Data Integration On Multi-Cloud Environments

Daniel A. S. Carvalho
(Supervised by Chirine Ghedira-Guegan)
Université Jean Moulin Lyon 3
Centre de Recherche Magellan - IAE
Lyon, France
daniel.carvalho@univ-lyon3.fr

ABSTRACT

In the traditional databases theory, data integration is a problem of merging data from data sources in order to provide a unified view of this data to the user. This PhD project address data integration in multi-cloud environments. Current data integration systems implies consuming data from data services deployed in cloud contexts and integrating the results. The project takes a new angle of the problem by proposing an SLA-based approach, which given a user query and her integration preferences, to match the user preferences (such as whether she accepts to pay for the data, its provenance, freshness and how much she is ready to pay for the integration, among others) with the services' SLA while delivering the results. The objective is to enhance the quality on data integration taking into consideration the economic model imposed by the cloud. Our preliminary experiments have shown that quality can be enhanced and the cost of the integration results can be minimized by using our approach.

1. INTRODUCTION

In recent years, the cloud have been the most popular deployment environment for data integration [5]. Data integration has evolved with the emergence of data services that deliver data under different quality conditions related to data freshness, cost, reliability, availability, among others. Data are produced continuously and on demand in huge quantities and sometimes with few associated meta-data, which makes the integration process more challenging. Some approaches express data integration as a service composition problem in which given a query the objective is to lookup and compose data services that can contribute to produce a result. Finding the best service composition that can answer a query can be computationally costly. Furthermore, executing the composition can lead to retrieve and process data collections that can require important memory, storage and computing resources.

Data integration has been addressed in the service-oriented architectures [6, 9, 14, 15]. [6] proposed a rewriting method based on SPARQL for RDF data integration. The work focused on avoiding ineffective data integration by solving the entity co-reference problem. [9] introduced SODIM, a system which combines data integration, service-oriented architecture and MapReduce distributed processing. [15] presented a solution focusing on data privacy in order to integrate data. [14] developed an inter-cloud data integration system that considers a trade-off between users' privacy requirements and the cost for protecting and processing data. According to the users' requirements, the query is created and executed meeting privacy and cost constraints. The novelty of these approaches is that they perform data integration in service oriented contexts, particularly considering data services. They also take into consideration the requirement of computing resources for integrating data. Thus, they exploit parallel settings for implementation costly data integration processes. However, they are manly focused on performance and privacy aspects putting aside other users' integration requirements such as data provenance, data integrity, confidentiality, reliability, availability, whether she wants to use free services, among others. Moreover, even with the cloud on-demand resources provisioning (which implies an associated cost), the user is limited to her cloud subscription and maximum budget she is ready to pay for her desired integration. The economic cost should be taken into consideration.

As highlighted in our previous work [5], we strongly believe that service level agreements (SLA) can be used in order to cover the limitations and enhance the quality in the current data integration solutions. Research contributions SLA in cloud computing mainly concern (i) the negotiation phase (step in which the contracts are established between customers and providers) and (ii) monitoring and allocation of cloud resources to detect and avoid SLA violations. Yet, to the best of our knowledge, we have not find works proposing SLA-based approach for data integration in a multi-cloud environment. Similaraly to our idea, [12] proposed a SLA-based data integration model for grid environments. The approach uses SLAs to define database resources and evaluated them in terms of processing cost, amount of data and price of using the grid. In addition, a matching algorithm is proposed to produce query plans using the selected resources. The most appropriated solutions based on these QoS are selected as final results. Our work differs from [12] in some aspects:

- Data is delivered as *data services* in a multi-cloud context. *Data services* and *cloud providers* export their SLA defining the quality conditions under which the service is delivered.

- SLAs are not limited only to describe the cost and amount of data, but also data quality aspects such its provenance, privacy, confidentiality, freshness, and service's delivery aspects such as response time, availability, reliability, among others.

- Users are able to express queries associating quality integration requirements to them. Then, the service selection and rewriting process in terms of service compositions are guided by the user's requirements and the SLAs exported by *data services* and *cloud providers*.

In this context, current SLA models are not sufficient to cover the data integration requirements and multi-cloud context. Thus, we are current working on new models to tackle these aspects. In summary, the objectives of this PhD project contribute as following: (1) we design a new SLA model for data integration; (2) propose data integration approach adapted to the vision of the economic model of the cloud. The originality of our approach consists in guiding the entire data integration solution taking into account (i) user preferences statements; (ii) SLA contracts exported by different cloud providers; and (iii) several QoS measures associated to data collections properties (for instance, trust, privacy, economic cost); and (3) validation of our approach in a multi-cloud scenario.

2. CONTRIBUTIONS

Escrever um paragrafo resumindo as contribuicoes pretendidas no projeto e depois em subsecoes falar de alguns pontos principais... approach, model, schema, algoritmo... fazer essa imagem da abordagem no inkscape...

2.1 New vision of Data Integration

Let us assume the following medical scenario in which users are able to retrieve and integrate data concerning (i) *patients that were infected by a disease;* (ii) *regions most affected by a disease in specific region*; (iii) *patients' personal information*; and (iv) *patients' DNA information.*

The figure 1 illustrates our vision of data integration. Data is delivered as *data services* deployed in a multi-cloud context. Each *data service* and cloud export their SLA specifying the level of services, the available services and their cost, and the access conditions the user can expect. Therefore, given a user query, her integration quality requirements and her cloud subscription, it is rewritten in terms of cloud services (*data services* and *data processing services*) composition that fulfill the integration requirements and deliver the expected results to the user.

Taking into consideration the medical scenario, Doctor *Marcel*'s research is interested in the type of people suffering of *flu* in the Europe. He has at his disposal a set of cloud services delivered by different clouds. To reach his needs, he wants to query the personal information and DNA information from patients that were infected by *flu*, using services with availability higher than 98%, price per call less than 0.2$ and integration total cost less than 5$. This new context brings challenges to data integration, such as:

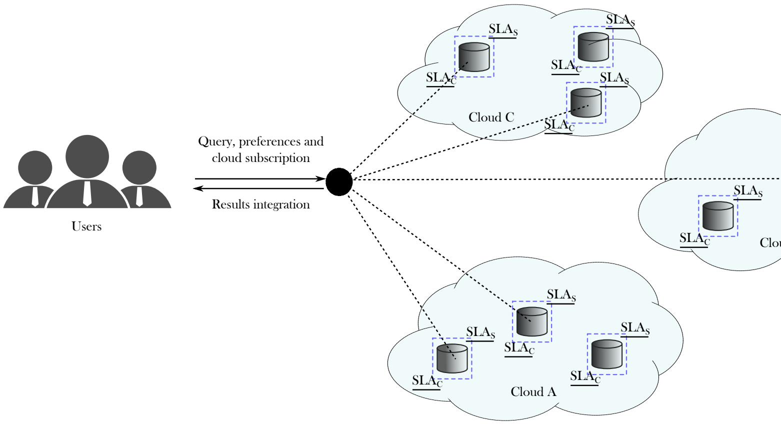

Figure 1: Data integration scenario

- **Performance**. In the multi-cloud, the amount of available services is high. Consequently, the processing time necessary to produce service compositions (given the high number of services) and to execute them is also high.

- **Economic model**. Even with the possibility of having an unlimited access to resources, the user is limited to the resources she has contracted and to the budget she is ready to pay for. Thus, it is necessary to produce the rewritings satisfying the user integration requirements.

- **Quality**. Part of the rewritings produced to the user query does not satisfy her quality requirements concerning privacy, data provenance, cost, among others. Producing these rewritings implies increasing time processing and cost.

- **Matching problem**. While producing the rewritings, it is necessary to match the user requirements with the different SLAs exported by the cloud providers and cloud services. In the multi-cloud, cloud providers and cloud services export SLAs with different semantics and structure that makes the matching SLA and user requirement challenging. In addition, it also deals with incompatibilities of SLAs.

- **Reuse**. Rewriting and executing the user query is computationally costly in terms of processing time and economic cost. Thus, it is necessary to propose a manner of reusing previous integration in order to save time and money, but also meeting the user expectations.

2.2 Approach

Thus, motivated by these challenges, a SLA guided data integration approach can be divided in four steps. Given a user query, a set of user preferences associated to it, cloud providers and cloud services:
SLA derivation. This step creates an *integrated SLA* that includes a set of measures corresponding to the user preferences. The *integrated SLA* guides the query evaluation, and

the way results are computed and delivered.

Filtering data services. The *integrated SLA* is used (i) to filter previous SLA derived for a similar request in order to reuse previous results; or (ii) to filter possible data services that can be used for answering the query.

Query rewriting. Given a set of data services that can potentially provide data for integrating the query result, a set of service compositions is generated according to the *integrated SLA* and the agreed SLA of each data service.

Integrating a query result. The service compositions are executed in one or several clouds where the user has a subscription. The execution cost of service compositions must fulfill the *integrated SLA* (that expresses user requirements). Here, the clouds resources needed to execute the composition and how to use them is decided taking in consideration the economic cost determined by the data to be transferred, the number of external calls to services, data storage and delivery cost.

Although *the SLA derivation* is the big challenge while dealing with SLAs and particularly for adding quality dimensions to data integration, the focus in this paper is our query rewriting algorithm which deals with user preferences and SLAs exported by different cloud providers and data services. Here, we are assuming that there is a mechanism responsible to extract the services' quality aspects from SLA, and to provide this information as input to the algorithm. The figure 2 illustrates the structure of SLA and its measures that are considered in the approach we will detail in the next section.

2.3 SLA model

2.4 Query rewriting algorithm

To serve as a proof of concept to our approach, we intend to develop a query rewriting algorithm which is guided by users' integration requirements and service level agreements exported by different data services and cloud providers. Query rewriting is an important issue in data integration. In cloud computing, researches have refereed to it as a service composition problem in which given a query the objective is to lookup and compose data services that can contribute to produce a result. [2] proposed a query rewriting approach which processes queries on data provider services. [4] introduced a service composition framework to answer preference queries. Two algorithms inspired on [2] are presented to rank the best rewritings based on previously computed scores. [1] extended [7] and presented an refinement algorithm that produces and order rewritings according to user preferences and scores. In general, these works share the same performance problem depending on the size of the query and on the number of available services. Furthermore, they do not take into consideration user's integration requirements what can lead to produce rewritings that are not satisfactory to the user in terms of quality requirements and cost. Currently, we have formalized and developed a rewriting algorithm that considers user preferences and services' quality aspects while selecting services and producing rewritings.

3. PRELIMINARY RESULTS

colocar parte dos resultados do adbis + um grafico do custo da reescrita.. This section describes the experiments performed as proof of concept to the algorithm. The Rhone prototype is implemented in Java. It includes 15 java classes in which 14 of them model the basic concepts (*query*, *abstract services*, *concrete services*, etc), and 1 responsible to implement the core of the algorithm.

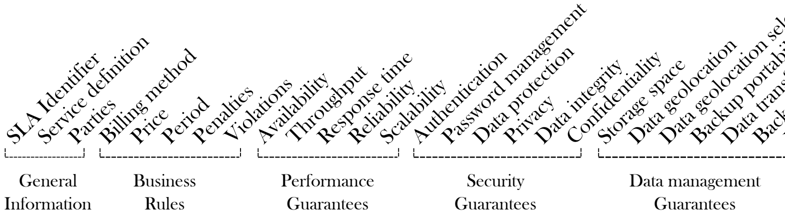

Figure 2: Cloud SLA

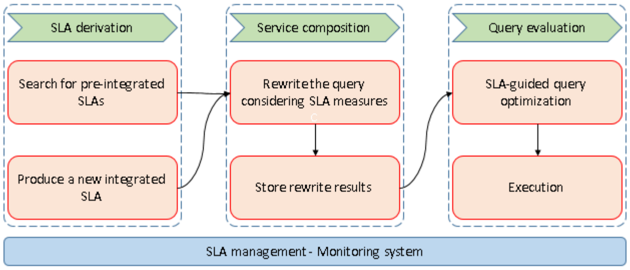
SLA derivation
Search for pre-integrated SLAs
Produce a new integrated SLA
Service composition
Rewrite the query considering SLA measures
Store rewrite results
Query evaluation
SLA-guided query optimization
Execution
SLA management - Monitoring system

Figure 3: A sample black and white graphic (.pdf format).

Currently, our approach runs in a controlled environment. Different experiments were produced to analyze the algorithm's behavior. We will present two experiments: *experiment 1* and *experiment 2*. The service registry used has 100 concrete services. In each experiment, there are a set of tests in which the number of concrete services varies from 5 until to reach 100.

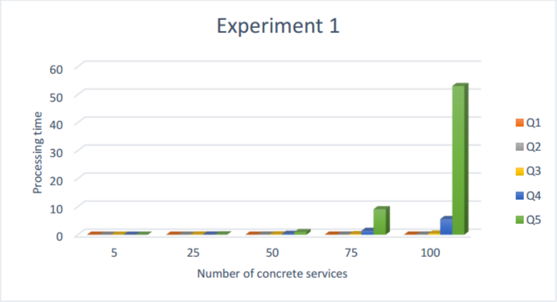

Figure 4: Query rewriting evaluation.

The *experiment 2* (figure 5) presents the results while testing the algorithm in the presence of user preferences and services' quality aspects extracted from SLAs. The difference between *Test 1* and *Test 2* concerns the way services are selected and the query is rewritten. Once *Test 1* do not consider quality measures as any other existing query rewriting approach, *Test 2* uses the user preferences statements and services' quality aspects to guide the service selection and query rewriting. Both include queries with six abstract services and quality requirements concerning availability, response time, price per call and integration cost (total cost). The figure 5 shows our results.

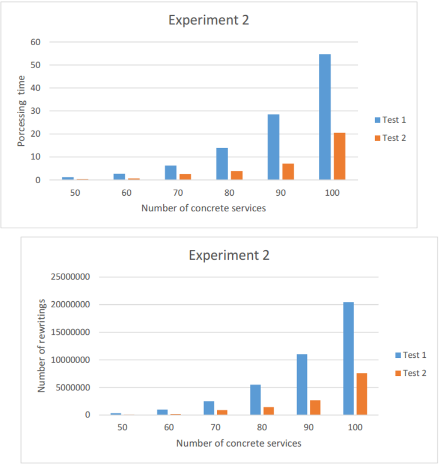

Figure 5: Results concerning processing time (left-side) and rewritings number (right-side).

The results while considering user preferences and SLAs are promisingly. The *Rhone* increases performance reducing rewriting number (around 50 percent) which allows to go straightforward to the rewriting solutions that are satisfactory avoiding any further backtrack and thus reducing successful integration time. Moreover, once the services selection and service composition (rewritings) process are fully guided by the user requirements and SLA, the algorithm avoid producing and executing composition that are not interest for the user. In this sense, the reduces the integration economic cost while delivering the expected results.

4. CONCLUSIONS

recapitular as ideias do projeto e o que vamos fazer... This work proposes a query rewriting algorithm for data integration quality named *Rhone*. Given a query, user preferences and a list of concrete services as input, the algorithm derives rewritings in terms of concrete services that matches with the query and fulfill the user preferences. The formalization and experiments are presented. The results show that the *Rhone* reduces the rewriting number and processing time while considering user preferences and services' quality aspects extracted from SLAs to guide the service selection and rewriting. We are currently performing improvements in the implementation and setting up a multi-cloud simulation in in order to evaluate the performance of the *Rhone* in such context.

5. ADDITIONAL AUTHORS

Additional authors: John Smith (The Thørväld Group, `jsmith@affiliation.org`), Julius P. Kumquat (The Kumquat Consortium, `jpkumquat@consortium.net`), and Ahmet Sacan (Drexel University, `ahmetdevel@gmail.com`).

6. REFERENCES

[1] C. Ba, U. Costa, M. H. Ferrari, R. Ferre, M. A. Musicante, V. Peralta, and S. Robert. Preference-driven refinement of service compositions. In *Int. Conf. on Cloud Computing and Services Science, 2014*, Proceedings of CLOSER 2014, 2014.

[2] M. Barhamgi, D. Benslimane, and B. Medjahed. A query rewriting approach for web service composition. *Services Computing, IEEE Transactions on*, 3(3):206–222, July 2010.

[3] N. Bennani, C. Ghedira-Guegan, M. Musicante, and G. Vargas-Solar. Sla-guided data integration on cloud environments. In *2014 IEEE 7th International Conference on Cloud Computing (CLOUD)*, pages 934–935, June 2014.

[4] K. Benouaret, D. Benslimane, A. Hadjali, and M. Barhamgi. FuDoCS: A Web Service Composition System Based on Fuzzy Dominance for Preference Query Answering, Sept. 2011. VLDB - 37th International Conference on Very Large Data Bases - Demo Paper.

[5] D. A. S. Carvalho, P. A. Souza Neto, G. Vargas-Solar, N. Bennani, and C. Ghedira. *Database and Expert Systems Applications: 26th International Conference, DEXA 2015, Valencia, Spain, September 1-4, 2015, Proceedings, Part II*, chapter Can Data Integration Quality Be Enhanced on Multi-cloud Using SLA?, pages 145–152. Springer International Publishing, 2015.

[6] G. Correndo, M. Salvadores, I. Millard, H. Glaser, and N. Shadbolt. SPARQL query rewriting for implementing data integration over linked data. In *Proceedings of the 1st International Workshop on Data Semantics - DataSem '10*, page 1, New York, New York, USA, Mar. 2010. ACM Press.

[7] U. Costa, M. Ferrari, M. Musicante, and S. Robert. Automatic refinement of service compositions. In F. Daniel, P. Dolog, and Q. Li, editors, *Web Engineering*, volume 7977 of *Lecture Notes in Computer Science*, pages 400–407. Springer Berlin Heidelberg, 2013.

[8] O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS '97, pages 109–116, New York, NY, USA, 1997. ACM.

[9] G. ElSheikh, M. Y. ElNainay, S. ElShehaby, and M. S. Abougabal. SODIM: Service Oriented Data Integration based on MapReduce. *Alexandria Engineering Journal*, 52(3):313–318, Sept. 2013.

[10] A. Y. Halevy. Answering queries using views: A survey. *The VLDB Journal*, 10(4):270–294, Dec. 2001.

[11] A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of the 22th International Conference on Very Large Data Bases*, VLDB '96, pages 251–262, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.

[12] T. Nie, G. Wang, D. Shen, M. Li, and G. Yu. Sla-based data integration on database grids. In *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International*, volume 2, pages 613–618, July 2007.

[13] R. Pottinger and A. Halevy. Minicon: A scalable algorithm for answering queries using views. *The VLDB Journal*, 10(2-3):182–198, Sept. 2001.

[14] Y. Tian, B. Song, J. Park, and E. nam Huh. Inter-cloud data integration system considering

privacy and cost. In J.-S. Pan, S.-M. Chen, and N. T. Nguyen, editors, *ICCCI (1)*, volume 6421 of *Lecture Notes in Computer Science*, pages 195–204. Springer, 2010.

[15] S. S. Yau and Y. Yin. A privacy preserving repository for data integration across data sharing services. *IEEE T. Services Computing*, 1(3):130–140, 2008.