Compression technique based on dynamic grouping model of XML documents

N. Belkacem, D. Aissani, F. Semchedine, A. Al-Shammari

Doctoriales de Recherche Opérationnelle, les 12 et 13 Décembre 2018

Introduction

The Simple Object Access Protocol (SOAP) is a basic communication protocol in Web services, which is based on eXtensible Markup Language (XML). SOAP could suffer from high latency and bottlenecks that might occur due to the high network traffic caused by the large number of client requests and the large size of XML Web messages. Previous works have proposed static and dynamic clustering models for SOAP messages to support compression based aggregation tool that could potentially reduce the overall size of SOAP messages in order to reduce the required bandwidth between the clients and their server and increase the performance of Web services. In fact, many applications can be supported by the proposed models such as stock quote marketing, health care information systems, and Web based insurance system. However, some appli-cations cannot get the same benefits and therefore the required features of SOAP-based applications to be supported need to be clarified and checked. Technically, this is based on the kind of SOAP messages used to exchange data as some applications use large-sized data in few XML items while most other applications are exchanging mainly features that require more XML tags. For example, messages with Shakespeare novels are not sup-ported as they have very large sized amount of data in very few XML tags. On the other hand, messages with more XML tags and small size of data items are likely to be supported by the proposed model.

In clustering, we start with traversing the order labelled XML tree, and then we generate the XML vector which is a combination of the structure and content vectors respectively. Technically, the term frequency-inverse document frequency (tf.idf) weighting scheme [8] is used to assign the weights to the terms of XML document, and the weights are stored in a vector matrix. Afterwards, Euclidean distance [3] is used for the similarity measurement by computing the minimum distance between the XML vectors. Then, the similar XML vectors are distributed into the clusters based on the agglomerative clustering model.

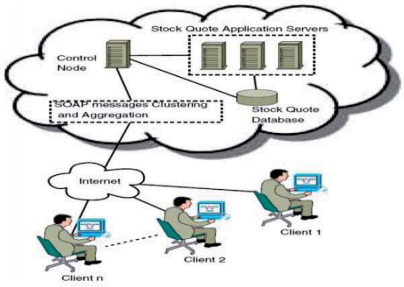


Figure 1: Clustering based aggregation model support Web services over the Internet

Proposed technique

In this section we focus on the main steps of the clustering of XML documents. The steps are as follows: (a) generating the vectors for the XML documents, (b) computing the similarity of the documents using their vectors, and (c) allocating the documents to their proper clusters.

a) Generating the XML Vectors

Any XML document in the dataset is modelled as a rooted tree. The XML tree has two kinds of nodes: (a) structure node and (b) content node. The structure refers to the nested tags (elements) that organise the content information while the content refers to the data values of the elements. We use depth-first search algorithm for traversing and indexing XML nodes level by level since all the nodes obtain a unique number as their index. To generate the XML vectors, we firstly generate the structure vector vs and content vector vc

We select m terms for the structure vector and n-m for the content vector, where m and n are usually application dependent and constrained by storage. For each term t in the structure or content vector, we use the tf.idf scheme to calculate the weight. The tf measures the frequency of the term t in the document denoted by tf (t, d) while the idf measures the importance of the term in the entire set of documents denoted by $idf(t) = log\frac{N}{df(t)}$

where df(t) presents the number of documents that contain t in the dataset and N is the total number of XML documents in the dataset. Formula 1 presents the tf.idf formula for a term t in the document d.

$$w_{t,d} = \mathrm{tf}(t, d) * idf(t) \quad (1)$$

After generating vs and vc, we combine these vectors to generate the XML vector of a document. This vector is used to measure the similarity score between the documents. The Eq.4presents the combination formula whereαis the tuning parameter which trades off between the importance of the structure and content terms of the document.

$$\vec{v} = (\alpha \times \vec{v}^{s}, (1-\alpha) \times \vec{v}^{c}) \quad (2)$$

For example purpose, assume we have 6 documents in the dataset. The XML vector for each document is generated by applying the combination formula presented in Eq.2 using the tuning parameter α=0.6. For each vector, there are 3 weights for the structural terms and 3 weights for the content terms. Table 1 presents the vectors for these documents.

Vectors	Structure			Content			Generating the vectors
	w_{s_1}	w_{s_2}	w_{s_3}	w_{c_1}	w_{c_2}	w_{c_3}	
v_1	0.3	0.6	0.2	0.4	0.9	0.3	(0.18,0.36,0.12,0.16,0.36,0.12)
v_2	0.2	0.3	0.3	0.6	0.3	0.2	(0.12,0.18,0.18,0.24,0.12,0.08)
v_3	0.3	0.6	0.2	0.6	0.9	0.3	(0.18,0.36,0.12,0.24,0.36,0.12)
v_4	0.3	0.6	0.4	0.4	0.9	0.3	(0.18,0.36,0.24,0.16,0.36,0.12)
v_5	0.2	0.2	0.3	0.6	0.3	0.2	(0.12,0.12,0.18,0.24,0.12,0.08)
v_6	0.2	0.5	0.2	0.3	0.8	0.3	(0.12,0.30,0.12,0.12,0.32,0.12)

Table 1: Vectors generation

b) Similarity Measurement

We use the data vectors to measure the similarity degree between their corresponding documents. The Euclidean distance measures the similarity between vectors that has several advantages in data clustering, such as simplicity and accuracy. Therefore, we use Eq.4 to calculate the Euclidean distance between a pair of XML vectors, for instancev1 andv2. In order to find the similar documents, we measure the distance between all the XML vector pairs. The output of this step is the similarity score for each vector with all other vectors.

$$\mathrm{dist}(v_1, v_2) = \sqrt{\sum_{i=1}^{n}(v_1 * w_i - v_2 * w_i)^2} \quad (4)$$

Example 1.The distance between the XML vectors in Table 1 are as follows:

$dist(\vec{v}_2,\vec{v}_5) = 0.06,\ dist(\vec{v}_1,\vec{v}_3) = 0.08,\ dist(\vec{v}_1,\vec{v}_6) = 0.1019, dist(\vec{v}_1,\vec{v}_4) = 0.12, dist(\vec{v}_3,\vec{v}_4) = 0.1442, dist(\vec{v}_3,\vec{v}_6) = 0.1523, dist(\vec{v}_4,\vec{v}_6) = 0.1574,\quad dist(\vec{v}_5,\vec{v}_6) = 0.2049,\ dist(\vec{v}_2,\vec{v}_6) = 0.2720, dist(\vec{v}_2,\vec{v}_3) = 0.3143,\quad dist(\vec{v}_1,\vec{v}_2) = 0.3243,\quad dist(\vec{v}_3,\vec{v}_5) = 0.3521, dist(\vec{v}_1,\vec{v}_5) = 0.3611.$

c) XML Vectors Distribution

After measuring the pairwise similarity between the XML vectors, we initialize the clusters for these vectors. To initialize the clusters, we start with sorting the pairwise distance between every two vectors, as shown in Example1.The pair with the minimum distance is first checked whether it is less than a given threshold δ. The two vectors of this pair are merged into a cluster if it is true. This process is carried out to all the other vector pairs (vi, vj) for which dist(vi, vj)< δ, in the order of increasing pairwise distance. After this first round, the pairwise distance between the centroids of every two clusters are computed and sorted in increasing order. Following the same process as the first round, the clusters are merged if their distance is less than δ. These rounds are continued until all the pairs satisfying the pairwise distance condition have been processed.

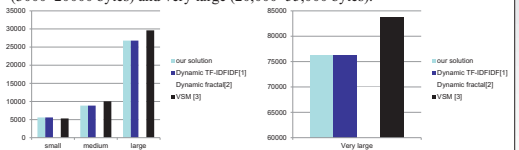
Considering Example 1 and δ=0.21055, the distance between v1 and v3, v6, and v4 is less than δ. While, the distance between v1andv2, v1and v5 is greater than δ. As a result, v2and v5have a high similarity and they will assign to the first clusterc1. While, v1, v3, v6, and v4 will assign to the second cluster c2.

d) Aggregation of XML messages

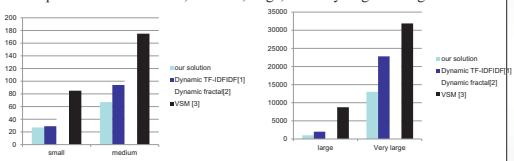
The Huffman encoding technique is adopted to generate the aggregated compact message by combining several similar XML textual expressions into one message. Huffman technique is well-known as lossless compression techniques that can delete the redundancies of letters by assigning binary codes for these letters.

Evaluation strategy

The dataset of SOAP messages used in this study is the same dataset that have been used in two previous studies (Al-Shammary and Khalil, 2011). The dataset are built based on WSDL (Web Service Description Language) at http://www.w3.org. It includes 160 XML documents divided into 4 groups according to thesize, each oneof them contains40messages. Thesegroups are described as follows: small (140–800 bytes), medium (800–3000 bytes), large (3000–20000 bytes) and very large (20,000–55,000 bytes).



Compression size for small, medium, large, and very large messages



Clustering time in Millisecond for small, medium, large, and very large messages

Results of clustering time and compression ratio

Dynamic clustering model has achieved better results in terms of compression ratio especially inXML Web messages with medium and large sizes in comparison with dynamic fractal clustering model and in medium, large and very large subsets with vector space model. Furthermore, dynamic frequency model has shown significant results in clustering time since comparison with VSM method and dynamic fractal model (seeFig. 12).Table 8shows the clustering time of dynamic frequency model, dynamic fractal and VSM in millisecond. Based on these results the clustering time of XML Web messages is based on two important factors: the amount of information in each subset of used dataset and the number of clusters generated. This emphasizes the applicability and sufficiency of applying new dynamic frequency model for clustering based compression and aggregation model for XML messages in real-world applications.

Conclusion

Two of the most important conclusions in this work,firstly, when the dynamic frequency model generates a large number of clusters based on similarities with small cluster size in this case, Huffman compression based aggregation tool would not be efficient to compress and combine several similar XML Web messages, and does not achieve high compression ratio as we noted in small and very large subsets. Secondly, when the dynamic frequency model generates a small number of clusters with large size of cluster in this case, Huffman compression based aggregation tool would be efficient to associate group of similar XML Web messages and achieve high compression ratio as we noted in medium and large subsets. Therefore, dynamic frequency based Huffman compression and aggregation model enables Web servers to generate one compact message that can be used by receivers (routers) to decompress the original messages. This model would notably increase the performance of Web services.

This improvement would support different types of Web situations such as short bandwidth states and generally the weakness of connectivity devices that are using format of XML Web messages and connected to Web server such as smartphones which are using AndroidManifest.xml to describe and manage its applications. For future work we use another term weighting scheme such as TF–ICF weighting schemes with Euclidean space method another similarity measure to estimate the distance between objects.