MR-SimLab: Scalable subgraph selection with label similarity for big data

Wajdi Dhifli, Sabeur Aridhi, Engelbert Mephu Nguifo

MR-SimLab: Scalable Subgraph Selection with Label Similarity for Big Data

Wajdi Dhifli[a,*], Sabeur Aridhi[b], Engelbert Mephu Nguifo[c,*]

[a] *Institute of Systems and Synthetic Biology (iSSB), University of Evry-Val-d'Essonne, Evry, France*
[b] *University of Lorraine, LORIA, Campus Scientifique, BP 239, 54506 Vandoeuvre-lès-Nancy, France*
[c] *LIMOS, Blaise Pascal University, Clermont University, Clermont-Ferrand, F-63000, France*

Abstract

With the increasing size and complexity of available databases, existing machine learning and data mining algorithms are facing a scalability challenge. In many applications, the number of features describing the data could be extremely high. This hinders or even could make any further exploration infeasible. In fact, many of these features are redundant or simply irrelevant. Hence, feature selection plays a key role in helping to overcome the problem of information overload especially in big data applications. Since many complex datasets could be modeled by graphs of interconnected labeled elements, in this work, we are particularly interested in feature selection for subgraph patterns. In this paper, we propose MR-SimLab, a MapReduce-based approach for subgraph selection from large input subgraph sets. In many applications, it is easy to compute pairwise similarities between labels of the graph nodes. Our approach leverages such rich information to measure an approximate subgraph matching by aggregating the elementary label similarities between the matched nodes. Based on the aggregated similarity scores, our approach selects a small subset of informative representative subgraphs. We provide a distributed implementation of our algorithm on top of the MapReduce framework that optimizes the computa-

*Corresponding author
Email addresses: wajdi.dhifli@univ-evry.fr (Wajdi Dhifli), sabeur.aridhi@loria.fr (Sabeur Aridhi), mephu@isima.fr (Engelbert Mephu Nguifo)

tional efficiency of our approach for big data applications. We experimentally evaluate MR-SimLab on real datasets. The obtained results show that our approach is scalable and that the selected subgraphs are informative.

Keywords: Feature selection, subgraph mining, label similarity, MapReduce

1. Introduction

In the era of big data, the number and size of available databases is becoming extremely large. These databases are composed of data that are often represented as graphs, where the nodes represent labeled entities that are interlinked through various relationships [1, 2, 3, 4]. In this context, it is crucial to develop scalable algorithms that allow to efficiently handle and mine such huge graph databases.

Frequent subgraph mining (FSM) is one of the most important and active fields in data mining. It consists on finding subgraphs that occur at least δ times in a graph database where δ is a user-defined support threshold. Many FSM algorithms have been proposed in the literature and made this task feasible such as FFSM [5], gSpan [6] and GASTON [7]. However, the exponential number of discovered subgraphs by these algorithms makes them prone to the problem of *"information overload"*, which may hinder data exploration or even makes it infeasable [8, 9]. For example, in an AIDS antiviral screen dataset composed of only 422 chemical compounds, there are more than 1 million frequent substructures when the minimum support threshold is 5%. This problem becomes even more serious with dense graphs such as social networks and protein 3D-structures.

In fact, the issues raised from the huge number of frequent subgraphs are mainly due to two factors, namely *redundancy* and *significance* [10]. Redundancy in frequent subgraphs is caused by structural and/or semantic similarity, since many discovered subgraphs differ slightly in structure and may infer similar or even the same meaning. Moreover, the significance of the discovered subgraphs is only related to their frequencies and occurrence lists.

In many real-world databases, the nodes of the graphs could be (a) heterogeneous referring to entities of different categories (*e.g.*, groups of genes playing different functions, tweets of different hash-tags, movies of different types, *etc.*) and (b) could carry rich semantics (*e.g.*, nodes in social networks represent persons described by names, friends lists, pictures, *etc.*). In this context, these nodes may share, at the semantic level, various kinds of similarity that could be measured using the domain knowledge. Based on this similarity, it is possible to design a distance matrix between node labels. Incorporating such rich information in graph mining algorithms will make an asset for detecting semantic similarity between graphs.

The advent of big data has raised unprecedented challenges for both subgraph mining and subgraph selection algorithms. The tremendously increasing size of existing graph databases makes it impossible to handle subgraph mining/selection on a single machine. Moreover, ultrahigh number of patterns implies massive memory requirements and a high computational cost for further exploration by learning algorithms. The use of parallel and/or distributed techniques for both subgraph mining and selection in big data contexts is becoming all the more urgent [11, 12].

In this paper, we present MR-SimLab, a scalable and distributed approach for representative subgraph selection based on *MapReduce* [13]. MR-SimLab is based on our previous work in [3] where we proposed an approach for smoothing the distribution of protein 3D-structure motifs. In [3], we proved the efficiency of the proposed approach through an extensive experimental evaluation and we showed that it outperformed multiple state-of-the art subgraph selection approaches on the classification of benchmark datasets. MR-SimLab extends our previous work on two levels. First, we propose a generalized formalization of the approach in [3] that allows MR-SimLab to be used on any type of data. Second, we propose a MapReduce-based implementation for MR-SimLab that allows it to scale efficiently in big data scenarios. Unlike the strict graph isomorphism, our approach performs an approximate graph isomorphism between subgraph patterns to detect semantic similarity between them. In particular,

our approach takes advantage of the similarity between node labels that are defined in the form of a similarity matrix that can be found or easily defined from the domain knowledge. We propose a heuristic method for selecting a small set of representative subgraphs based on an approximate label matching. We design a MapReduce-based implementation of our subgraph selection algorithm that can efficiently scale to very large graph sets. We empirically evaluate the effectiveness and efficiency of our approach. Experimental results on real-world graph datasets show that our approach is able to summarize the initial large set of subgraphs into a small subset of informative representatives compared to multiple existing state-of-the-art approaches. We also show that our approach scales efficiently to big input graph sets.

2. Related Works

Subgraph selection. Several subgraph selection approaches have been proposed in the literature [10, 14, 15, 16, 17, 18, 19, 20]. To the best of our knowledge, most existing subgraph selection approaches are based on structural similarity [14, 21] and/or statistical significance such as frequency and coverage (closed [22], maximal [23]) or discrimination power [10, 24]. Yet, the *prior* information and knowledge about the domain are often ignored and most existing subgraph mining and selection approaches ignore the semantic similarity between labels of the graph nodes. Only very few recent works have considered similarity between node labels in subgraph mining. In [1], the authors focused on the problem of mining an approximate set of frequent subgraphs in a single large graph while considering similarities between node labels. Two subgraphs are considered to be similar if they have the same structure and if the similarity between the labels of all pairs of mapping nodes is below a cost threshold. In [2], the authors proposed *NeMa*, a graph querying approach that allows ambiguity in node labels. The aim of *NeMa* is to identify the (top-k) matches of a given query graph in a (typically large) target graph. *NeMa* uses a label cost function

to approximate subgraph isomorphism where the similarity between pairs of nodes is measured by the amount of shared words in their labels. Two nodes are considered to be a match if their label similarity is less then a predefined *label noise* threshold. Both approaches of [1] and [2] focused on mining from a single large graph, respectively, top-k matches of a given query graph and a set of approximate frequent subgraphs in the presence of label similarities. Both approaches are similar to the one proposed in this paper in the sense that they also consider semantic similarities between node labels. However, here we focus on a different problem that is the selection of an exact representative subset of subgraphs that are extracted from a dataset of multiple graphs in a distributed manner.

Subgraph selection for big data. Whereas several large scale subgraph extraction approaches have been proposed in the literature [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35], far less attention has been devoted to subgraph selection and only very few generic feature selection approaches exist for big data scenarios [8, 36, 37]. It is unclear how existing subgraph selection approaches could scale to very large data. In fact, in [8], the authors emphasized the growing need for scalable feature selection methods in various types of applications and showed how existing methods proved to be inadequate for big data. We are unaware of any existing distributed subgraph selection approach that operates in feature space (*i.e.*, on the relation between subgraphs themselves and not their transaction lists) and big data scenarios.

3. Preliminaries and Definitions

3.1. Definitions

Let $\mathcal{G}$ be a set of graphs. Each graph $G = (V, E, L)$ of $\mathcal{G}$ is given as a collection of nodes V and edges E. The nodes of V are labeled within an alphabet Σ and L is the function that maps each node in V to its respective

label in Σ, $L : V \rightarrow \Sigma$. We denote by $|V|$ the number of nodes (also called the graph order) and by $|E|$ the number of edges (also called the graph size).

Definition 1. *(Subgraph isomorphism) A subgraph isomorphism exists between two graphs $G = (V, E, L)$ and $G' = (V', E', L)$, denoted by $G \subseteq G'$, if there exists an injective function $f : V \rightarrow V'$, such that:*

- $\forall u, v \in V : \forall (u, v) \in E \rightarrow (f(u), f(v)) \in E'$
- $\forall v \in V, L(v) = L(f(v))$
- $\forall (u, v) \in E : L(u, v) = L(f(u), f(v))$

Under these conditions, the function f is called an embedding of G in G', G is called a subgraph of G' and G' is called a supergraph of G.

Definition 2. *(Graph isomorphism) A graph isomorphism exists between G and G' if the function f is bijective.*

Definition 3. *(Frequent subgraph) Given a subgraph g, a graph database $\mathcal{G}$, and a minimum frequency threshold δ (minimum support), let $\mathcal{G}_g$ be the subset of $\mathcal{G}$ where g appears (i.e., g has a subgraph isomorphism in each graph in $\mathcal{G}_g$). The number of graphs where g occurs is denoted by $\mid \mathcal{G}_g \mid$. The subgraph g is considered as frequent if:*

$$support(g) = \frac{\mid \mathcal{G}_g \mid}{\mid \mathcal{G} \mid} \geq \delta \tag{1}$$

Definition 4. *(Label similarity matrix) A similarity matrix $\mathcal{A}$ over L is defined as:*

$$\mathcal{A} : \left| \begin{array}{rcl} \Sigma^2 & \rightarrow & [\bot, \top] \subset \mathbb{R}^{\geq 0} \\ (l, l') & \rightarrow & x \end{array} \right. \tag{2}$$

where $l, l' \in \Sigma$ and x is the similarity score between them such that $\forall l, l' \in \Sigma, \bot \leq x \leq \top$. We suppose that $\mathcal{A}$ exists or can be defined from the domain knowledge. Typically, $\mathcal{A}$ is symmetric and the diagonal entries are the maximal values in their respective column and row entries in $\mathcal{A}$. This is because we

consider that no label could be similar to another one more than itself, $\forall l, l' \in \Sigma, \mathcal{A}(l, l') \leq \mathcal{A}(l, l)$.

Definition 5. *(Structural graph isomorphism) We denote by* ϕ *the function that checks if two graphs* $G = (V, E, L)$ *and* $G' = (V', E', L)$ *are structurally isomorphic (having the same topology). We denote* $\phi(G, G') = true$ *, if there exists a bijective function* $f : V \leftrightarrow V'$ *such that* $\forall u, v \in V$ *if* $(u, v) \in E$ *then* $(f(u), f(v)) \in E'$ *and vice versa. Note that* ϕ *tests only the structure and ignores the labels.*

Definition 6. *(Elementary identity score) Given a node* v *having a label* $l \in \Sigma$*, the elementary identity score* $I_{el}(v)$ *measures the degree of distinction of* v *from any other node depending on its label* l*.*

$$I_{el}(v) = \frac{\mathcal{A}(l, l)}{\sum_{i=1}^{|\Sigma|} \mathcal{A}(l, l_i)} \tag{3}$$

The lower is $I_{el}(v)$*, the more likely* v *is to be similar with other nodes based on their labels.*

Remark 1. *Note that Definition 6 could be straightforwardly used for graphs with single labeled nodes. For graphs of multi-labeled nodes, it could be easily generalized by averaging the scores over all the labels of the nodes. Formally:*

$$I_{el}(v) = \frac{1}{|L_v|} * \sum_{i=1}^{|L_v|} \frac{\mathcal{A}(l_i, l_i)}{\sum_{j=1}^{|\Sigma|} \mathcal{A}(l_i, l_j)} \tag{4}$$

where L_v *is the set of all labels of the node* v*, i.e.,* $L_v = \{\forall l \in L(v)\}$*,* $L_v \subseteq L$ *and* $|L_v|$ *is the number of labels over the node* v*.*

Let Ω be the set of frequent subgraphs extracted from $\mathcal{G}$.

Definition 7. *(Graph representativity estimation) Given a graph* $G = (V, E, L) \in \Omega$*,* $\widehat{R}_s(G)$ *measures the estimated representativity of* G *according to the similarities shared by its nodes' labels in the similarity matrix* $\mathcal{A}$*. Formally:*

$$\widehat{R}_s(G) = 1 - \prod_{i=1}^{|V|} I_{el}(V[i]) \tag{5}$$

where $\prod_{i=1}^{|V|} I_{el}(V[i])$ *estimates the identity score of* G*, i.e., it measures how different* G *is from all the other possible subgraphs of* Ω *in terms of label similarity.*

Definition 8. *(Elementary label similarity) Given two nodes* v *and* v' *having respectively the labels* $l, l' \in \Sigma$*,* $E_{ls}(v, v')$ *measures the label similarity between* v *and* v' *with respect to* $\mathcal{A}$*.*

$$E_{ls}(v, v') = \frac{2 * \mathcal{A}(l, l')}{\mathcal{A}(l, l) + \mathcal{A}(l', l')} \tag{6}$$

Remark 2. *Similarly to Definition 6, Definition 8 could be easily extended for multi-labeled nodes by averaging the scores over all the matching labels of the nodes. Formally:*

$$E_{ls}(v, v') = \frac{1}{|L_v|} * \sum_{i=1}^{|L_v|} \frac{2 * \mathcal{A}(l_i, l'_i)}{\mathcal{A}(l_i, l_i) + \mathcal{A}(l'_i, l'_i)} \tag{7}$$

where L_v *and* $L_{v'}$ *are respectively the sets of all labels of the nodes* v *and* v'*, i.e.,* $L_v = \{\forall l \in L(v)\}$ *and* $L_{v'} = \{\forall l' \in L(v')\}$*.*

Definition 9. *(Graph label similarity) Given two graphs* $G = (V, E, L)$ *and* $G' = (V', E', L)$ *such that* $\{G, G'\} \subseteq \Omega$ *and* $\phi(G, G') = true$*, we denote by* $Sim_\phi(G, G')$ *the label similarity score between* G *and* G'*. In other words, it measures the similarity between the labels of every matching pair of nodes from* G *and* G' *according to* ϕ *and with respect to* $\mathcal{A}$*. Formally:*

$$Sim_\phi(G, G') = \frac{\sum_{i=1}^{|V|} E_{ls}(V[i], V'[i])}{\mid V \mid} \tag{8}$$

Definition 10. *(Representative graph) A graph* G^* *is said to be representative for another graph* G*, denoted by* $R(G^*, G, \tau) = true$*, iff:*

- $Sim_\phi(G^*, G) \geq \tau$*, where* τ *is a user-defined threshold and* $\tau \in [0, 1]$

- *and* $\widehat{R}_s(G^*) \geq \widehat{R}_s(G)$.

Definition 11. *(Representative graph-set) Given a threshold* τ *and a graph-set* Ω, Ω^* *is said to be a representative subset of* Ω, *denoted by* $R_{set}(\Omega) = \Omega^*$, *iff:*

$$\forall G^* \in \Omega^*, \nexists G \mid R(G, G^*, \tau) = true \tag{9}$$

Proposition 1 (Estimated null representativity). *For a graph* $G = (V, E, L) \in \Omega$, *if* $\widehat{R}_s(G) = 0$ *then* G *is directly considered as a representative, i.e.,* $G \in \Omega^*$.

Proof 1. *The proof can simply be deduced from Definitions 7 and 11. If* $\widehat{R}_s(G) = 0$ *then,* $\nexists G^* \in \Omega^* \mid R(G^*, G, \tau) = true$, *with respect to* $\mathcal{A}$.

Definition 12. *(Merge support) Given two subgraphs* $\{g^*, g\} \subset \Omega$, *if* $R(g^*, g, \tau) = true$ *then* g^* *will represent* g *in the list of graphs where the latter occurs, i.e., in* $\mathcal{G}_g$. *Formally:*

$$\mathcal{G}_{g^*} = \mathcal{G}_{g^*} \cup \mathcal{G}_g \mid \forall (g^*, g), R(g^*, g, \tau) = true \tag{10}$$

where $\mathcal{G}_{g^*}$ *and* $\mathcal{G}_g$ *are respectively the set of graph occurrences of* g^* *and* g.

3.2. Illustrative example

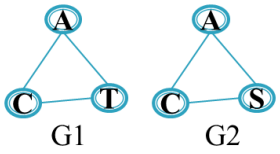

Figure 1: An example of two similar subgraphs.

Given the following toy subgraphs G1 and G2 (see Figure 1), we want to check the label similarity between them with respect to a label similarity matrix given in Table 1. If both subgraphs are considered to be similar (according to a predefined similarity threshold), which subgraph will be considered the representative one? The similarity test is performed based on the definitions in Section 3.1. The similarity test is processed as follows:

Table 1: An example of a label similarity matrix.

A	4			
C	0	6		
S	2	1	4	
T	0	1	2	5
	A	C	S	T

- Elementary identity score (we compute the elementary identity scores for all nodes of G1 and G2 according to Definition 6):
  - $I_{el}(A) = \frac{\mathcal{A}(A,A)}{\sum_{i=1}^{4} \mathcal{A}(A,l_i)} \simeq 0.667$
  - $I_{el}(C) = \frac{\mathcal{A}(C,C)}{\sum_{i=1}^{4} \mathcal{A}(C,l_i)} \simeq 0.75$
  - $I_{el}(S) = \frac{\mathcal{A}(S,S)}{\sum_{i=1}^{4} \mathcal{A}(S,l_i)} \simeq 0.444$
  - $I_{el}(T) = \frac{\mathcal{A}(T,T)}{\sum_{i=1}^{4} \mathcal{A}(T,l_i)} \simeq 0.625$
- Graph representativity estimation (according to Definition 7):
  - $\widehat{R}_s(G1) = 1 - (I_{el}(A) * I_{el}(C) * I_{el}(T)) \simeq 1 - (0.667 * 0.75 * 0.625) \simeq 1 - 0.313 \simeq 0.687$
  - $\widehat{R}_s(G2) = 1 - (I_{el}(A) * I_{el}(C) * I_{el}(S)) \simeq 1 - (0.667 * 0.75 * 0.0.444) \simeq 1 - 0.022 \simeq 0.978$
  - Thus $\widehat{R}_s(G1) < \widehat{R}_s(G2)$.
- Structural isomorphism (Definition 5): $\phi(G1, G2) = true$. This function checks if $G1$ and $G2$ are isomorphic and returns all possible mappings between them.

Note that we compute the similarity scores for every possible mapping between $G1$ and $G2$, until a similarity score with a value greater than or equal to the given similarity threshold is found or no other mapping is possible. Here, we only show, as an example, how the similarity score is computed for only one mapping between $G1$ and $G2$ among all possible ones. The considered mapping for this example is: $A \leftrightarrow A$, $C \leftrightarrow C$, $S \leftrightarrow T$.

- Pattern substitution score (Definition 9):
  - $Sim_\phi(G1, G2) = \frac{E_{ls}(A,A)+E_{ls}(C,C)+E_{ls}(S,T)}{|V1|} = \frac{1+1+0.444}{3} \simeq 0.815$
  - Thus, G1 and G2 are considered similar for all substitution thresholds $0 \leq \tau \leq 0.815$

For all similarity thresholds $\forall \tau \leq 0.815$, $G2$ is considered a representative for $G1$ since $\widehat{R}_s(G1) < \widehat{R}_s(G2)$ (see Definition 10). We then merge the support lists of $G2$ and $G1$, and we remove $G1$:

- Joining support (Definition 12): $\mathcal{G}_{G2} = \mathcal{G}_{G2} \cup \mathcal{G}_{G1}$.
- Remove $G1$.

3.3. MapReduce

MapReduce is a programming model that has been proposed by Google in 2004 [13] to deal with parallel processing of large datasets. The basic components of a MapReduce program are as follows:

1. **Data reading:** in this step, the input data is transformed into a set of key-value pairs. These data may be gathered from various data sources such as file systems, database management systems or the main memory. The input data is split into several fixed-size chunks. Each chunk is processed by one instance of the Map function.
2. **Map phase:** for each chunk having the key-value structure, the respective Map function is triggered. The latter produces a set of intermediate key-value pairs.
3. **Reduce phase:** the Reduce function merges all key-value pairs having the same key and computes the final result.

4. Scalable Subgraph Selection using MapRedcue

In this section, we first present SimLab, a selection algorithm that summarizes an input set of subgraphs (having the same number of nodes) into a small

subset of representatives. Then, we present MR-SimLab, a MapReduce-based implementation of SimLab.

4.1. SimLab *algorithm*

Given a set of subgraphs $P_i \subseteq \Omega$, a label similarity matrix $\mathcal{A}$ and a minimum similarity threshold τ, SimLab captures a subset of representative subgraphs P_i^* such that $P_i^* \subseteq P_i$ and $P_i^* \subseteq \Omega^*$ with respect to $\mathcal{A}$ and τ. The general procedure of SimLab is described in Algorithm 1. First, SimLab ranks the input set of subgraphs P_i in descending order by their graph representativity estimation ($\widehat{R}_s$) that is computed according to Definition 7. Then, P_i^* is browsed starting from the subgraph having the highest $\widehat{R}_s$. For each subgraph g, we look for all the other ones it could represent such that if $R(g, g', \tau) = true$ then we remove the subgraph g' from P_i^* and we update the support list of g to contain the additional occurrences of g', *i.e.*, $\mathcal{G}_g = \mathcal{G}_g \cup \mathcal{G}_{g'}$. The remaining subgraphs form the final subset of representatives P_i^*. Note that SimLab uses Proposition 1 to avoid unnecessary computation related to subgraphs with an estimated null representativity. Note also that ranking subgraphs in a descending order by $\widehat{R}_s$ allows SimLab (1) to favor the selection of subgraphs with the highest estimated representativity and (2) to select a subset of representatives P_i^* that is as small as possible since the selected subgraphs are estimated to have a large number of similars.

Based on our label similarity concept, all the remaining representatives in P_i^* are dissimilar, since the latter does not contain any pair of subgraphs g and g' such that $R(g, g', \tau) = true$. This is a reliable summarization of P_i.

Theorem 1. *Let P_i be a set of subgraphs and P_i^* its subset of representatives ($P_i^* = R_{set}(P_i)$) with respect to a label similarity matrix $\mathcal{A}$ and a threshold τ, i.e.,* SimLab$(P_i, \mathcal{A}, \tau) = P_i^*$. *$P_i^*$ cannot be summarized by one of its proper subsets except itself. Formally:*

$$\text{SimLab}(P_i^*, \mathcal{A}, \tau) = P_i^* \tag{11}$$

Proof 2. *Let us suppose that :*

Algorithm 1 SimLab

Require: A set of subgraphs having the same order $P_i = \{g_1, \ldots, g_k\}$, a label similarity matrix $\mathcal{A}$, a similarity threshold τ, $< key = i, value = P_i >$

Ensure: Representative subgraphs $P_i^* = R_{set}(P_i)$

1: $P_i^* \leftarrow \text{sort}(P_i \text{ by } \widehat{R}_s(g))$ {in descending order according to Definition 7}
2: **for all** $g \in P_i^*$ **do**
3: **if** $\widehat{R}_s(g) > 0$ **then**
4: **for all** $g' \in P_i^* \backslash g \mid \widehat{R}_s(g') < \widehat{R}_s(g)$ **do**
5: **if** $R(g, g', \tau) = true$ **then**
6: $\mathcal{G}_g = \mathcal{G}_g \cup \mathcal{G}_{g'}$
7: remove g' from P_i^*
8: **end if**
9: **end for**
10: **end if**
11: **end for**

- ***hypothesis 1:*** $P_i^* \setminus \text{SimLab}(P_i^*, \mathcal{A}, \tau) \neq \emptyset$

- ***hypothesis 2:*** $\text{SimLab}(P_i^*, \mathcal{A}, \tau) \setminus P_i^* \neq \emptyset$

Hypothesis 1 supposes that P_i^* still contains similar subgraphs. This is impossible since according to Definition 11 there does not exist any pair of subgraphs in P_i^* that are similar, i.e., $\forall g^* \in P_i^*, \nexists g \mid R(g^*, g, \tau) = true$.

As for hypothesis 2 to be true, SimLab *is supposed to generate new patterns that were not originally in P_i^*. This contradicts* SimLab *basics especially Definition 11 since* SimLab *is supposed to remove similar subgraphs and not to generate new ones.*

The minimum description length (MDL) principle [38, 39] suggests that given a set of observed data, the best explanation is the one that permits the greatest compression of the data. According to the MDL and Theorem 1, P_i^* represents a reliable summarization of P_i.

Complexity. Suppose that P_i contains n subgraphs of order k. Each group

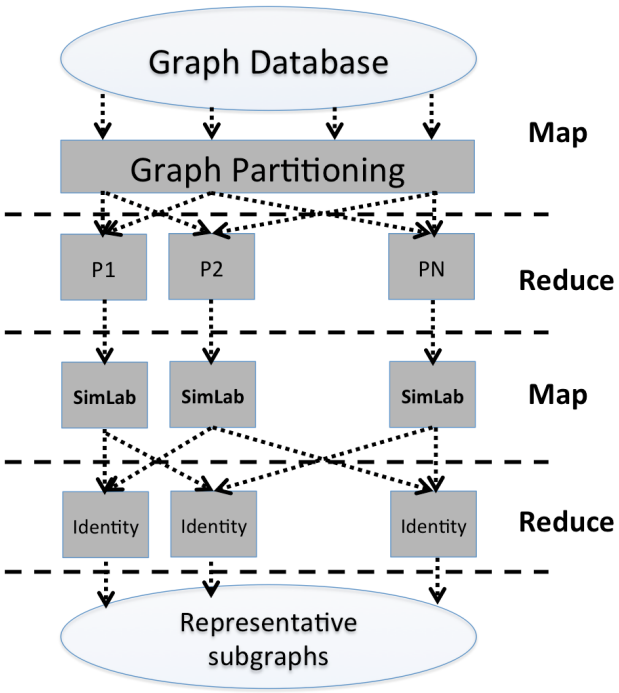

Figure 2: System overview of MR-SimLab

P_i is sorted in $O(n\ log\ n)$. Searching for similar subgraphs requires browsing P_i in $(O(n))$ and browsing in the worst case all remaining subgraphs in $(O(n))$ for each subgraph of P_i. For each pair of subgraphs, we need to check the structure matching in $(O(k))$ and the label similarity in $(O(k))$. This means that searching for representative subgraphs in P_i^k can be done in $O(n^2k^2)$.

4.2. MR-SimLab

The system overview of MR-SimLab is illustrated by Figure 2. MR-SimLab involves two steps: (1) a graph partitioning step and (2) a distributed subgraph selection step. In the following, we give a detailed description of both steps.

Graph partitioning. The aim of this step is to partition the input graph database DB into N partitions where each partition represents a group of graphs having the same order (number of nodes). In our context, the graph database

DB is the input set of subgraphs/patterns Ω extracted from another graph database $\mathcal{G}$. This partitioning step is achieved by a MapReduce pass. The input-set Ω is composed of K subgraph partitions $\Omega = \{GP_1, \ldots, GP_K\}$ where $\Omega = \bigcup_{i=1}^{K} GP_i$ such that $\forall\{GP_i, GP_j\} \subseteq \Omega, GP_i \cap GP_j = \emptyset$. The Map function Map_i reads the assigned data partition $GP_i \subseteq \Omega$ and computes the order of each subgraph in the partition. Map_i outputs a set of $key/value$ pairs each representing a subgraph $g \in GP_i$ and its respective order $\langle order(g), g)\rangle$. The *Reduce* function outputs for each unique intermediate key its respective set of intermediate values (subgraphs of equal number of nodes). The final output is a set of partitions $\{P_1, \ldots, P_N\}$ where $\forall g, g' \in P_i, order(g) = order(g')$ and $\Omega = \bigcup_{i=1}^{N} P_i$ such that $\forall\{P_i, P_j\} \subseteq \Omega, P_i \cap P_j = \emptyset$. The Algorithms 2 and 3 present our Map and *Reduce* functions.

Algorithm 2 Map function

Require: A graph partition $GP_i = \{g_1, \ldots, g_K\}, < key = i, value = GP_i >$

Ensure: Annotated graph partition $AGP = \{\{order(g_1), g_1\}, \ldots, \{order(g_k), g_K\}\}$

1: **for all** g in GP_i **do**
2: $EmitIntermediate(order(g), g)$
3: **end for**

Algorithm 3 Reduce function

Require: $Intermediates = < key = order(g), value = g >$

Ensure: Partitions of subgraphs of equal number of nodes P_{key}

1: $P_{key} \leftarrow \emptyset$
2: **for all** $< key, g >$ in $Intermediates$ **do**
3: $P_{key} \leftarrow P_{key} \cup g$
4: **end for**

Distributed subgraph selection. In this phase, we apply MR-SimLab in order to select the set of representative subgraphs in parallel. This step is achieved by a MapReduce pass. The Map function takes as input one parti-

tion P_i of subgraphs of equal number of nodes and outputs the set of selected representative subgraphs using SimLab. The *Reduce* function of this step is the identity function. Algorithm 4 presents the Map function. In the current implementation of MR-SimLab, we distribute the computation based on the subgraph orders. Yet, we can also distribute for each subgraph all the computations of pairwise similarities such that each comparison will be performed in a separate worker without affecting the quality of the result. This could be useful when the size of each group of subgraphs of the same order is extremely large and does not fit the memory of the running machine. However, it is worth noting that in most cases, the groups of subgraphs of the same order can each fit in memory and be processed in a fast time making the computation faster by avoiding network communication cost.

Algorithm 4 Map function

Require: A partition of subgraphs having the same order $P_i = \{g_1, \ldots, g_k\}$, a label similarity matrix $\mathcal{A}$, a similarity threshold τ and $< key = i, value = P_i >$

Ensure: Representative subgraphs $P_i^* = R_{set}(P_i)$

1: $P_i^* \leftarrow \text{SimLab}(P_i, \mathcal{A}, \tau)$

4.3. Usefulness in real-world applications

MR-SimLab can be used in any real-world application where the data can be represented by a labeled graph and where it is possible to define a distance matrix that quantifies pairwise similarities between node labels. Such a distance matrix can be found or defined based on the domain knowledge that makes leveraging such an important information an asset for defining graph mining approaches that best fit the data. For instance, in gene interaction networks each gene is represented by a node in the graph and is defined by a genomic sequence. Similarity between pairs of genes can be computed by measuring the distances between their genomic sequences (for instance through a pairwise alignment [40]). The same procedure can also be used for protein-protein

interaction networks (PPI) to measure similarity between pairs of proteins represented by nodes in the PPI network. In social networks, similarity between pairs of users (represented by nodes in the network) can be computed by, for instance, measuring the distance between their personal profile information or between their neighborhood connections (like shared friends). Bitmap images can be seen as matrices and thus they can be represented by labeled graphs as well where each node in the graph will hold the information about the color of the unit it represents. Measuring similarity between the nodes of such a graph can easily be computed by measuring the distances between their color information. In the following, we present concrete application examples of MR-SimLab through which we show (1) the efficiency of our approach in selecting informative subgraphs, and (2) its scalability for big data scenarios.

5. Experimental Evaluation

5.1. Experimental Setup

5.1.1. Datasets

In order to assess the efficiency of our approach, we test it on four benchmark graph datasets of protein 3D-structures that have been used previously in multiple studies such as [15] and [20]. Each dataset consists of two classes equally divided into positive and negative sets. Positive examples are proteins that are selected from a considered protein family whereas negative examples are proteins that are randomly gathered from the Protein Data Bank [41]. Each 3D-structure can be represented by a graph where amino acids are graph nodes labeled with the type of amino acid they represent. Two nodes u and v are linked by an edge $e(u, v)$ if the Euclidean distance between the 3D coordinates of their α-Carbon atoms is below a distance threshold δ (we use 7Å). Table 2 summarizes the characteristics of each dataset. SCOP ID, Family name, Pos., Neg., Avg.|V|, Avg.|E|, Max.|V| and Max.|E| correspond respectively to the identifier of the positive protein family in SCOP [42], its name, the number of positive examples, the number of negative examples, the average number of

Table 2: Experimental data.

Dataset	SCOP ID	Family name	Pos.	Neg.	Avg.|V|	Avg.|E|	Max.|V|	Max.|E|
DS1	52592	G-proteins	33	33	246	971	897	3544
DS2	48942	C1-set domains	38	38	238	928	768	2962
DS3	56437	C-type lectin domains	38	38	185	719	775	3016
DS4	88854	Protein kinases, catalyc subunits	41	41	275	1077	775	3016

nodes, the average number of edges, the maximal number of nodes and the maximal number of edges in each dataset.

5.1.2. Experimental Environment

We implemented MR-SimLab on top of MapReduce framework. In order to evaluate the performance of MR-SimLab, we used a cluster of 20 `t2.small` instances on Amazon EC2. Each `t2.small` instance contained 1 virtual 64-bit CPU, 2 GB of main memory, and 8 GB of local instance storage. Experiments for Seq-SimLab (the sequential implementation of MR-SimLab) were performed on an i7 CPU 2.49GHz PC with 6 GB of memory and a Linux Ubuntu operating system.

5.1.3. Label Similarity Matrix

During the evolution, the amino acids composing protein structures can substitute each others. These substitutions are quantified in the so-called substitution matrices. Since there are 20 amino acids, these matrices are of size 20x20 where each entry in the matrix denotes the score of substitution of the i^{th} amino acid by the j^{th} one and inversely. The commonly used substitution matrix for protein alignment is Blosum62 [43]. In this matrix, the substitution score between the i^{th} and j^{th} amino acids is defined as $\frac{1}{\lambda}\ log\ \frac{p_{ij}}{pp_i pp_j}$ where λ is a constant, p_{ij} is the probability that the i^{th} amino acid substitutes the j^{th} one and pp_i, pp_j are respectively the prior probabilities for observing the i^{th} and j^{th} amino acids. In Blosum62, both positive and negative values represent possible substitutions. However, positive scores are given to the more

likely substitutions, whereas negative scores are given to the less likely ones. In our evaluation, we use the Blosum62 to derive our label similarity matrix $\mathcal{A}$ as $\mathcal{A}_{i,j} = e^{Blosum62_{i,j}}$ where $\mathcal{A}_{i,j}$ is the label similarity score between the i^{th} and j^{th} amino acids and $Blosum62_{i,j}$ is the score of substitution between the same pair of amino acids. This transformation allows $\mathcal{A}$ to respect Definition 4 and to give more weight to the positive scores of the more favored substitutions.

5.1.4. Protocol and Settings

We used the state-of-the-art method gSpan [6] to find frequent subgraphs in each dataset with a minimum support of 30%. Then, we use MR-SimLab to select the representative subgraphs with a similarity threshold τ of 30% and the substitution matrix $Blosum62$ [43]. We evaluate MR-SimLab in terms of selection rate (number of selected subgraphs) and classification performance on the datasets. We perform a 5-fold cross-validation classification (5 runs) using the support vector machine classifier (SVM).

5.2. Results and Discussion

5.2.1. Accuracy and Selection Rate

Table 3 shows the number of frequent subgraphs, the number of representative subgraphs that are selected by MR-SimLab as well as the selection rate on the four datasets. The high number of discovered frequent subgraphs is due to the combinatorial nature of graphs. It may increase or decrease depending on the number of graphs in the dataset, their densities and the similarities between them since the more similar are the graphs of the dataset the more common fragments they would have.

The results reported in Table 3 show that MR-SimLab is able to decrease considerably the number of subgraphs by selecting a small subset of representatives. The selection rate shows that the number of representatives $\mid \Omega^* \mid$ does not exceed 13% of the input set of frequent subgraphs $\mid \Omega \mid$ in the worst case with DS3 and it even reaches less than 1% with DS1 and DS4. This shows that incorporating the similarity between labels in the selection constitutes an asset

Table 3: Number of frequent subgraphs (Ω), representative subgraphs (Ω^*) and the selection rate

Dataset	$\mid \Omega \mid$	$\mid \Omega^* \mid$	**Selection rate (%)**
DS1	799094	7297	0.91
DS2	258371	15948	6.17
DS3	114792	14713	12.82
DS4	1073393	10000	0.93

in detecting many similarities between subgraphs that are ignored by current selection approaches.

Effect of variation of the similarity threshold. We perform the same experiments following the same protocol and settings while varying the similarity threshold from 0% to 90% with a step-size of 10%. Figure 3 presents the selection rate using each similarity threshold. In order to check the significance of the set of representatives Ω^* and the effect of varying the similarity threshold on its quality, we use Ω^* as a feature set for the classification of the four datasets using SVM. The classification accuracy using the input set of frequent subgraphs Ω (the line in red) is considered as a standard value for comparison.

In Figure 3, we notice that MR-SimLab reduces considerably the number of subgraphs especially with lower similarity thresholds. In fact, the number of representatives does not exceed 50% for all the similarity thresholds below 80% and it even reaches less then 1% in some cases. Figure 4 shows that this important reduction in the number of subgraphs comes with a notable enhancement of the classification accuracy over all the datasets. In fact, MR-SimLab even reaches full accuracy in some cases (with DS2 and DS4). This shows that our selection is reliable and that our approach allows selecting a subset of representatives that are highly informative.

Comparison with other approaches. In this section, we compare MR-SimLab

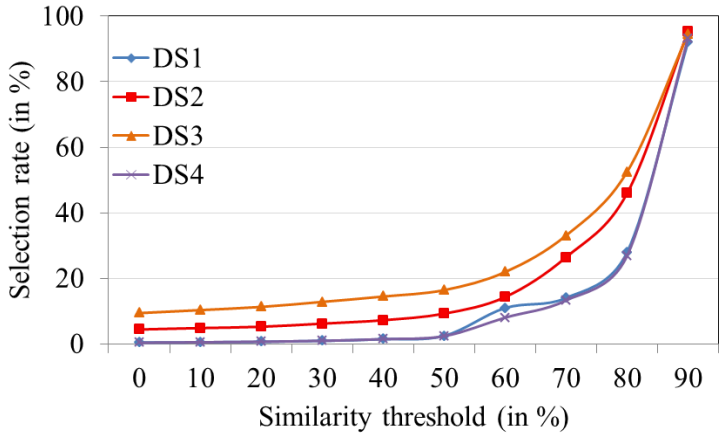

Figure 3: Selection rate of Ω^* from Ω depending on the similarity threshold (τ).

with multiple subgraph selection approaches from the literature namely LEAP [20], gPLS [19], COM [18], GAIA [44], LPGBCMP [15] and D&D [24]. Table 4 shows the obtained classification accuracies using each approach on each of the benchmark datasets. For MR-SimLab, we report the results using a similarity threshold τ of 30%, the previously derived similarity matrix from Blosum62 and the SVM classifier. We also report MR-SimLab$_{max}$ as the best accuracies among all the similarity thresholds $\tau \in [0\%, 90\%]$. For LEAP+SVM, LEAP is used to iteratively discover discriminative subgraphs with a leap length of 0.1. The discovered subgraphs are consider as features to train SVM. For gPLS, the frequency threshold is 30% and the best accuracies are reported among all parameters combinations for $m = \{2, 4, 8, 16\}$ and $k = \{2, 4, 8, 16\}$ where m is the number of iterations and k is the number of patterns per search. COM is used with $t_p = 30\%$ and $t_n = 0\%$. For LPGBCMP, the threshold values of $max_{var} = 1$ and $\delta = 0.25$ were respectively used for feature consistency map building and for overlapping. All the evaluations are performed with a 5-fold cross validation.

As shown in Table 4, our approach outperforms all the other methods in the classification of all the datasets. MR-SimLab$_{max}$ was even able to reach full accuracy with DS2 and DS4. This proves that our approach is competitive, very

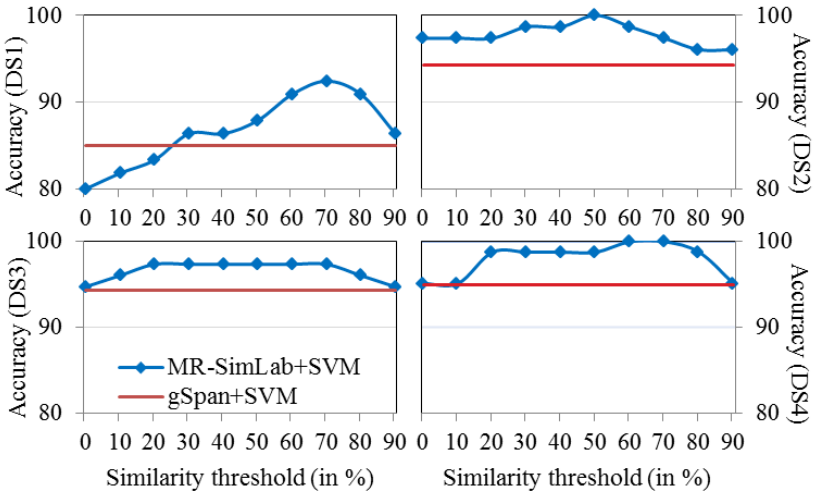

Figure 4: Classification accuracy (in %) using SVM.

promising and that using the label similarity between graph nodes constitutes a big asset in both detecting semantic similarities between subgraph patterns and in selecting a very small subset of informative representatives.

5.2.2. Scalability and Speedup

In this section, we study the scalability of MR-SimLab to big graph databases. In Table 5, we show the running time results for Seq-SimLab, a sequential implementation of our approach in a single process mode, as well as for MR-SimLab the MapReduce-based implementation of our selection approach. As shown in Table 5, the MapReduce implementation of our subgraph selection algorithm is much faster than the sequential one.

Effect of Variation of the Number of Workers. In order to evaluate the influence of some MapReduce parameters on the performance of MR-SimLab, we study the effect of the variation of the number of computation nodes in the MapReduce environment on the running time of our approach. Figure 5 illustrates the obtained results on each of the four datasets.

As shown in Figure 5, MR-SimLab scales with the number of workers for all the datasets. The downward tendency of running time with higher number of

Table 4: Classification accuracy comparison (in [0,1]) of MR-SimLab with other subgraph selection approaches.

Approach/Dataset	DS1	DS2	DS3	DS4	Average accuracy
MR-SimLab$_{max}$+SVM	**0.92**	**1**	**0.97**	**1**	**0.97** ±0.04
MR-SimLab+SVM	0.86	0.99	**0.97**	0.99	0.95 ±0.06
gPLS	0.87	0.91	0.94	0.94	0.92 ±0.03
COM	0.83	0.92	0.96	0.94	0.91 ±0.06
D&D+SVM	0.76	0.96	0.93	0.95	0.9 ±0.09
LEAP+SVM	0.8	0.9	0.91	0.89	0.88 ±0.05
LPGBCMP	0.74	0.9	0.9	0.91	0.86 ±0.08
GAIA	0.66	0.89	0.89	0.87	0.83 ±0.11

workers is very clear. We notice that the running time of MR-SimLab is stable for almost all the datasets starting from 12 workers. This can be explained by the fact that for all datasets, the number of *Map* functions running in parallel is equal to the number of subgraph partitions (subgraph groups P_i having the same number of nodes) which is dataset dependent. Hence, we can say that at the point where we have as many workers as the number of subgraph partitions, adding more workers will have no effect on the running time of MR-SimLab.

6. Conclusion and Further Study

In this paper, we proposed MR-SimLab, a scalable feature selection approach for large sets of subgraph patterns. We introduced a MapReduce-based implementation for our approach that allows it to scale efficiently for extremely large input sets. In contrast to most existing feature selection approaches that focus on the relations between features in transaction space, our approach focuses on the relations between subgraphs in pattern space that is more complex. MR-SimLab leverages the semantic similarities between node labels that are expressed in the form of a similarity matrix that can easily be found or constructed from the domain knowledge. We experimentally show that our approach allows

Table 5: Running time (in seconds) for Seq-SimLab and MR-SimLab on the four datasets using different similarity thresholds

Dataset	τ (in %)	Seq-SimLab	MR-SimLab
DS1	0	1104	237
	30	1243	344
	50	2782	935
DS2	0	187	60
	30	290	110
	50	709	332
DS3	0	73	21
	30	127	26
	50	171	48
DS4	0	1538	296
	30	1916	597
	50	5008	1945

an efficient summarization of the input set of subgraphs by selecting a subset of informative representatives.

An important future direction is to address the skew in our map and reduce functions. This could happen when the constructed subgraph partitions are of unequal sizes leading to longer job execution times. A possible solution for this problem is to perform a nested distribution for jobs with very large input subgraph partitions. Another interesting extension could be to consider an online pairwise comparison between subgraphs to maintain a parsimonious model over time in an online manner such as in [36].

References

[1] P. Anchuri, M. J. Zaki, O. Barkol, S. Golan, M. Shamy, Approximate graph mining with label costs, in: Proceedings of the 19th ACM SIGKDD

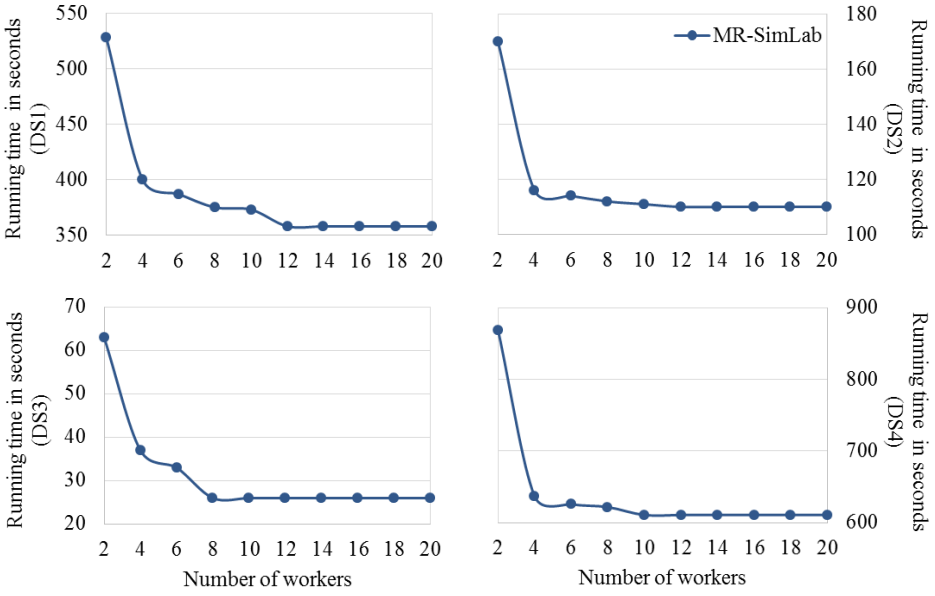

Figure 5: Effect of the number of workers on the running time (in seconds). We used MR-SimLab with τ=30%.

international conference on Knowledge discovery and data mining, KDD '13, ACM, 2013, pp. 518–526.

[2] A. Khan, Y. Wu, C. C. Aggarwal, X. Yan, Nema: Fast graph search with label similarity, Proc. VLDB Endow. 6 (3) (2013) 181–192.

[3] W. Dhifli, R. Saidi, E. Mephu Nguifo, Smoothing 3D protein structure motifs through graph mining and amino-acids similarities, Journal of Computational Biology 21 (2) (2014) 162–172.

[4] W. Dhifli, A. B. Diallo, Protnn: fast and accurate protein 3d-structure classification in structural and topological space, BioData Mining 9 (1) (2016) 30.

[5] J. Huan, W. Wang, J. Prins, Efficient mining of frequent subgraphs in the presence of isomorphism, in: IEEE International Conference on Data Mining (ICDM), 2003, pp. 549–552.

[6] X. Yan, J. Han, gspan: Graph-based substructure pattern mining, in: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM), Vol. 02, IEEE Computer Society, 2002, pp. 721–724.

[7] S. Nijssen, J. N. Kok, A quickstart in frequent structure mining can make a difference, in: ACM knowledge discovery and data mining conference (KDD), 2004, pp. 647–652.

[8] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, Recent advances and emerging challenges of feature selection in the context of big data, Knowledge-Based Systems 86 (C) (2015) 33–45.

[9] M. A. Hasan, Pattern summarization in pattern mining, Encyclopedia of Data Warehousing and Mining, (2nd Ed) (2008).

[10] M. Thoma, H. Cheng, A. Gretton, J. Han, H.-P. Kriegel, A. Smola, L. Song, P. S. Yu, X. Yan, K. M. Borgwardt, Discriminative frequent subgraph mining with optimality guarantees, Statistical Analysis and Data Mining 3 (5) (2010) 302–318.

[11] J. E. Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M. J. Franklin, I. Stoica, Graphx: Graph processing in a distributed dataflow framework, in: Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation, OSDI'14, USENIX Association, Berkeley, CA, USA, 2014, pp. 599–613.

[12] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, J. M. Hellerstein, Distributed graphlab: a framework for machine learning and data mining in the cloud, Proceedings of the VLDB Endowment 5 (8) (2012) 716–727.

[13] J. Dean, S. Ghemawat, Mapreduce: Simplified data processing on large clusters, in: Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6, OSDI'04, USENIX Association, Berkeley, CA, USA, 2004, pp. 10–10.

[14] V. Chaoji, M. A. Hasan, S. Salem, J. Besson, M. J. Zaki, Origami: A novel and effective approach for mining representative orthogonal graph patterns, Statistical Analysis and Data Mining 1 (2) (2008) 67–84.

[15] H. Fei, J. Huan, Boosting with structure information in the functional space: an application to graph classification, in: ACM knowledge discovery and data mining conference (KDD), 2010, pp. 643–652.

[16] M. A. Hasan, M. J. Zaki, Musk: Uniform sampling of k maximal patterns, in: Proceedings of the SIAM International Conference on Data Mining, 2009, pp. 650–661.

[17] M. A. Hasan, M. J. Zaki, Output space sampling for graph patterns, PVLDB 2 (1) (2009) 730–741.

[18] N. Jin, C. Young, W. Wang, Graph classification based on pattern co-occurrence, in: ACM International Conference on Information and Knowledge Management, 2009, pp. 573–582.

[19] H. Saigo, N. Krämer, K. Tsuda, Partial least squares regression for graph mining, in: ACM knowledge discovery and data mining conference (KDD), 2008, pp. 578–586.

[20] X. Yan, H. Cheng, J. Han, P. S. Yu, Mining significant graph patterns by leap search, in: Proceedings of the ACM SIGMOD international conference on Management of data, SIGMOD, ACM, New York, NY, USA, 2008, pp. 433–444.

[21] C. Chen, C. X. Lin, X. Yan, J. Han, On effective presentation of graph patterns: a structural representative approach, in: Proceedings of the 17th ACM conference on Information and knowledge management, ACM, 2008, pp. 299–308.

[22] X. Yan, J. Han, Closegraph: mining closed frequent graph patterns, in: ACM knowledge discovery and data mining conference (KDD), 2003, pp. 286–295.

[23] L. T. Thomas, S. R. Valluri, K. Karlapalem, Margin: Maximal frequent subgraph mining, ACM Transactions on Knowledge Discovery from Data (TKDD) 4 (3) (2010) 10:1–10:42.

[24] Y. Zhu, J. X. Yu, H. Cheng, L. Qin, Graph classification: a diversified discriminative feature selection approach, in: 21st ACM International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2012, pp. 205–214.

[25] S. Aridhi, L. d'Orazio, M. Maddouri, E. M. Nguifo, Density-based data partitioning strategy to approximate large-scale subgraph mining, Information Systems 48 (2015) 213 – 223.

[26] B. Bahmani, R. Kumar, S. Vassilvitskii, Densest subgraph in streaming and mapreduce, Proc. VLDB Endow. 5 (5) (2012) 454–465.

[27] M. Bhuiyan, M. A. Hasan, An iterative mapreduce based frequent subgraph mining algorithm, IEEE Trans. Knowl. Data Eng. 27 (3) (2015) 608–620.

[28] M. A. Bhuiyan, M. A. Hasan, Fsm-h: Frequent subgraph mining algorithm in hadoop, in: Proceedings of the 2014 IEEE International Congress on Big Data, BIGDATACONGRESS '14, IEEE Computer Society, Washington, DC, USA, 2014, pp. 9–16.

[29] S. Hill, B. Srichandan, R. Sunderraman, An iterative mapreduce approach to frequent subgraph mining in biological datasets, in: Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine, BCB '12, ACM, New York, NY, USA, 2012, pp. 661–666.

[30] Y. Liu, J. G. Carbonell, V. Gopalakrishnan, P. Weigele, Conditional graphical models for protein structural motif recognition, Journal of Computational Biology 16 (5) (2009) 639–657.

[31] Y. Luo, J. Guan, S. Zhou, Towards efficient subgraph search in cloud computing environments, in: Proceedings of the 16th International Conference

on Database Systems for Advanced Applications, DASFAA'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 2–13.

[32] A. Silva, W. Meira, Jr., M. J. Zaki, Mining attribute-structure correlated patterns in large attributed graphs, Proc. VLDB Endow. 5 (5) (2012) 466–477.

[33] C. H. C. Teixeira, A. J. Fonseca, M. Serafini, G. Siganos, M. J. Zaki, A. Aboulnaga, Arabesque: a system for distributed graph mining, in: Proceedings of the 25th Symposium on Operating Systems Principles, SOSP 2015, Monterey, CA, USA, October 4-7, 2015, 2015, pp. 425–440.

[34] B. Wu, Y. Bai, An efficient distributed subgraph mining algorithm in extreme large graphs, in: Proceedings of the 2010 International Conference on Artificial Intelligence and Computational Intelligence: Part I, AICI'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 107–115.

[35] S. Aridhi, E. M. Nguifo, Big graph mining: Frameworks and techniques, Big Data Research 6 (2016) 1 – 10.

[36] K. Yu, X. Wu, W. Ding, J. Pei, Towards scalable and accurate online feature selection for big data, in: 2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, 2014, pp. 660–669.

[37] Y. Zhai, Y.-S. Ong, I. W. Tsang, The emerging "big dimensionality", IEEE Computational Intelligence Magazine 9 (3) (2014) 14–26.

[38] P. D. Grünwald, The Minimum Description Length Principle, Adaptive computation and machine learning, The MIT Press, 2007.

[39] J. Rissanen, Modeling by shortest data description, Automatica 14 (5) (1978) 465 – 471.

[40] S. Altschul, W. Gish, W. Miller, E. Myers, D. Lipman, Basic local alignment search tool, Journal of Molecular Biology 215 (1990) 403–410.

[41] H. M. Berman, J. D. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, P. E. Bourne, The protein data bank, Nucleic Acids Research 28 (1) (2000) 235–242.

[42] A. Andreeva, D. Howorth, J.-M. Chandonia, S. E. Brenner, T. J. P. Hubbard, C. Chothia, A. G. Murzin, Data growth and its impact on the scop database: new developments, Nucleic Acids Research 36 (1) (2008) D419–D425.

[43] S. R. Eddy, Where did the blosum62 alignment score matrix come from?, Nature Biotechnology (2004) 1035–1036.

[44] N. Jin, C. Young, W. Wang, GAIA: graph classification using evolutionary computation, in: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, ACM, New York, NY, USA, 2010, pp. 879–890.