Mashup Service Implementation on Multi-cloud Environment Using Map Reduction Approach

Venkateswara Rao Peddada
K L University
India
pvrao.pd@gmail.com

Ali Hussain Mohd
K L University
India
alihussain.phd@gmail.com

ABSTRACT: *Developing a mashup service is possible with integrating different functionalities and their data from different clouds. For example, one of the popular website twazzup.com contains some features from different sources like tweets from Twitter, images taken from Flicker, and new updates based on search terms we entered. Here we discussed advantage of using Enhanced Map reduced service in Multi-cloud Mashup service. We compare three popular MapReduced Skyline algorithms- MR-Angle[19], MR-Bitmap[37], MR-GPMRS[24] with the evaluated results taken from different sources. In this paper we tried to select best MapReduced skyline algorithm suitable for implementation of parallel processing to improving Mashup Services*.

Keywords: Service Selection - Skyline Query - MapReduce - QoS - Mashup Service

Received: 31 August 2017, Revised 5 October 2017, Accepted 14 October 2017

1. Introduction

Now a day's customer requirements are increasing rapidly, to cope up with this demand we required multi-cloud mashup services. By introducing service oriented architecture, introduction to web 2.0, and bigdata improved management [3, 20, and 28] demand for cloud mashup increased rapidly. In inter-cloud mashup services uses large data sets. In inter cloud mashup service selection of component service is difficult and it can be consider as NP-hard problem [1, 7].

To support Multi-Cloud Mashup services we can use MapReduce paradigm[11,13, 25, 36] and skyline operators[2, 4, 17, 21, 22]. Previous researches shows that the above two powerful tools enhance service oriented composition inner process for gaining High-Quality-of-service (HQoS) [5, 9, 23].

Our aim is to improving use of big data analytics by upgrading with cloud mashup services [16, 20]. Skyline helps to utilize qualified Web-services in a multilevel decision environment [14, 29, 32, and 33]. To improve quality of web services using improved Map-Reduce skyline query. We have to take care of two more things while solving problems here, they are

1. How to handle when skyline search space increases exponentially.

2. Providing High Quality of Services in cloud mashup services

In this paper we are concentrated, how to improve the skyline selection process, for that we adopted distribute parallelism in skyline selection. MapReduce is adopted to implement parallelism for this selection in mashup cloud platform.

We adopted some represent skyline services [30] to achieve QoS-assured services and to shorten the mashup process. Here our research is shown in following points:

1. Here we compared grid-portioning, angular-portioning and algorithm based on block-elimination and evaluated their advantages and disadvantages.

2. The partitioning based on block elimination [26] is updated to enhance the process of MapReduce [18].

3. Hadoop experiments are done to validate the mashup composition scheme. The Quality- of-Web Services (QoWS) points [14,15,34] are used to reporting results.

This paper contains details about our work mentioned in different sections (i.e. 2,3,4,5). The second section contains about cloud-mashup formation. Section three contains related work. Here in section four we consider the best three MapReduced skyline query processing method for Mashup composition. Section five contains Performance comparison for the three MapReduced skyline algorithms along with analysis.

At the last we summarized our work with remarks and given suggestions for future research direction.

2. About Cloud-Mashup Services

When two or more cloud services are combined to form a new service then that service is called as cloud-mashup. In this data and functionality required for web page is taken from different sources [3]. Expanding the cloud computing with other Internet applications or web services is the motivation of providing agility and scalability for composite cloud. The design motivation for integrated services is combining of different cloud services provided by different social networks and mobile platforms. One of the best cloud-mashup example is to combine and using Amazon AWS, Twitter, Drop Box and Google-services. Building Cloud-mashup is by choosing specific application program interface and data types.

To explain cloud-mashup we consider one example “Online Tour planner” it shows complex services involved. Here Six cloud-services are mapped with five task in fig.1.

When customer request for optimistic tour plan and our Task-one is to receive the request and identify the starting preferences. Then Task-two, Task-three and Task-four performing selecting best suitable transport services (it may be road, rail, bus, air service), suitable hotels and create a map for the trip. Fifth task integrate and analyze optimistic tour pack from the given data space.

Here different services are provided by different clouds. Here task one and five interact directly with customer. Remaining services are handle by different web sites using clouds. In this case the five cloud services together give-up mashup with combining their services. This workflow is connected by Direct Acyclic Graph shown in fig.1. Here to give customer optimistic tour package, the complexes services is changed to the cloud mashup.

Each task is handled by one or more cloud-based platforms. Various cloud functions supported large service space considered to select every candidate service. For example considered tourism service and deployed with cloud having faster response time and satisfactory matching result with high cost. Customers with more specifications need to choose different services by considering total weighting time and cost involved.

In real life applications available cloud service pool is very large. Waiting list would be long when same cloud service provider selected by multiple customers. Then it is difficult to predict waiting time and cost. So QoS is required to select a compound

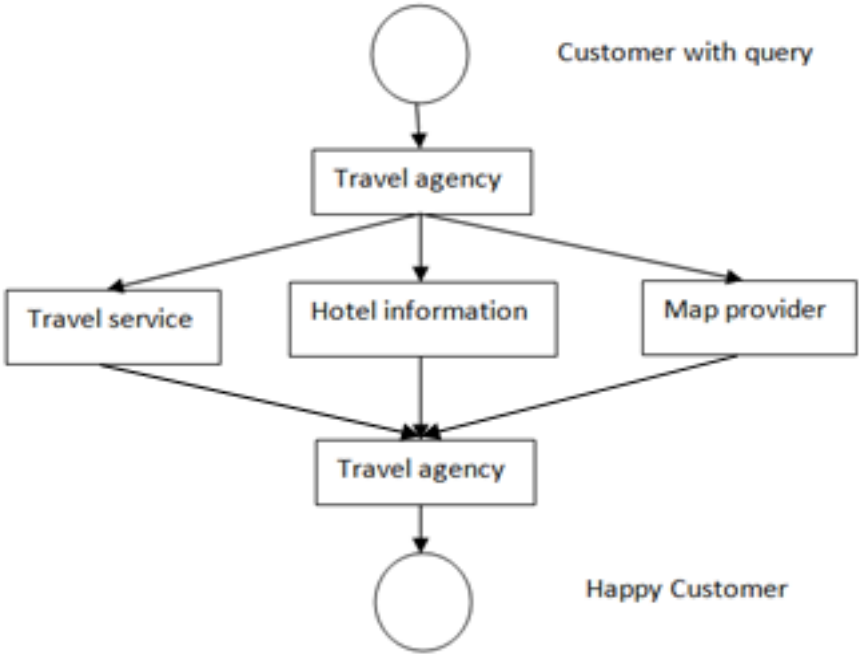


Figure 1. Tourist planner workflow on mashup with cloud-services

workflow of the cloud-service.

3. Related Work

3.1 Skyline Computation Process for Cloud-Mashup Service

The fast and optimized skyline query processing in inter-cloud applications can improve quality of composite web services. The following Figure 2.shows our work as three parts:

1. Skyline-Selection process

2. Pruning skyline

3. Service-Composition process.

We selected block elimination partitioning data space for skyline service. . The skyline produces large number of candidate services. Pruning Skyline method can be used to discover the best choice in skyline sub space, to accelerated subsequent QoS. Here we are trying to build composite cloud service with three components.

To reduce composite time in various skyline sectors, selection can be done through reduce redundancy.

Based on the specifications given by QoS and QoE mashup service bundle is given here. We will discuss these things in section 4.

2.2 Our Approach

Benatallah et al. in [3] provided thought of good cloud Mashup. There is one problem in cloud services, namely: select service from large pool.

Some of well-known skyline algorithms are MR-Angle, MR-Bitmap, MR-GPMRS. Here we incorporate MapReduced paradigm in skyline to introduce distributed parallelism. Zhang, given good MapReduced methods in skyline [36].

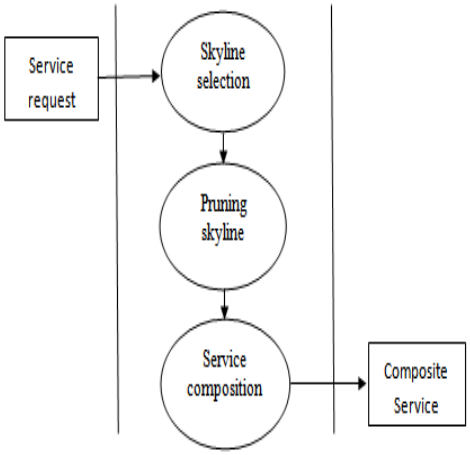


Figure 2. Skyline composition process Model

We made the following improvements:

(1) To shorten the Reduced time, we significantly merged the grid-based strategy

(2) In MR-angular we reduced 25% unnecessary domination tests.

Experimental results prove our claims. Here we combine Skyline, Mapreduce [2, 17] are shown in figure 2.Here Our work is to compare three Mapreduce skyline methods and make a decision which method is suits best for Mashup service cloud.

4. MapReduced Skyline Algorithm And Their Role in Service Discovery

4.1 Using MapReduce for Skyline Service Discovery

The idea is adding process between Map and Reduce, this one upgrade computing process capability along with improving performance in large-scale Skyline query processing. Here the approach is block-elimination method.

Mapping and *Reducing* process is shown in Fig.3 as three steps.

(1) The Mapping Process: In this main server based on QoS partitioning Server data points in to multiple blocks. For parallel processing data-blocks it will be send to slave servers.

(2) Computing Local-Skyline. *Here slave servers generate local syline*.

The Reduce Process: Here all the lacal skylines are integrated to generate global skylines. In Map Stage to implement parallelism will improve efficiency MapReduce Skyline process based on QoS.

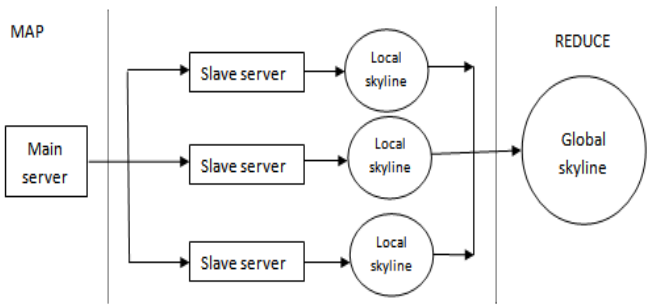


Figure 3. Method for selecting Services using MapReduce

The Efficiency of the Map is based on Data space partitioning. The aim is to avoid repetition, fit into memory through achieving load balancing. Dynamically New services are added when old services are dropped.

When number of candidate services becomes large then services then it become expensive. To overcome this, we introduced local skyline computation before the process of Reduce at step2. In step3 local skyline services are delivered to reduce. Processing large number of services is reduced in Reduce stage.

First service is data space is partitioned into groups and added it to the local skyline-computation. Next in second reduced stage compute Global skyline by integrating local skyline.

In previous works [8, 27], we have QoS problem, to solve it we adopted skyline method.

4.2 Popular MapReduce Skyline Methods
Below we have discussed about **MR-Angle, MR-Bitmap, MR-GPMRS.** Here MR stands for MapReduce.

1. MR-Angle. In this, angle based data space partitioning approach is employed.[24]

2. MR-Bitmap. In this, whole dataset forms the bit structure by mapped every point [6].

3. MR-GPMRS. MapReduce Grid Partitioning based Multiple Reducer Skyline Computation (MR-GPMRS) further divides local skylines and distributes them to multiple reducers that compute the global skyline in an independent and parallel manner [19].

Here different MapReduced skyline algorithms compared for best suitable in Multi-cloud Mashup services to apply distributed parallelism. To evaluate and compare the performance of three MapReduce based skyline service.

1. MR-Angle[10]
The MR-angular partitioning [24] is built on VDK method, which is developed by Vlachou, Doulkeridis and Kotidis (2008) . To make this method faster and space-efficient we modified in two levels: First Level, we made the given space into partitions using equal volume partitioning. In Second level, we are not transformed the reduce part into spherical data space.

This method contains major steps:
1. Transform the Cartesian coordinate ($x_1,\ldots, x_N$) into the local data space.

2. Based on angular coordinates [31], the data space is portioning into *M* sectors.

$$[\varphi_1^{i,\ldots} \varphi^{i}_{D-1}]$$

Partitioning the data space into sectors by differentiating sectors with angles in the range [0, Π/8), [Π/8, Π/4), [Π/4, 3Π/8) and [3Π/8, Π/2) in a 2-dimensional.

To deliver the services *s* to corresponding slave server, we will check the service point *s* belonging to the range [$tan\varphi_1 1,\ldots, tan\varphi_{D-1}$] and identify the sectors. For mapping process hyper spherical coordinates are used. The local skyline used Cartesian coordinate.

2. MR-Bitmap[37]
In Bitmap [6], Bitmap structure forms with whole collection of dataset, each dataset is a mapping of every point. Here data records in all dimensions, ignoring their magnitude is consider as bitmap structures. Here lightweight weight bitwise comparison is performed

Here each record compared with other records for examining the records. It is possible to implement progressive skyline with bitmap. When a point is recognized as member of skyline it return to the user. Bitmap Spatial cost is very high. The size of Bitmap structure is $Sb = N \times \Sigma^{d}_{i=0} Ki$ (bit), let k_i be the number of distinct values. If $K_i = N$(*N* is size of dataset) and worst case dimension *i*, then dN^2 is the space cost. To avoid frequent memory access for I/O always structure must be kept in memory.

The MR-Bitmap performs two levels. First level we built bitmap structure, in second level, based on bitmap structure we have to examine each point. For example R_n is the number. Each reducer needs to read bitmap into memory. When R_n is it too large, then there is chance of degrading performance. On the other side Degree of parallelism is also too low for R_n is too small. In practice there is R_n is set to the number of cluster nodes. Checking whether the slice in memory or not based on accessing bit-slices. The reason to store bitmap in a column manner not to read from disk.

3. MR-GPMRS[19]

This is Grid partitioning based Multiple-Reducer Skyline computation algorithm. This partition scheme allows for identifying independent groups. To obtain the skyline sets these sets can be processed independently. The MR-GPMRS algorithm takes a subset $R_i f R$ and the global bit string BS_R as input, and process each tuple t in R_i. First it finds the partition p_j that contains t. Tuple t is further processed by p_j's corresponding bit in the bit string is 1. Pj corresponding local skyline SPJ is either initialized as $\{t\}$ or updated with respect to t. The updation is done like BNL algorithm. Here t is added to local skyline if t is not dominated by current local skyline tuples that turn out to be dominated by t.

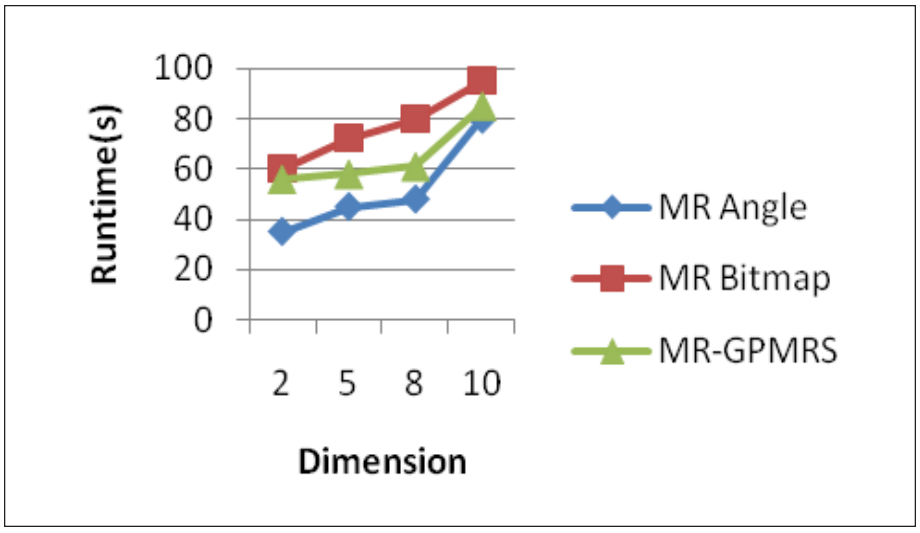
After processed all tuples in R_i, variable S contains a series of local skylines (S_ps) each of which corresponds to an unpruned partition p that contains tuple(s) in R_i. After that, the bitstring BS_R is used to generate the independent groups. Independent group across mappers would cause wrong skyline results on reducers. The given algorithm distributes the local skylines to corresponding reducers according to the independent groups. Algorithm sends independent groups to reducers in a round-robin way.

The Reduce step MR-GPMRS receives the local skylines $S_1, S_2, \ldots, S_m$ from all mappers and merge them into the correct global skyline. This reducer organizes local skylines with respect to a partitions in set P. The workload of singe reducer is distributed to multiple reducers independently. This independency and parallelism significantly improves the overall skyline computation efficiency.

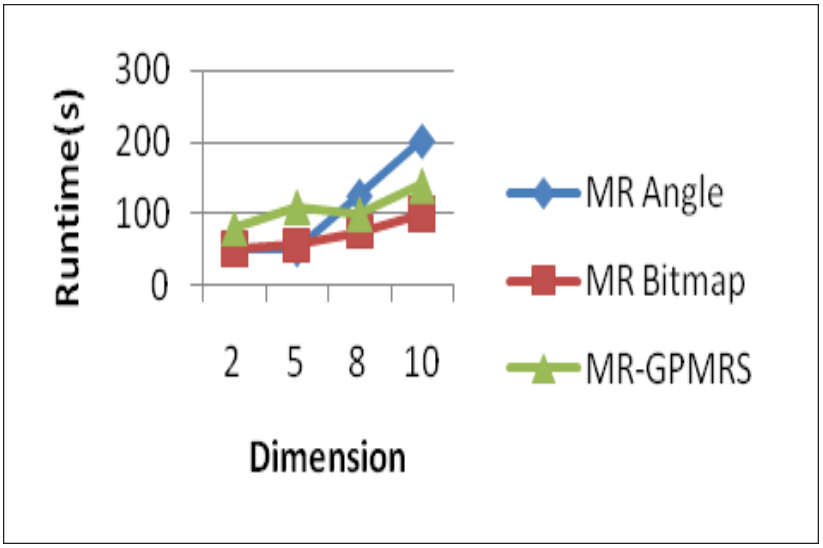
Here three different MapReduced skyline algorithms compared for best suitable in Multi-cloud Mashup services. The Evaluation process results are taken from different research sources [19], [37], [24]. After Comparing above three methods with graph, it is understood that MRGPMRS is best suitable for Multi-cloud Mashup services to apply distributed parallelism.

5. Performance Comparison With Respect to Dimensionality

We first interpret the effect of data dimensionality on the MapReduce skyline algorithms. For either data distribution, we use one cardinality ($1x10^5$), and change the Dimensionality from 2 to 10. The figure 4 shows the results of runtime of all algorithms 4.a and 4. b for independent and anti-correlated distributions respectively.



(a) Runtime of Independent Dataset



(b) Runtime of Anti-related dataset

Figure 4. Performance of MR-Angle[10], MR-Bitmap[37] and MR-GPMRS[17] based on different datasets

For independent data distribution, MR-Angle performs the best according to the results shown in Figure 4.a When the dimensionality is low (from 2 to 5), MR-Angle performs slightly high compare to alternatives, as shown in Figure 4(a). When the dimensionality increases, MR-GPMRS performs very steadily, whereas MR-Angle as increased highly. In particular, comparing the dimensionality is from 8 to 10, MR-GPMRS and MR-Angle perform comparably and both are significantly same. Finally MR-GPMRS runtime is in between MR-Bitmap and MR-Angle.

For Anti-related data distribution, MR-Angle and MR-GPMRS performs well between the Dimensionality from 2 to 5. After the Dimensionality 5 MR-Angle Runtime is increased drastically, but MR-Bitmap shown slow and study increment in runtime. When we discuss about MR-GPMRS shows study increment in runtime when Dimensions increased.

So our observation is Figure 4 (a) shows MR-Angle is best in performance and Figure 4 (b) shows MR-Bitmap is best in performance. Any one of these two shown best performance in both datasets, but when we saw MR-GPMRS shows average runtime in both datasets. Not only that MR-GPMRS avoid the bottleneck by utilizing the bitstring to partition the final skyline computation among multiple reducers.

This work studies how to make good use of the parallelism of MapReduce for skyline computation, whereas the local skyline computation on a single node is not the research focus. It is still interesting to optimize the local skyline computations and explore how such optimizations would affect the overall performance in the context of MapReduce.

6. Conclusion

In this paper we tried to derive best suitable MapReduced skyline algorithm for Multi-cloud mashup services. For this we have taken three best known MapReduced skyline algorithms: **MR-Angle, MR-Bitmap, MR-GPMRS.** Here we have taken resultant values for these algorithm from different sources[10][17][37] and later compared by plotting graph. This comparison done on Independent and Anticorrelated dataset, based on dimensions and processing time. Finally it is proven that MR-GPMRS is best suitable for implementation of MapReduced skyline computation in multi-cloud mashup services. Scope of this paper is to test and prove this proposal with real time dataset.

References

[1] Al-Masri, E., Mahmoud, Q. (2007). QoS-based Discovery and Ranking of Web Services, *ICCCN*, p. 529-534.

[2] Alrifai, M., Skoutas, D., Risse, T. (2010). Selecting Skyline Services for QoS-based Web Service Composition, *In*: Int'lConf. on World Wide Web (WWW) 11-20.

[3] Benatallah, B., Casati, F., Daniel, F., Yu, J. (2009). Mashups, SaaS and Cloud Computing, slide presentation, *In*: *IEEE Int'l Conf.on Data Engineering* (ICDE).

[4] Borzsonyi, S., Kossmann, D., Stocker, K. (2001). The Skyline Operator, *Int'l Conf. on Data Engineering*, p 421-430.

[5] Boutaba, R. (2005). QoS-aware Service Composition in Large Scale Multi-domain Networks, 2005 9th *IFIP/IEEE International Symposium on Integrated Network Management*, 2005, p. 397–410.

[6] Tan, K. L., Eng, P. K., Ooi, B. C. (2001). Efficient progressive Skyline computation. *In*: Proceedings of VLDB, p. 301–310.

[7] Candan, K. S., Nagarkar, P., Nagendra, M., Yu, R. (2011).RanKloud?: A Scalable Ranked Query Processing Framework on Hadoop, *In*: Proceedings of the 14th International Conference on *Extending Database Technology*, p. 574-577.

[8] Cao, J., Hwang, K., Li, K., Zomaya, A. (2013). Optimal Multiserver Configuration for Profit Maximization in Cloud Computing, *IEEE Trans. Parallel and Distributed Systems*, July.

[9] Cardellini, V., Casalicchio, E., Grassi, V., Presti, F. L. Flow-based Service Selection for Web Service Composition.

[10] Chen, L., Hwang, K., Wu, J. (2012). MapReduce skyline query processing with a new angular partitioning approach. *In:* IPDPS Workshops & PhD Forum, p. 2262- 2270.

[11] Chen, L., Wu, J., Deng, S., Li, Y. (2010). Service Recommendation: Similarity-based Representative Skyline, IEEE World Congress on Services, p. 360-366.

[12] Dean, J., Ghemawat. S. (2008). MapReduce: Simplified Data Processing on Large Clusters, *Communications of the ACM*, 51, (1) 107-113.

[13] Dixit, V. (2010). Cloud Mashup: Agility and Scalability, EE 657 Final Project Report, University of Southern California, May.

[14] Doulkeridis, C., Nørvåg, K. (2013). A survey of large-scale analytical query processing in MapReduce, *The VLDB Journal*, 123, June

[15] Eyhab, A., Mahmoud. Q. H. (2007). Discovering the Best Web Service, International World Wide Web Conference, p. 1257-12

[16] Fei, X., Lu, S., Lin, C. (2009). A MapReduce-Enabled Scientific Workflow Composition Framework, *In*: International Conference on Web Services, 2009, p. 663-670.

[17] Fox, G. (2010). MPI and MapReduce in Clusters, Clouds, and Grids for Scientific Computing, *In*: Proceedings of CCGSC, September 8, 2010.

[18] Han, H., Jung, H., Kim, S., Yeom, H. (2009). A Skyline Method to the Matchmaking Web Service, *Int'l Symp.on Cluster Computing and the Grid* (CCGrid).

[19] Mullesgaard, K., Pederseny, J. L., Lu, H., Zhou, Y. (2014). Efficient Skyline Computation in MapReduce. *In*: Proceedings of the 17th International Conference on Extending Database Technology, 2014, p. 37-48.

[20] Hwang, K., Bai, X., Shi, Y., Li, M.Y., Chen, W. G., Wu, Y. W. (2015). Cloud Performance Modeling with Benchmark Evaluation of Elastic Scaling Strategies, IEEE Trans. On Parallel and Distributed Systems, accepted to appear with on-line publication on March.

[21] Hwang, K., Fox, G., Dongarra, J. (2012). Distributed and Cloud Computing: from Parallel Processing to the Internet of Things, Morgan Kaufmann.

[22] Kossmann, D., Ramsak, F. (2002). Shooting Stars in the Sky: an Online Algorithm for Skyline Queries, *In*: *Int'l Conf. on Very Large Data Base*, p 275-286.

[23] Lin, X., Yuan, Y., Zhang, Q., Zhang, Y. (2007). Selecting Atars: The k Most Representative Skyline Operators, *In*: *Int'l Conf. on Data Engineering*, 2007, p 86-95.

[24] Wu, Jian., Chen, Liang., Yu, Qi., Kuang, Li., Wang, Yilun., Wu, Zhaohui. Selecting skyline services for QoS-aware composition by upgrading MapReduce paradigm

[25] Nahrstedt, R. N., Chang., Ward, C. (2003). QoS-assured service composition in managed service overlay networks, *In*: Proc. 23rd *Int'l Conf.on Distributed Computing Systems*. p. 194–201.

[26] Zeng, L., Benatallah, B., Ngu, A. H. H.,Dumas, M., Kalagnanam, J., Chang, H. (2004). QoS-aware middleware for Web Services Composition, *IEEE Trans. on Software Engineering*, 30 (5) 311–327 May.

[27] Roy, T., Wolsey, L. (1987). Solving Mixed Integer Programming Problems using Automatic Reformation, *Journal of Operation Research*, 35 (1) January.

[28] Shen, H., Hwang, K. (2012). Locality-Preserving Clustering and Discovery of Resources in Wide-Area Computational Grid, *IEEE Trans. on Computers*, April, p. 458-473.

[29] Taher, Y., Benslimane, D., Fauvet, M., Maamar, Z. (2006). Towards a Method for Web services Substitution, *Int'l Database Engineering and Apps. Symp*. p.166-173

[30] Tan, K., Eng, P., Ooi, B. (2001). Efficient Progressive Skyline Computation, *Int'l Conf. on Very Large Data Base*, p. 301-310.

[31] Tao, Y., Ding, L., Liu, X., Pei, J. (2009). Distance-based Representative Skyline, International Conference on Data Engineering, p. 892-903.

[32] Vlachou, A., Doulkeridis, C., Kotidis, Y. (2008). Angle-based Space Partitioning for Efficient Parallel Skyline Computation, *In*: Proceedings of the 2008 ACM SIGMOD *Int'l Conf.on Management of data* , p. 227, 2008..

[33] Wang, S., Ooi, B. C., Tung, A. K. H., Xu, L. (2007). Efficient Skyline Query Processing on Peer-to-Peer Networks, *Int'l Conf. on Data Engineering*, p. 1126-1135.

[34] Wendg, M. (2007). A Multimedia Social Networking Community for Mobile Devices, *Technical Report, Tisch School of The art*, New York University .

[35] Wu, Q., Bouguettaya, A. (2010). Marketing Service Skyline from Uncertain QoWS, *IEEE Transaction on Service Computing*, 3 (1) 16-29.

[36] Yu, T., Zhang, Y., Lin, K. J. (2007). Efficient Algorithms for Web Services Selection with end-to-end QoS Constraints, *ACM Trans. on the Web*, 1 (1) 1-26.

[37] Zhang, B., Zhou, S., Guan, J. (2011). Adapting Skyline Computation to the MapReduce Framework: Algorithms and Experiments, DASFAA Workshop, CS Lecture Note 6637, *Springer-Verlag, Berlin*, 2011, p. 403-411.