of workflow clustering [1, 4]. Workflow clustering approaches can be broadly categorized as text based and structure based approaches. In text-based approaches, string distance measures, such as the Hamming or Levenshtein distances, are applied to assess dissimilarities between workflows [1]. Text based methods rely on the text mining of workflow textual description and the use of keyword similarity measures. In structural scope, some papers use workflow encoding for workflow indexing to further reduce runtime complexity and scale similarity search to sizes of current repositories. For example, some of them are based on measuring process similarities based on change operations [6]. In structure based approaches workflow dissimilarities depends on workflow task relations. However, sometimes one maybe cannot reach to effective results in structure based workflow clustering. For example, in paper [1] data encoding is only advantageous about task existence and task occurrence of the workflow, not about task relations. So, to discuss this problem, this work embeds workflows in a Deep Neural Network (DNN) similar the work in [17]. It maps textual descriptions of keywords to weighted distributed vectors. If a vector representation is considered, similarity metrics such as the Cosine, Euclidean or squared Euclidean distances can be employed to estimate distances between observed workflows [1]. The main challenge in this case is the representation of data imperfection. In this case, sets are rights descriptors; a natural way to handle the variable number of elements in a workflow sequence is to describe the workflow system state as finite sets instead of conventional vectors. Hence, the workflow clustering problem is addressed with exploiting set descriptors. Note that the clustering is done on the entire workflows without considering the sub-clustering of individual workflow tasks. In spite of some other works similar the work of [22]. In the work of [22] normalization with regard to sizes of compared workflows is done as a post processing step.

Recently a set based clustering method with one of the most common iterative algorithm, k-means, has been introduced [9]. In this method, if number of clusters to form is k, it finds all the required partitions (k) at a time. The evaluation method of it is based on Optimal Sub-Pattern Assignment (OSPA) Barycenter metric. However, one of its drawbacks is the need for the number of clusters, k, to be specified before the algorithm is applied.

In this paper the proposed method is inspired from the set based clustering method [9]. It uses the centroid to cluster workflows. The centroid is a representative workflow of a cluster which is calculated as a mean of workflow sets. Besides, it proposes a method based on *k-means* clustering

operation to select the number of clusters, k. The method employs an objective evaluation function to suggest suitable values for k, thus avoiding the need for trial and error. For suggesting the best k values, this work compares some recently popular population-based cooperative search algorithms based on the analogy of Darwinian evolution. The methods are Shuffled Frog Leaping Algorithm (SFLA) [10], Particle Swarm Optimization (PSO) [15] and Genetic Algorithm (GA) [16]. The SFLA is a mimetic meta-heuristic algorithm. It has been proposed as an efficient tool for solving complex nonlinear optimization problems. The SFLA is a population-based cooperative search. It is inspired by mental content theory [10]. Diverse fields have used it. GA is an adaptive stochastic algorithm based on natural selection and genetics [24]. The SFLA, GA, and PSO algorithms have shown good results in many studies [24]. So, this work has a comparison based on these methods to attain good results.

Workflow representation in this paper is based on workflow motifs as sub-workflows. Motifs are "patterns of interconnections occurring in complex networks" [7]. A considerable amount of effort has been dedicated to automatic meaningful motifs discovery in sets of graphs [7, 18]. The proposed method of this paper extracts workflow motifs. Then it embeds motifs in workflow sets. Finally, with respect to similarity of scientific workflows in workflow context, clustering is done.

Steps of the proposed work are shown in fig. 2. The fig. 2 shows the framework that implements steps for clustering. In first step the scientific workflow is imported. Then the structural and textual data is handled by the framework. In workflow level the proposed method does the pre-processing. Then in motif level sub-workflows are mapped to motifs in workflows. The mapping step is based on DNN. Other processes are done in workflow levels to obtain the clusters.

For simulation results the work extracted information from a resource holding over 69663 experimental design workflows (ArrayExpress) [14]. It used a subset of this collection, comprising 120 scientific workflows. This subset of workflows, available in the ISA-Tab format [13], offers a good representation of experimental typology.

The main contributions of the proposed method are as follows:

(1) Define a new workflow representation to improve workflow clustering.
(2) Define a method for a set based workflow clustering.
(3) Adapt SFLA method to improve the proposed method of workflow clustering.
(4) Define heuristic method in fitness function for SFLA to map it to our problem.

The rest of the paper is organized as follows: Section II deals with workflow representation. Section III introduces the finite set based workflow clustering method. Section IV presents an approach to find the number of clusters, k. Section V presents comparison of simulation results of the proposed workflow clustering method. Finally, section VI concludes the paper.

2. WORKFLOW REPRESENTATION

Workflow is a depiction of a sequence of process tasks with their specified structural dependencies. In workflows context each component can be seen as a black box, only exposing a limited set of information for assessment of its functionality. As an insight to workflows, a number of workflows are represented as multiple black boxes in Fig. 3.

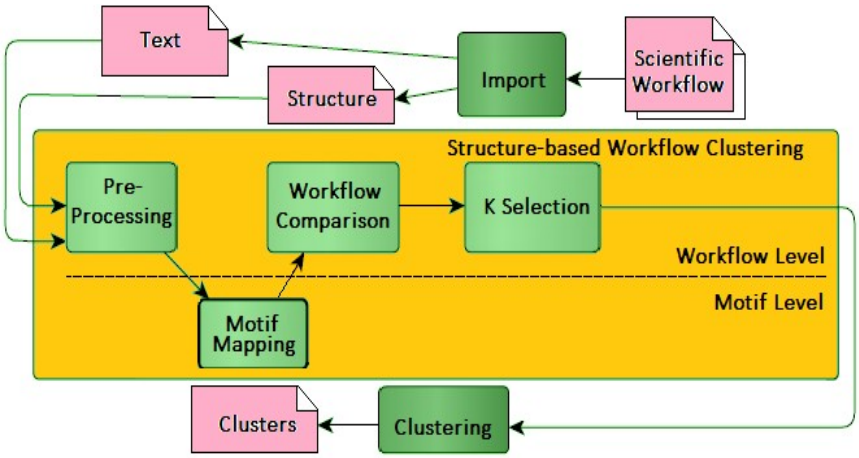

Fig. 2. Scientific workflow clustering framework

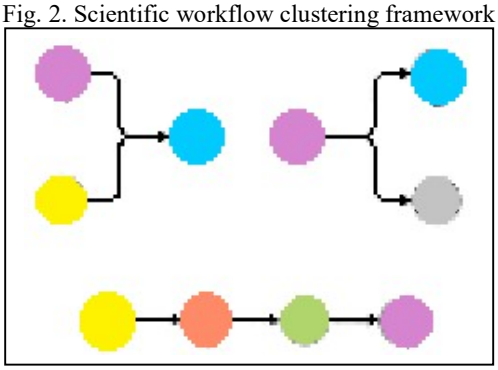

Fig. 3. Workflow samples

Functionality of a scientific workflow is determined by components it is composed of, and how these components are connected by linkages. While we are ultimately interested in comparing and clustering whole workflows, each component represents a distinct functional entity in its own right. How to best identify and compare entities between workflows given this information is still an open problem. This section tries to give a good solution to the representation of workflows as a pre-processing step before workflow clustering.

Clustering can be applied in the context of workflows to derive workflow specifications from sequences of process operations. Further cluster analysis as an established method allows discovering the structure in collections of data by exploring similarities between data points. By cluster analysis one can group data objects in such a way that data objects within a cluster are similar, while data objects of different clusters are dissimilar to one another [5].

So far, several solutions have focused to the problem of workflow clustering methods. Several works have addressed the control of task granularity of bags of tasks. For instance, Muthuvelu et al. [3] proposed a clustering algorithm that groups bags of tasks based on task runtime are grouped up to the resource capacity. Later, they extended their work [3] to find task granularity based on task file size, CPU time, and resource constraints.

Generation of workflow clusters can be categorized either into text-based approaches or into structure-based approaches. Text-based methods rely on looking for patterns of workflow textual descriptions and the use of keyword similarity measures. Structure-based workflow clustering methods usually have higher algorithmic complexities and pose severe challenges. Encoding workflow sequences can reduce complexities. Workflows can encoded into binary vector representations, where each available workflow task (i.e. method, element or activity) is either present (1) or absent (0). Another way of representing a workflow is as a multidimensional vector [2]. If a vector representation is considered, similarity metrics such as cosine, Euclidean or squared Euclidean distances can be employed to estimate distances between observed workflows [1]. However, using only the presence-absence data in the workflow representation discards structural information characterizing the data flow. To circumvent this representation bias, one can apply a multiple vector encoding strategy, such as a transition vector or process vector encoding [1]. This representation may seem not very suitable for workflows because the structural information is completely lost. In paper [1] as a solution, the tasks are used as the description of the smallest encoding unit in a workflow clustering. Available tasks are encoded as vectors with four general workflow encoding types. The first type of workflow encoding is the data presentation in the form of a binary matrix accounting for the presence and absence of the available tasks. The workflow encoding of Type II is based on the tasks occurrence information. The workflow encoding type III preserves the essential structural information as a pair of tasks representation without carrying out lengthy graph theory methods for determining the distance matrix between workflows. Finally, addition of input and output port information to the pair-of-tasks matrix is considered, as workflow type IV. Both encodings of Types I and II, based on the presence-absence and occurrence information, generally outperformed more advanced encodings of Types III and IV, taking into account structural workflow information and formats of input and output ports. This is mainly due to a greater sparseness of data corresponding to encodings of Types III and IV. To address this problem this paper uses a scalable method based on learning 'workflow embeddings' using light-weight tree-structured neural language models, inspired from earlier motif discovery methods similar the work of [17]. Tasks in paper [1] are used as descriptions of the smallest encoding units in a workflow clustering. This work to represent workflow sets, defines workflows as sets of tasks. Then inspired from DNN based motif discovery proposed in [17] it encodes workflow tasks as recurring steps in the context of similar processes. The method specifies a hidden predicate to train neural embedding. It offers a unique representation of workflows in the context of similar processes.

To have a good workflow representation it can be formulated in two sub steps:

A. Quantifying workflow motifs
B. Workflow encoding

This section deals with these two sub steps as follows:

A. QUANTIFYING WORKFLOW MOTIFS

In distributed representation, workflows are mapped from workflow structures to vectors space. To measure the similarity of workflow motifs, they can be quantified. Motifs are defined as small functional units that occur significantly more frequently than expected.

Quantifying workflow motifs can be done in two steps. The first step is to identify sets of motifs. To identify sets of motifs, this section distinguishes motifs by the number of input tasks and output

tasks, and the density of workflow motifs. Identifying motifs of workflow can be done by the vector (1). This section extracts common workflow or sub-workflow structures. Motif detection is done based on a fast parallel method proposed in [20]. For the next step suppose that workflow motif observations are as follows:

$$U = (u_1^i, u_2^i, u_3^i, u_4^i),\ \text{i=1,...,n} \tag{1}$$

where $(u_1^i, u_2^i, u_3^i, u_4^i)$ is a random vector of observations from the number of input tasks and output tasks, the textual context of sub-workflow and the density of workflow motifs. The density of workflow motif can be computed by the formula (2).

$$\text{Number of edges of graph/ max number of edges} \tag{2}$$

The similarity measure for workflow elements compares motifs by the means of their sets of observation in workflow.

The challenge in generalizing the workflow clustering algorithm to the motif-motif comparison problem arises because one does not have a fixed set of workflow motifs. Instead, one can have an infinite set of workflow motifs. To solve this problem this work utilizes Gaussian Copula Probability Density Function (PDF) [23].

All the parameters are known:

The *Gaussian Copula* PDF plays an important role in the fields of non-independent variables. Specifically, from Sklar's theorem [12] the Gauss copula is

$$C_p(u_1,...,u_d) = \Phi_p(\Phi^{-1}(u_1),...,\Phi^{-1}(u_d)),$$

where Φ denotes the standard normal distribution function, and Φ_p denotes the multivariate standard normal distribution function with correlation matrix P. So, the Gauss copula is simply a standard multivariate normal distribution where the probability integral transform is applied to each margin. In order to solve Gaussian Copula, this work computes the empirical copula of observations.

B. WORKFLOW ENCODING

Let us denote by y = f (u; w) a generic DNN, taking a vector $u \in \mathbb{R}^d$ as input, and returning a vector $y \in \mathbb{R}^o$ after propagating it through H hidden layers. The vector $w \in \mathbb{R}^Q$ is used as shorthand for the column-vector concatenation of all adaptable parameters of the network. The generic kth hidden layer, $1 \le k \le H+1$, operates on a L_k -dimensional input vector h_k and returns an L_{k+1} -dimensional output vector h_{k+1} as equation (3).

$$h_{k+1} = g_k(w_k h_k + b_k),, \tag{3}$$

where $\{w_k, b_k\}$ are the adaptable parameters of the layer, while $g_k(.)$ is a properly chosen activation function to be applied element-wise. By convention we have $h_1 = u$. For training the weights of the network, consider a generic training set of N examples given by (1). The network is trained by minimizing a standard regularized cost function:

$$w^* = \arg\min_w \{\frac{1}{N} L(f(u_d^i)) + \lambda R(w)\}, \quad (4)$$

where $L(.)$ is a proper cost function, $R(\cdot)$ is used to impose regularization, and the scalar coefficient $\lambda \in \mathbb{R}^+$ weights the two terms. Regularization is done to prevent over-fitting. Standard choices for $L(.)$ are the squared error for regression problems, and the cross-entropy loss for classification problems [19]. By far the most common choice for network regularization is to impose a squared `2 norm constraint on the weights [19]:

$$R_{\ell_2}(w) \triangleq \|w\|_2^2 \quad (5)$$

The advantage of vector based workflow representation enables utilizing common vector based computations. However, it assumes workflow data are presented in a vector representation. The vector based computations constrains the clustering to simplest assumptions of workflows instead of more realistic assumptions. So, this work addresses the problem of constraints computation with the aim of correctly clustering of workflow sets. Finally, with respect to the identified similar workflow motifs the similarity is judged.

3. FINITE SET BASED WORKFLOW CLUSTERING

This section is about the finite set based workflow clustering. The system overview is as follows:

1- Load files that have handler (e.g., ISA-Tab in this work case) into a graph dataset.
2- Analysis of all sets based on DNN
3- Cluster weighted workflow sets with *k-means*

The system has to be able to capture descriptions of complex scientific workflows. So, it uses ISA-Tab handler that allows the linkage of a single sample to multiple analyses employing various assays [21].

Workflows can be loaded in a graph form of tasks prior to the application of clustering algorithms. Let T be the *finite set* of tasks t_i ($1 \le i \le n$), and E be the set of directed arcs, of the *form* (t_i, t_j), where t_i is *called* a *parent task* of t_j. (6)

In addition, a vector of task weights can be provided to characterize the workflow tasks. The variable weights are often used to indicate the importance of some variables or to reduce the data dimension [1]. For example, the weights can be considered to account for inverse term-frequencies when clustering textual data [1]. In this study the weights are only subject to the non-negativity constraint.

The algorithm of workflow clustering results from the standard *k-means* algorithm with systematic replacement of the squared error with the OSPA distance. Based on proposed method in paper [9] the set base *k-means* clustering steps are as follow:

1) Cluster assignment step: Calculates the cluster assignments by assigning each observation to its nearest cluster center with respect to the OSPA distance.
2) Cluster set-centroid update step: Based on the updated clusters, new cluster centroids are calculated using the OSPA barycenter [9].

For each $p \geq 1$, $c > 0$ and $X = \{x_1, ..., x_m\}$, $Y = \{y_1, ...y_n\}$, OSPA is defined in three distinct states. If $m = n = 0$, then $\bar{d}_{c,p}(X,Y) = \bar{d}_{c,p}(Y,X) = 0$. Otherwise if $m \leq n$ then the result is equation (7).

$$d_{c,p}(X,Y) \square \frac{1}{n}\left(\min_{\pi \in \Pi_n} \sum_{i=1}^{m} d_c(x_i, y_{\pi(i)})^p + c^p(n-m)\right)^{\frac{1}{p}} \tag{7}$$

where Π_n is permutation set on $\{1,2,...,n\}$ in state space.

If $m > n$, then $\bar{d}_{c,p}(X,Y) = \bar{d}_{c,p}(Y,X) = 0$. Two parameters of OSPA metric are p order and c cut-off; the order of p shows noise sensitiveness, while the cut-off c indicates relative weighting of penalties assigned to cardinality and occurrence errors.

In especial case if $p = 2$ then for $x, y \in \chi$ one has:

$$d_{c,p=2}(x,y) := \min(c, \|x - y\|)$$

In this paper has been considered $p = 2$ and $c = 10$.

Distance error is measured by OSPA metric [8]. The reason of using this metric is that other vector metrics (e.g. MMSE[1]) aren't suitable for finite-set-valued estimation error. Vector based metrics are based on the classical theory, and it fails in dealing with vectors of variable length. So OSPA is used as an alternative metric. This metric is based on Wasserstein Construction and it jointly evaluates the error in the weight of the task in workflow and number estimates; when workflow task states have similar cardinalities, it inherits optimal interpretation of miss-distance assignment.

In the next section the selection of *k* in *k-means* algorithm is considered.

4. SELECTION OF K IN K-MEANS

Basic *k-means* is an extremely simple and efficient algorithm that clusters data with a detected number of clusters, *k*. Detection of the measure often labeled *k* in the *k-means* is an important task. Improper selection of *k* leads to incorrect clustering results (see fig. 4). On the other hand, detection of *k* automatically is a hard algorithmic problem.

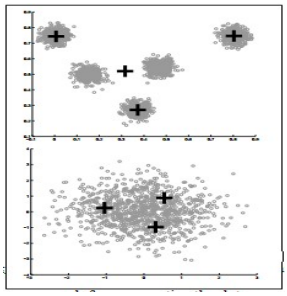

Fig. 4. Two clustering where *k* is improperly selected [20]. Dark crosses are *k-means* **c**enters. On the **up**, there are too few centers; five should be used. On the **bottom**, too many centers are used; one center is enough for representing the data.

Some approaches can be used for *k* detection in *k-means* algorithm. Detection of *k* in *k-means* can be attained using an evaluation function that considers factors affecting the selection of *k* in *k-means*. The proposed method selects *k* of *k-means* with automated methods, without the need of user selection. The methods have a cost function of solving the trade-off between level of detail of each motif index and the resolution of tasks. In this case, workflow indices are distributed vector weights of workflows.

This section presents a meta-heuristic algorithm based on SFLA for detection of *k*. SFLA flowchart is shown in Fig. 5.

The fitness function is shown in Algorithm 1. First, the algorithm sort clusters ascending based on the ratio of the size of clusters and cluster weights. Then based on the nearest amount of the ratio it selects the detected k as the best *k*. Based on the selected best *k* in the SFLA algorithm all other frogs come near to the new determined *k*.

Algorithm 1: Fitness function of SFLA

```
Fitness function(k, clusters){
 for (all k in clusters){
     calculate_ratio(clusters, k);
     find suitable k based on the calculated ratio;
 if (find suitable k)
     return k;
 }
}
```

In order to test the work on real datasets, this work used biological experiment workflows (120 workflows) extracted from ArrayExpress experimental design workflows [14]. Finally, the proposed method for *k* detection is compared with the classic GA and PSO algorithms as benchmarks.

5. RESULTS AND DISCUSSION

In order to test the performance and effectiveness of the proposed method, the work performed various experiments on different datasets. For this purpose, this work performed workflow clustering on a synthesis dataset. Then, it applied the clustering method on a subset of ArrayExpress experimental design workflows [14], as a real dataset.

This work has evaluated the accuracy of the proposed method by comparing predictions of smaller representative experiments run on a synthetic dataset. Besides, it has studies the structure of so-called real workflows, obtained from biological experiments, as a real dataset. The dataset used in this paper is based on ISA-Tab format [13]. As a result the fig. 6 shows the *k* selection error of *k* learning in *k-means* clustering.

With respect to considered assumptions the results are as follows:

The selection error which has been attained for 3 true *K*s (2, 10 and 20) has been shown in the Figs. 6 and 7.

The results of the fig. 6 are for synthetic dataset, and the results of the fig. 7 are for biological experiments dataset.

In the Figs. 6 and 7 the horizontal axis show the true *k* should be selected by the method. The vertical axis shows the detected *k*. Based on the fig. 6 and fig. 7, the SFLA algorithm is better than GA and PSO algorithms. The results of the PSO algorithm in lowers Ks is the best, but in higher Ks the SFLA works better than other.

In summary, the SFLA worked better for the k selection of all the models selected.

Most workflows are complex, in the sense that they present many non trivial topological features. In this paper the utilized workflow similarity measure aggregates the similarity of the set of occurring workflow elements and the similarity of the abstracted control flow structure. The abstracted control flow structure has been obtained by learning flexible workflow representations as the first step towards learning semantics.

6. CONCLUSION

In this paper, solutions of workflow clustering problem reviewed. Some clustering methods are text based and others are structure based. While some increases of complexity of workflows are the results of the workflow structures, some increases of workflow complexity are the results of workflow concepts. This work, with learning workflow representations has embedded workflows in the context of similar workflows. Then, by the finite set based clustering it has evaluated the model on executions of synthetic workflow dataset and real biological experiment dataset. Finally, it concluded that the SFLA algorithm is better than GA and PSO algorithms. The results of PSO algorithm in lowers *K*s is the best, but in higher Ks the SFLA works better than other. In summary, the proposed method based on SFLA worked better for the *k* selection of all the models selected.

Most of the clustering methods focused on task data as the smallest unit of data in clustering. One way forward is the use of aggregation, specifically by aggregating common functional structures

called motifs. In the future work, more advanced measures for the similarity of workflow elements could be investigated. For instance, further properties of the workflow elements could be included like the input and output parameters specifying the data flow.

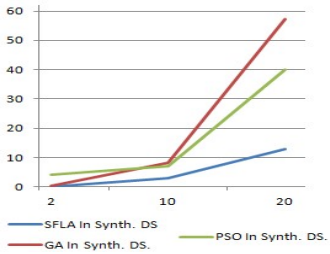

Fig. 6. *k* selection error in synthetic dataset; horizontal axis shows true *k*

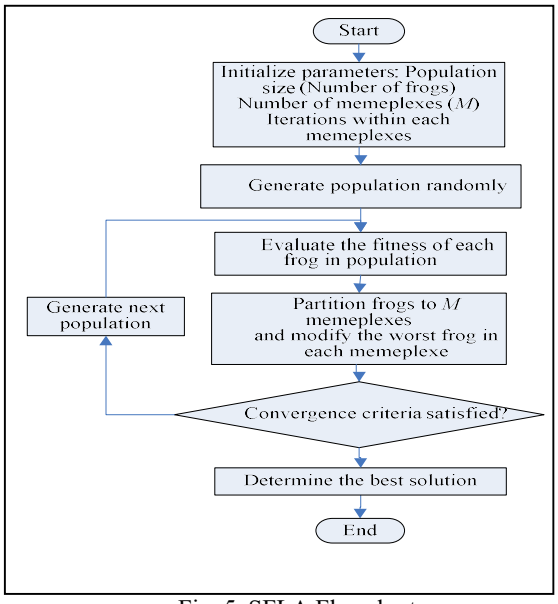

Fig. 5. SFLA Flowchart

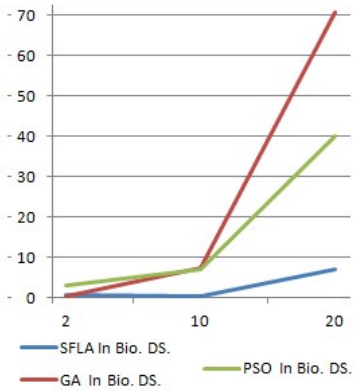

Fig. 7. *k* selection error in real biological dataset; horizontal axis shows true *k*

REFRENCES

[1] E. Lord, A.B. Diallo, and V. Makarenkov, Classification of bioinformatics workflows using weighted versions of partitioning and hierarchical clustering algorithms, BMC Bioinformatics 16(68), 2015.

[2] E. Santos, L. Lins, J. Ahrens, J. Freire, and C. Silva, A First Study on Clustering Collections of Workflow Graphs, 2008.

[3] W. Chen, R.F.d. Silva, E. Deelman, R. Sakellariou, Using imbalance metrics to optimize task clustering in scientific workflow executions, Future Generation Computer Systems 46: 69–84, 2015.

[4] J. Starlinger, S. C.ohen-Boulakia, S. Khanna, S. Davidson, U. Leser, Effective and Efficient Similarity Search in Scientific Workflow Repositories , Future Generation Computer Systems, Elsevier, 2015.

[5] R. Bergmann, G. Muller, and D. Wittkowsky, Workflow Clustering Using Semantic Similarity Measures, in proceedings of 36th Annual German Conference on AI, 2013.

[6] Ch. Li, M. Reichert, and A. Wombacher, On Measuring Process Model Similarity based on High-level Change Operations, in Proceedings of 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008.

[7] E. Maguire, Ph. Rocca-Serra, S.-A. Sansone, J. Davies, and M. Chen, Visual Compression of Workflow Visualizations with Automated Detection of Macro Motifs, IEEE Transactions on Visualization and Computer Graphics, vol. 19, no. 12, 2013.

[8] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, A consistent metric for performance evaluation of multi-object filters, IEEE Trans. Signal Process., 86 (8): 3447–3457, 2008.

[9] M. Baum, B. Balasingam, P. Willett, and U.D. Hanebeck, OSPA Barycenters for Clustering Set-Valued Data, In Proceedings of the 18th International Conference on Information Fusion (Fusion 2015), Washington, USA, July 2015.

[10] M. Eusuff, K. Lansey, and F. Pasha, Shuffled frog-leaping algorithm: A mimetic meta-heuristic for discrete optimization, Eng. Optim., 38(2): 129–154, 2006.

[11] A. Fiannaca, M.L. Rosa, R. Rizzo, A. Urso, S. Gaglio, An expert system hybrid architecture to support experiment management, in Journal of Expert Systems with Applications, 41: 1609-1621, 2014.

[12] L.-F. Wang, J.-Ch. Zeng, and Y. Hong, Estimation of Distribution Algorithm Based on Copula Theory, Chapter book of Exploitation of Linkage Learning in Evolutionary Algorithms, pp 139-162, 2010.

[13] A. González-Beltrán, S. Neumann, E. Maguire, S.-A. Sansone, Ph. Rocca-Serra, The Risa R/Bioconductor package: integrative data analysis from experimental metadata and back again, in Journal of BMC Bioinformatics, 15(Suppl 1):S11, 2014.

[14] Arrayexpress. https://www.ebi.ac.uk/arrayexpress/, Last Accessed: 13 January 2017.

[15] S. Talukder, Mathematical Modeling and Applications of Particle Swarm Optimization, Master of Science Thesis, 2011.

[16] M. Mitchell. An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA, 1996.

[17] D. Quang and X. Xie, DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences, in Journal of Nucleic Acids Research, 44 (11), 2016.

[18] T. Koohi, and M. Zahedi, Linear Merging Reduction: A Workflow Diagram Simplification Method, 8th International Conference on Information and Knowledge Technology, 2016.

[19] S. Scardapane, D. Comminiello, A. Hussain and Aurelio Uncini, Group Sparse Regularization for Deep Neural Networks, arXiv:1607.00485 [stat.ML], 2016.

[20] S. Shahrivari and S. Jalili, Fast Parallel All-Subgraph Enumeration Using Multicore Machines, in Journal of Scientific Programming,, 2015.

[21] S.A. Sansone, P. Rocca-Serra, M. Brandizi, A. Brazma, D. Field, J. Fostel, A.G. Garrow, J. Gilbert, F. Goodsaid, N. Hardy, and P. Jones, The first RSBI (ISA-TAB) workshop:“can a simple format work for complex studies?”. OMICS A Journal of Integrative Biology, 12(2), pp.143-149, 2008.

[22] J. Starlinger, B. Brancotte, S. Cohen-Boulakia, U. Leser, Similarity Search for Scientific Workflows, in Proceedings of the VLDB Endowment (PVLDB), VLDB Endowment, 7 (12), pp.1143-1154, 2014.

[23] L. Wang, J. Zeng, and Y. Hong. Estimation of distribution algorithm based on copula theory. In Proceedings of the IEEE Congress on Evolutionary Computation, IEEE Press, pp. 1057–1063, 2009.

[24] X.l. DONG, S.q. LIU, T. TAO, Sh.p. LI, K.l. XIN, A comparative study of differential evolution and genetic algorithms for optimizing the design ofwater distribution systems, Journal of Zhejiang University-SCIENCE A (Applied Physics & Engineering), 13(9):674-686, 2012.