Web Service Selection based on Parallel Cluster Partitioning and Representative Skyline

Sahar Regaieg[1,2], Saloua Ben Yahia[1] and Samir Ben Ahmed[1,2]
[1]*LISI Laboratory, INSAT Institute, Carthage University, Tunis, Tunisia*
[2]*Faculty of Mathematical, Physical and Natural Sciences of Tunis (FST), Tunis El Manar University, Tunis, Tunisia*

Keywords: Web Service Composition, Quality of Service, Skyline, Representative Skyline, Cluster, Partition.

Abstract: Optimizing the composition of web services is a multi-criteria optimization problem that consists in selecting the best web services candidates from a set of services having the same functionalities but with different Quality of Service (QoS). In a large scale context, the huge number of web services leads to a great challenge: how to find the optimal web services composition while satisfying all the constraints within a reasonable execution time. Most of the solutions dealing with large scale systems propose a parallel Skyline phase performed on a partitioned data space to preselect the best web services candidates. The Global Skyline is computed after the consolidation of all the Local Skylines and, eventually the optimization algorithm is applied. However, these partitioning approaches are only based on pure geometric rules and do not classify the web services according to their real contribution to the optimal or sub-optimal solution search area. We will propose in this paper an intelligent partitioning approach using a cluster based algorithm combined with the representative Skyline.

1 INTRODUCTION

Web services Composition is a complex process that involves several activities such as discovery, composition, selection or execution. Generally, the composition is considered, initially, as an abstract workflow. The selection phase consists of looking for concrete instances (concrete services) that can replace abstract nodes. These services should meet QoS requirements and user preferences as much as possible.

Optimizing the Web Services (WS) Composition in terms of QoS consists in selecting the concrete WS for each task while satisfying the global constraints and the user preferences. The best overall QoS for the global composition according to a utility or fitness function will hence be reached (Zeng et al., 2003). As such, this optimization process consists in identifying the best services among a group of services with similar functions but different QoS.

The problem of QoS-based web service selection and composition has received a lot of attention during the last years. Local selection methods using techniques such as Simple Additive Weighting (SAW) were conducted to select services that ensure an optimal composition. However, local selection could not satisfy global constraints as it treats each service class individually. Zeng et al., (2003) tackled this problem using a global planning composition based on mixed Integer Programming technique for quality-driven dynamic selection.

However, the cost of this approach is very high in a large space. Linear programming methods are very effective when the size of the problem is small, but suffer from poor scalability due to the complexity of the applied search algorithms. All these contributions focused on service selection in a small space but for large scale systems, a further reduction of the number of candidates is required.

The rest of this paper is organized as follows: Section 2 presents the related work of the web services composition problem. In sections 3 and 4, we present the background of this work with a reminder of Skyline and Representative Skyline concepts. Sections 5 will describe the different steps of our proposed approach. Section 6 contains an experimental study of our approach and analysis of the obtained results and we will eventually conclude this paper.

Regaieg, S., Yahia, S. and Ahmed, S.
Web Service Selection based on Parallel Cluster Partitioning and Representative Skyline.
DOI: 10.5220/0006904106480655
In *Proceedings of the 13th International Conference on Software Technologies (ICSOFT 2018)*, pages 648-655
ISBN: 978-989-758-320-9

2 RELATED WORK

Many researchers have used Skyline requests to reduce the search space. The Skyline operator was first introduced by Börzönyi et al., (2001). Then many efficient skyline algorithms have been proposed. Algorithms like Block Nested Loop (BNL), Divide-and-Conquer (D&C), Sort Filter Skyline (SFS), Bitmap and Nearest Neighbor (NN) can process skyline query in the datasets without indexes.

These algorithms can eliminate the low-quality web services among large amounts of candidates and return a much smaller and a higher quality set to the user. But the obvious limitation of these studies is that they can only compute Skyline on one combination of QoS parameter.

However, in practice, different users may be interested in different combinations of QoS parameters and choose different preferences. In order to tackle this issue Yang et al., (2015) introduced the service SkyCube which consists in applying the Skyline on all possible combinations of QoS parameters. As it is computed previously in an off-line manner, using SkyCube method can speed up the response time in real-time web service selection. Unfortunately, the current SkyCube computation solutions suffer from the issue of dimension scalability. For this reason, the researchers moved towards the parallel Skyline approach.

All of the aforementioned scientific publications point the readers to one of the major difficulties when mashing up cloud services, namely: the large amount of services in the service pool to choose from, the diversity set of services representing different QoS, and the extended attribute dimensions to treat.

As such, some works introduced Skyline algorithms by incorporating the MapReduce paradigm to exploit distributed parallelism in a cloud mashup. A good evaluation of various MapReduce (MR) in Skyline processing was given by Zhang et al., (2015).

The grid-partitioning MapReduce is introduced by Zhang et al., (2011) and the MR-angular partitioning of Skyline space is studied in the research of Vlachou et al., (2008). With such partitioning methods, sometimes one partition may contain data that does not satisfy a specific request. Then, as an important variant of Skyline, the *K-Representative Skyline* is a useful tool if the size of the full Skyline is large (Bency, 2014). Bai et al., (2016) introduced the distance-based representative Skyline (k-DRS). This method is based on a 2-step approach to solve the k-DRS problem efficiently. Step 1 divides the full Skyline set into k clusters. In step 2, a point in each cluster is selected as the representative Skyline point. The k selected Skyline points consist of the k-DRS. So, the concept of Skyline used in the selection of web services according to their QoS applied to allow the reduction of the search space in the local phase, and subsequently, ensure a considerable time saving during the overall optimization process.

In this paper, we present a parallel Skyline service selection method designed to improve the efficiency of the existing partitioning approaches and propose a more intelligent and reduced search space for the web services selection. This approach is based on a cluster partitioning based method is employed with representative Skyline for web service selection.

3 SKYLINE

Skyline is a mechanism that acts as a filter in the search space that will select only the interesting points. Skyline can be formally defined as follows:

Given a set of points P in a space with q dimensions, Skyline points are the points who are not dominated by any other point in the search space according to those dimensions.

Given a data space D defined by a set of q dimensions $\{d_1, ..., d_q\}$ and a dataset P on D with cardinality n, a point $p \in P$ can be represented as $p = \{p_1, ..., p_q\}$ where p_i is a value on dimension d_i. Without loss of generality, let us assume that the value p_i in any dimension d_i is greater or equal to zero $(p_i \geq 0)$ and that for all dimensions the minimum values are more preferable

Definition 1 (Dominate). A point $p \in P$ is said to dominate another point $q \in P$, denoted as $p \prec q$, if (1) on every dimension $d_i \in D$, $p_i \leq q_i$; and (2) on at least one dimension $d_j \in D$, $p_j < q_j$.

Definition 2 (Skyline Point). The Skyline is a set of points $SKY_P \subseteq P$ which are not dominated by any other point in P. The points in SKY_P are called Skyline points (H. Köhler, J. Yang, and X. Zhou, 2011).

3.1 Parallel Skyline

Due to the complexity of the computation on the large datasets, parallel approaches have been studied

to calculate the Skyline. A data set can be divided into subsets of data according to a partitioning strategy. For each subset located on a server, a Local Skyline is computed. These Local Skylines are then consolidated on a single server and a Global skyline is eventually computed. In parallel Skyline solutions, an intuitive data partitioning scheme is based on random partitioning, in which objects in a partition are randomly selected. Since a dataset can be evenly divided across partitions by a random partitioning method, the number of Skyline objects in each partition is expected to be the same under a uniform data distribution.

This approach does not minimize the output of Local Skyline computation, which would significantly increase the communication cost between Local and merging Skyline computation components (Köhler et al., 2011).

A grid partitioning scheme that partitions the data space into multiple grids was employed in parallel and distributed computing environments (see figure 1). By utilizing the domination relation among partitions, objects in a partition that is dominated by another partition can be discarded without Skyline calculation.

To minimize the Local Skyline results, an angle based partitioning method maps datasets from a Cartesian coordinate space to a hyper spherical space and groups objects based on their angles to the origin (see figure 1)

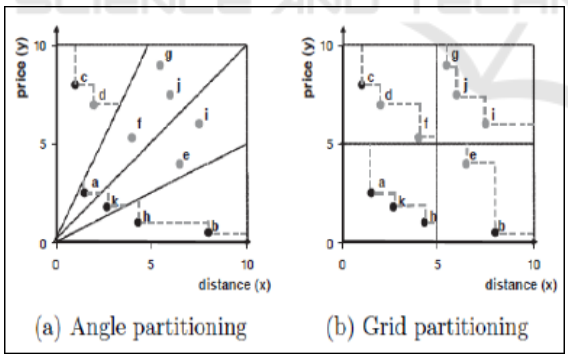

Figure 1: Partitioning example (Vlachou et al., 2008).

3.2 Problem Statement

It has been found that the angle-based partitioning method is effective in terms of execution time. However the main drawback is that some partitions do not contribute to the Global Skyline and do contain relevant points that will be considered as good candidates for optimal or sub-optimal solution of our problem. As such some servers are running inefficiently and Local Skyline results are sent over the network to the central server without really contributing to the final solution causing extra bandwidth consumption.

Moreover most of the existing contributions consider partitioning the data in a two-dimensional space while in reality we have *D*-dimensions each dimension for a specific QoS. We must also consider the dynamic aspect of the data since new web services are regularly added as candidates and we must position them on the right partition or the right server.

Even though the parallel computing Skylines became a major solution to recover results in reasonable response times, still in a large scale context, such methods are effective but have a certain number of disadvantages:

- Some partitions do not contribute neither to the Global Skyline nor to the optimization phase
- Data are inefficiently partitioned according to a pure geometrical aspect (angular, grid) without taking into account the efficiency of each partition according to the utility function.
- A large amount of data is generated dynamically,
- The attribute values of the objects do not cease to change.
- User preferences change continuously.
- Existence of irrelevant service areas.
- Existing partitioning methods are limited to two dimensions.
- The optimality aspect of the composition is not introduced into the Skyline calculation phase.

As aforementioned, partitioning method for Skyline computation have been widely studied in a large scale environment. However, these approaches have their own deficiencies as analyzed above.

In the next part, top-k representative skyline will be presented. This method will provide a new way to overcome some of the aforementioned deficiencies of ranking multi-criteria applications (Nguyen and Cao, 2015).

4 REPRESENTATIVE SKYLINE

In the service selection process, the more constraint conditions are considered the more services will be on the Skyline (Gang and Chunli, 2015). It is also difficult for users to identify the desirable services from Skyline.

In order to deal with this problem, the top-k approach can return fewer Skyline services which will significantly reduce the overall processing time.

However, this approach may potentially ignore the tradeoff of QoS required by users. In many candidate services, different tradeoff of QoS information gives user a summary of the performance distribution of services and offers an efficient way for user to identify desirable services.

The top-k representative Skyline is recently studied to offer few set Skyline services with different tradeoffs for the users.

The Representative Skyline can give a concise summary of the entire Skyline and allows the selection of the most appropriate web services for service composition with end-to-end constraints.

In the literature, the main challenge is how to identify a set of representative Skyline services that best represent the trade-offs between different qualities of service parameters so that a solution can be found that satisfies users constraints and also has a high service score (Bency, 2014). This is essentially a compromise on the number of representatives to choose.

A representative skyline is a set of k points that are considered as a representation of Full Skyline. Representative skyline fits very well the multi-criteria decision making. Up to now, Representative Skyline calculation algorithms are designed for Static data.

Mei Bai et al., (2016) proposed distance-based Representative Skyline query (DRS) which applies a distance metric to measure the "Representativeness" of a chosen set.

Given a set *K* with *k* Skyline points from the full Skyline set *S*, $Er(K, S) = max_{p \in S-K}\{min_{p' \in K} |p, p'|$ *where* $|p, p'|$ is the Euclidean distance between *p* and *p'*. The DRS query will select a set K, k Skyline points, that minimizes Er(K, S). The DRS performs well only if Skyline points follow a cluster distribution. Therefore, in data stream environments, the DRS cannot guarantee a high representativeness, because the dominance ability of the chosen set is neglected Skyline based on the significance and diversity.

5 PROPOSED APPROACH

In this section, we will explain our approach to compute the Skyline of a web services class in a parallel and representative way.

Our method consists of three steps. The main purpose of the first step is to realize an intelligent partitioning in order to compute the local skylines. We can consider two main classes of partitioning approaches for the Web Services DataSet. The first one is based on a pure geometrical approach like Grid, Angular and Random Partitioning (Vlachou et al., 2008). These techniques have already been applied on the problematic of Web Services Skyline computation. In this study, we considered a second class of clustering techniques widely used in the literature and inspired from the DataMining area like K-Means, hierarchical Clustering, Partitioning Around Medoids…. In this paper, we choose not to use the geometrical approach and to base our work on K-Means algorithm. This algorithm has been selected because of its low complexity as well as its good performances when applied on a big data set which is our case. The second Step allows the parallel Skyline computation since the Local Skyline is evaluated on each partition that is on each cluster identified during Step 1. The third step consists in the Representative Skyline calculation on each partition. The proposed solution is detailed in figure 2. We can notice that in this figure the cluster 2 (*C2*) is not a good candidate since the score of its centroïd is among the weakest ones. Indeed, the average of all scores is computed and all the clusters whose centroïds have a score better that this average are considered as being good candidates.

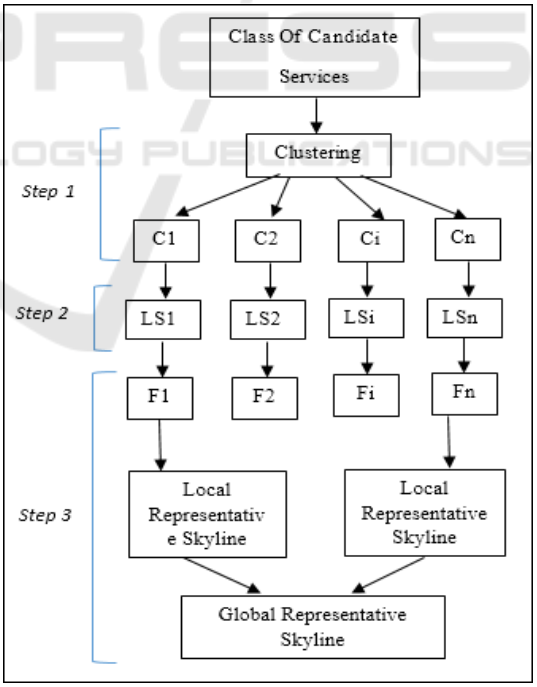

Figure 2: ¨Proposed approach.

5.1 Step 1: Clustering

In this step, we will focus on the clustering of services using the K-Means algorithm. Concretely,

this can be done as follows: suppose a set of WS of the same class of service (WS having the same functionality but different QoS).

A WS is characterized by a set of QoS criteria represented as an attribute vector. The goal is to group in the same cluster, WS considered similar.

K-Means is an iterative clustering algorithm. It starts with a set of *K* reference individuals randomly selected. The data are partitioned in K groups; an individual belongs to a group if the center of this cluster is the most close to him (in terms of distance). Updating of centroïds and assigning individuals to clusters of data are performed during the successive iterations. K-Means works to optimize the inertia criteria in order to make the data close to each other in the same cluster and widely separated among clusters.

To measure the similarity or dissimilarity between web services, we adopt the distance from Manhattan:

$$d_{i,j} = \sqrt[n]{|q_{i1} - q_{j1}|^n + |q_{i2} - q_{j2}|^n + .. + |q_{ip} - q_{jp}|^n} \quad (1)$$

Where *i, j* are two web services each defined by its *p* QoS; $i = (q_{i1}, q_{i2}, .., q_{ip})$, $j = (q_{j1}, q_{j2}, .., q_{jp})$ and $n=1$.

The intra-class inertia I_A is used to evaluate the heterogeneity within classes. The clustering is even better than its I_A is small.

$$I_A = \sum_{K=1}^{g} I_T(C_K) \quad (2)$$

When *P* is a data set consisting of *N* objects, $P = \{I_1, I_2, \ldots I_N\}$ and B_p their center then I_T of a Cluster *k*, $I_T(C_k)$ is given by:

$$I_T(C_K) = \frac{1}{N} \sum_{K=1}^{n} d(B_p, I_K) \quad (3)$$

Where N_K is the number of objects in the cluster *k* and *g* is the number of the generated clusters.

The inter-class inertia I_E evaluates the similarity between classes. More the I_E is great, the better clusters are separated, so heterogeneous and therefore the clustering is better.

$$\mathrm{I_E} = \sum_{k=1}^{g} d(B_p, B_{C_K}) \quad (4)$$

Where B_{Ck} is the center of the group *k* and $\mathrm{d}(\mathrm{B_p}, \mathrm{B_{C_k}})$ is the Manhattan distance as we defined previously.

5.2 Step 2: Parallel Skyline

All the calculation done in step 2 and step 3 is done in a parallel way. Indeed, each server is responsible of a cluster identified in step 1. In this step, we are interested in extracting Skyline services from each cluster. Thus, each server does the calculation of the Skyline relative to its cluster. To calculate the Skyline, we adopt the Block Nested Loop Skyline (Borzsonyi et al., 2001).

The BNL algorithm consists of comparing all the points of the search space two-by-two and returning as Skyline all points that are dominated by no other point in space. This algorithm fits well small and medium sized databases. This is why we adopt this algorithm in our approach since we have already partitioned the search space into several smaller sized search spaces.

5.3 Step 3: Representative Skyline

For each cluster identified in Step 1, a virtual point is created in the multidimensional space of the QoS, whose coordinates are calculated and this virtual point is considered as the centroïd of the cluster.

The representative Skyline services of each cluster are the Skyline services that have a higher score than the cluster centroid. This score is calculated by a F_{ij} which is also called the Utility Function.

Suppose there are *x* QoS attributes to be maximized and *y* QoS attributes to be minimized. The utility function for service S_{ij} is defined as follows

$$F_{ij} = \frac{1}{x+y} * \left(\sum_{\alpha=1}^{x} w_\alpha * \left(\frac{q_{ij}^{\alpha} * \mu^\alpha}{\delta^\alpha}\right) + \sum_{\beta=1}^{y} w_\beta * \left(1 - \frac{q_{ij}^{\beta} * \mu^\beta}{\delta^\beta}\right)\right) \quad (5)$$

Where w_α and w_β are the weights for each QoS attribute, $(0 < w_\alpha, w_\beta < 1)$, $\sum_{\alpha=1}^{x} w_\alpha + \sum_{\beta=1}^{y} w_\beta = 1$. μ and δ are the average and standard deviation of the QoS values for all candidates in a service class (M. Alrifai, D. Skoutas and T. Risse, 2010)..

In the utility function definition, all QoS attributes are weighted by their importance. They are also normalized by their averages and standard deviations so that the utility function will not be biased by any attribute with a large value. At the end of this step, each server gives the Local Representative Skyline services for the cluster in which it is responsible. The master server merged all Local Skylines to give a representative Global Skyline.

6 EXPERIMENTAL EVALUATION

6.1 Experimental Setup

In this section, we will study the performance of our method. In each experiment, a dataset of 150 web services characterized by 4 QoS is split to N partitions, so that each point is assigned to one partition. Then, for each partition the Local Skyline is computed. In our method, each server may use any Skyline algorithm, thus the choice of Skyline algorithm is not restrictive in general.

In a single experiment, we use the same algorithm for all partitions and all partitioning methods, in order to present comparable results. The algorithm employed is BNL. Each experimental setup is repeated 10 times and we report the average values. First, we performed a number of tests on the set of web services to find the optimal number of clusters.

Table 1 shows us the classification results of web services by the K-Means method. We know that a good classification is obtained when we find a minimum intra-class inertia value and a maximum inter-class inertia value. Test 3 is the one with the best results (intra-class inertia and inter-class inertia), hence the optimal number of cluster is equal to 4.

Table 1: Test for the validation of the optimal number of cluster.

Test number	k	Intra-class inertia	Inter-class inertia
Test 1	2	3.88	0.23
Test 2	3	2.22	14.75
Test 3	4	1.06	29.5
Test 4	5	1.62	18.96
Test 5	6	1.69	29.4

In the second experiment, we compare the compactness value (C) of these different values of k (A. Vlachou, C. Doulkeridis and Y. Kotidis, 2008). We define this metric as:

$$compactnes = 100 * avg_{1<i<N}\left(\frac{SKY_{pi} \cap SKY_p}{SKY_{pi}}\right) \quad (6)$$

Where SKY_{pi} is the Skyline of the partition i, SKY_p is the Skyline of the union of all Skyline partitions.

The compactness expresses how many points of the local result set of a partition, belong also to the global result set.

In the best case scenario, all data points that do not belong to the Global Skyline set, would be discarded during Local Skyline computation. Therefore, high values of compactness indicate that the local result sets are more compact and contain less redundant data points.

Table 2: Test of compactness value.

Method	Compactness
2-Means	38%
3-Means	63%
4-Means	88%
5-Means	69%
6-Means	76%

4-Means partitioning method had compactness 88%, showed remarkable compactness gain compared to remaining methods. This is an expected result since we have found best partitioning with 4-Means (presented in Table1).

In order to validate our approach, we have tested our approach on different data bases. We took 3 databases which are randomly generated. Each database is characterized by Number of Web Services, we refer to it by WS, and Number of QoS attributes, we refer to it by QoS.In order to experiment the impact of our method on WS and QoS. We generated the following databases:

DB1: WS=400 and QoS=4.

DB2: WS=800 and QoS=4.

DB3: WS=400 and QoS=8.

If the size of the database increases, in term of Number of Web Services, our Best Clustering will have better gain compared to remaining clustering methods.

In order to compare different databases, we will try to use the difference between the best compactness and the second best compactness: DeltaCompactness (DC).

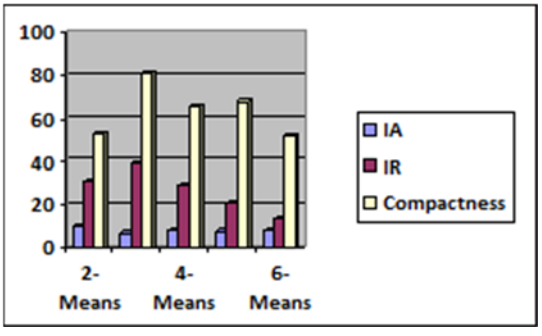

Figure 3: Data Base 1: WS=400 QoS=4.

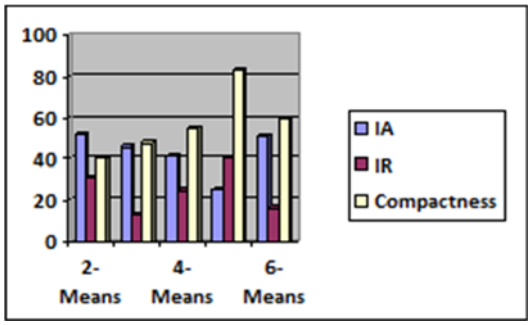

Figure 4: Data Base 2: WS=800 QoS=4.

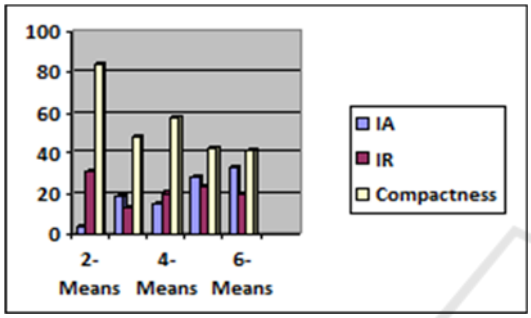

Figure 5: Data Base 3: WS=400 QoS=8.

So, when we take DB1, Figure 3, The DC is equal to 13%. And with DB 2 Figure 4, the DC is equal to 23%. High DC means that the best partitioning method is selected with less error probability. This proves that with our method we will select the best partitioning method in a Big Data context.

If we take DB1 (WS=400 and QoS=4) and DB3 (WS=400 and QoS=8), the first database showed DC=13% and the second database showed DC=26%. The more QoS attributes we have the more reliable our partitioning method selection will be. This result is a real plus in a Big Data context. Based on the different databases, Figure 3, 4 and 5 we may notice that compactness results and inertia results are correlated. In future work, we may rely on Inertia calculation when deciding about partitioning to be used.

6.2 Parallel Cluster Partitioning and Representative Skyline Vs Angle Partitioning Method

In this section, we are comparing the proposed method based on Parallel Cluster Partitioning and Representative Skyline with the Angle Partitioning method (Vlachou et al., 2008). We have selected the latter because it showed better performances compared to Grid Partitioning and Random Partitioning (Vlachou et al, 2008). We took the same data sets used in previous experiments. Then we have applied the Angular Partitioning with different space partitioning: from 2 partitions up to 6 partitions. The purpose is to compare the compactness metric obtained with the K-Means clustering and with the Angular method given the same number of partitions.

Angular partitioning showed a better compactness only when 2 partitions are considered. Otherwise the compactness is always better when the proposed clustering approach is used. Still, it is clear that the higher the number of partitions, the lower the compactness value is for both methods. This demonstrates that with angular partitioning, local skyline computation will be executed on irrelevant areas which will lead to extra computation time and extra bandwidth consumption since more local skyline points are send to the central server node while not contributing to the global skyline. With the cluster based method associated with representative skyline the risk of extra load and inefficient local skyline computation is lowered leading to an intelligent and efficient parallel approach.

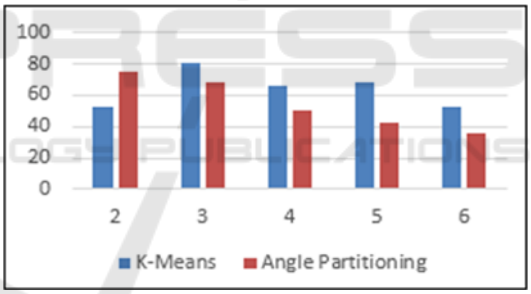

Figure 6: Data Base 1: WS=400 QoS=4.

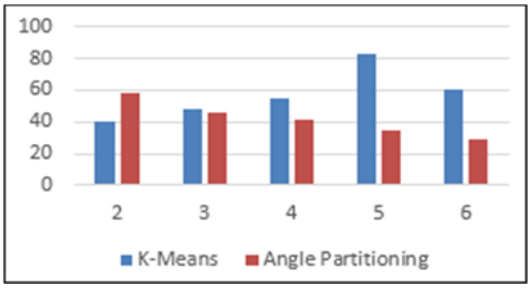

Figure 7: Data Base 2: WS=800 QoS=4.

With DB1 (see figure 6), we can for example observe that the compactness value with K-means clustering is 81% (K=3) while its value equals 66% for angular partitioning. For the second experiment using DB2 (see figure 7) we obtained mainly the

same observations. For example when using K-means the value of the compactness is C=83% given 5 partitions which is higher than the angular based method where the obtained compactness is C=37%.

7 CONCLUSION

In this work, we presented a new parallel approach for selecting web services based on clusters and representative Skyline. This method consists of three steps: Clustering, Parallel Skyline computation and eventually Representative Skyline evaluation.

Intelligent partitioning of the data set and parallelization allows us to minimize the computation time and the extra bandwidth load (better compactness) and eventually improves the Global Skyline quality since only the efficient points are gathered by Local Skylines. During this study we mainly focused on the compactness metric, we will conduct further evaluations in a future work. The experiments carried out on different data bases showed the effectiveness of our method. The proposed partitioning approach gave higher values of compactness which indicates that the local result sets are more compact and contain less redundant data points.

Our method which relies on representative skyline services with intelligent clustering appeared to be a more efficient approach. The optimization algorithm that will be executed after this first stage will then look for the optimal or sub-optimal solution that will satisfy the global QoS constraints on a more efficient data set of web services.

REFERENCES

A. Bency, 2014. A Study on Dynamic Skyline Queries. *International for research in applied science and engineering technology (IJRASET),* 2(11), pp. 2321-9653.

A. Vlachou, C. Doulkeridis and Y. Kotidis, 2008. Angle-based space partitioning for efficient parallel skyline computation. p. 227.

B. Zhang, S. Zhou and J. Guan, 2011. Adapting skyline computation to the MapReduce framework: Algorithms and experiments. p. 403–411.

D. Gang and C. Chunli, 2015. A novel approach to the selection of representative skyline web services.

H. Köhler, J. Yang, and X. Zhou, 2011. Efficient parallel skyline processing using hyperplane projections.

H. T. Huynh and J. Cao, 2015. Preference-Based Top-k Representative Skyline Queries on Uncertain Databases. *Springer International Publishing Switzerland,* p. 280–292.

H. T. Huynh Nguyen and J. Cao, 2015. Preference-Based Top-k Representative Skyline. *Springer International Publishing Switzerland,* p. 280–292.

J. Zhang, X. Jiang, W.-Shinn Ku and X. Qin, 2015. Efficient Parallel Skyline Evaluation using MapReduce Transactions on Parallel and Distributed Systems.

L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam and Q. Sheng, 2003. Quality-driven Web Service Composition.

M. Alrifai, D. Skoutas and T. Risse, 2010. Selecting Skyline Services for QoS-based Web Service Composition.

M. Bai, J. Xin, G. Wang and X. Wang, 2016. The Distance-Based Representative Skyline Calculation Using Unsupervised Extreme Learning Machines. *Learning and Optimization.*

S. Borzsonyi, D Kossmann and K. Stocker, 2001. The Skyline Operator.

Y. Yang, F. Dong and J. Luo, 2015. Computing Service Skycube for Web Service Selection.