Satisfaction Balanced Mediation*

Jorge-Arnulfo Quiané-Ruiz† , Philippe Lamarre, Sylvie Cazalens, and Patrick Valduriez
Atlas group, INRIA and LINA – Université de Nantes
2 rue de la Houssinière, 44322 Nantes, France
{quiane, lamarre, cazalens}@univ-nantes.fr, Patrick.Valduriez@inria.fr

ABSTRACT

We consider a distributed information system that allows autonomous consumers to query autonomous providers. We focus on the problem of query allocation from a new point of view, by considering consumers and providers' *satisfaction* in addition to query load. We define *satisfaction* as a long-run notion based on the consumers and providers' *preferences*. We propose and validate a mediation process, called $SBMediation$, which is compared to $Capacity\ based$ query allocation. The experimental results show that $SBMediation$ significantly outperforms $Capacity\ based$ when confronted to autonomous participants.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems — *Distributed databases, Query processing*; H.4.0 [**Information Systems Applications**]: General

General Terms

Economics, Management, Performance

Keywords

Autonomous participants, participants' satisfaction, query allocation, imposition, payment

1. INTRODUCTION

We consider a distributed information system with a mediator that enables consumers to access distributed information providers through queries [5]. Consumers and providers (for clarity, we refer to both together as participants) are autonomous in the sense that they are free to enter and leave the system at will and do not depend on anyone to do so. Then, the main function of the mediator is to allocate each incoming query to the providers that can answer it. Much work in this context has focused on *query load balancing (QLB)* [3, 8]. This is obviously important for the efficiency of the system. However, participants may have certain expectations with respect to the mediator, which are not only performance-related.

Providers' expectations reflect their *preferences* in performing some queries rather than others. For example, a provider p_c could represent a pharmaceutical company, which wants to promote a new insect repellent. Thus, it is more interested in treating the queries related to mosquitoes or insect bites than general queries. Once the advertising campaign is over, the provider's *preferences* may change. Consumers expect the mediator to provide them with information that best fits their *preferences*. However, *preferences* are usually considered as private data by participants, since revealing them means revealing strategies. Thus, participants express their *preferences* via an *intention* notion, which can combine different criteria such as *preferences* and *load*. For instance, a provider may not intend to perform queries (even if it prefers them) because of local reasons, e.g. by *overload*. Since it is autonomous, a participant which is dissatisfied too long may just leave the mediator. Intuitively, the system satisfies the participants if the mediator meets their expectations.

In this context, query allocation is a challenge for several reasons. First, to our knowledge, there is no definition of *satisfaction* to characterize how well the system meets the participants' expectations in the long-run. Economical models consider utility and rationality [7], but they are not long-run notions. Second, participants' expectations are usually contradictory. Third, the query demand should be satisfied even if sometimes consumers and providers do not desire to deal with providers and queries, respectively. Finally, participants' departures may have consequences on the functionalities provided by the system. Providers' departure may mean the loss of important system capabilities and consumers' departure is a loss of queries for providers.

After giving some preliminary concepts in Section 2, we present the main contributions of this paper. In Section 3, we define the notion of participants' *satisfaction* that allows knowing whether the query allocation method meets the participants' expectations. We then propose in Section 4 a query allocation mediation process, called $SBMediation$, with the objective of satisfying participants by finding not only relevant providers (i.e. interesting results) to con-

*Work partially funded by ARA "Massive Data" of the French ministry of research (Respire project) and the European Strep Grid4All project.

†This author is supported by the Mexican National Council for Science and Technology (CONACyT).

CIKM'07, November 6–8, 2007, Lisboa, Portugal.

sumers, but also finding interesting queries to providers. In Section 5, we compare, through experimentation, the $SBMediation$'s behavior to a classic QLB method, namely *Capacity based* (e.g. [3]). We mainly study both processes from a *satisfaction* point of view, and analyze the impact on performance of the participants' autonomy. Finally, we conclude in Section 6.

2. PRELIMINARIES

The system consists of a mediator, m, of a set of consumers, C, and of a set of providers, P. Providers are heterogeneous in terms of capacity, thereby some providers are more powerful than others and can treat more queries per time unit. Providers have a finite capacity to perform queries, which denotes the number of computational units they can have. The *utilization* of a provider $p \in P$ at time t, $U_p(t)$, denotes how much it is loaded w.r.t. its capacity at that time. Queries are formulated in a format abstracted as a triple $q = < c, d, n >$ such that $q.c \in C$ is the identifier of the consumer that has issued q, $q.d$ is the description of the task to be done, and $q.n \in \mathbb{N}^*$ is the number of providers to which $q.c$ wishes to allocate q. Set P_q denotes the set of providers registered to a mediator m, which does not appear in the notation for simplicity, and that can treat the query q. There is a large body of work on matchmaking, see e.g. [2, 4] which we could simply reuse. The allocation of some query q is denoted by a vector $All\overrightarrow{oc}_q$ of length N_q such that,

$$\forall p \in P_q,\ All\overrightarrow{oc}_q[p] = \begin{cases} 1 & \text{if } p \text{ gets the query} \\ 0 & \text{otherwise} \end{cases}$$

As we assume all incoming queries should be treated, this leads to $\sum_{p \in P_q} All\overrightarrow{oc}_q[p] = min(q.n,\ N_q)$. In the case where $q.n > N_q$, the consumer $q.c$ gets N_q results instead of $q.n$.

3. PARTICIPANTS SATISFACTION

Two kinds of *satisfaction* could be considered: *(i)* the *satisfaction* of a participant with what it gets from the system, e.g. a consumer that receives results from the providers it wants to avoid is simply not satisfied and *(ii)* the participant's *satisfaction* with the job that the query allocation method does for it, e.g. a provider that performs queries it does not want is not satisfied with the query allocation method if there exist queries of its interests that it does not get. Because of space limitations, we just consider the former in this paper, which we simply call *satisfaction*. The *satisfaction* notion may have a deep impact on the system, because participants may decide whether to stay or to leave the system based on it.

We assume that participants have a limited memory capacity and regularly assess only their k last interactions with the system. Notice that the k value may be different for each participant depending on its memory capacity, but also on its strategy. For simplicity, we assume that they all use the same value of k. Thus, we define the *satisfaction* of participants over their k last interactions. Let us make two general remarks. First, the participant's *satisfaction* may evolve with time, but for the sake of simplicity, we do not introduce time in our notations. Second, the following presentation is completely symmetrical for the participants' *intentions* as well as for their *preferences*. However, for simplicity, we develop the following characteristics only for *preferences*, whose values are in the interval $[-1..1]$.

3.1 Consumer Satisfaction

Intuitively, the consumer's *satisfaction* is useful to answer the following question: *how far the providers that have dealt with the last queries of a consumer meet its expectations?* This notion is based on the memory of a consumer, which is denoted by set IQ_c^k. The *preferences* of a consumer $c \in C$ to allocate its query q to a each provider $p \in P_q$ are stored in vector $Pr\overrightarrow{f}_c^q$. The satisfaction with respect to c and concerning its query q is related to those providers that performed q, which are denoted by set $\widehat{P_q}$. The average of *preferences* expressed by the providers in set $\widehat{P_q}$ is an intuitive technique to define such a notion. Nevertheless, a simple average does not take into account the fact that a consumer may desire different results. The following equation takes into account of this point using n instead of $||\widehat{P_q}||$.

$$\delta_s(c, q) = \Big(\big(\frac{1}{n} \sum_{p \in \widehat{P_q}} Pr\overrightarrow{f}_c^q[p] \big) + 1 \Big) \Big/ 2 \qquad (1)$$

where n stands for $q.n$. Values of function $\delta_s(c, q)$ are in the interval [0..1]. Thus, we define the *satisfaction* of a consumer as the average of its obtained *satisfactions* concerning its k last queries.

Definition 1. *Consumer Satisfaction*

$$\delta_s(c) = \frac{1}{||IQ_c^k||} \sum_{q \in IQ_c^k} \delta_s(c, q)$$

Its values are between 0 and 1. The closer the *satisfaction* to 1, the more a consumer is satisfied.

3.2 Provider Satisfaction

Intuitively, with this notion, we strive to answer the following question: *how well the last queries that a provider has treated meet its expectations?* To define this notion, each provider tracks its shown *preferences* to perform the k last proposed queries in vector $Pr\overrightarrow{f}_p$. The k last proposed queries to a provider p are denoted by set PQ_p^k. Conversely to a consumer that always receives results at each interaction, a provider does not receive all queries that have been proposed to it. Hence, a provider cannot evaluate its satisfaction at each interaction. Thus, we define the satisfaction of a provider $p \in P$ as in Definition 2, where set SQ_p^k ($SQ_p^k \subseteq PQ_p^k$) denotes the set of queries that provider p performed among set PQ_p^k.

Definition 2. *Provider Satisfaction*

$$\delta_s(p) = \begin{cases} \Big(\big(\frac{1}{||SQ_p^k||} \sum_{q \in SQ_p^k} Pr\overrightarrow{f}_p[q] \big) + 1 \Big) \Big/ 2 & \\ 0 & if\ SQ_p^k = \emptyset \end{cases}$$

Its values are in the interval [0..1]. The closer the value to 1, the greater the satisfaction of a provider.

4. THE SBMEDIATION PROCESS

The satisfaction balanced mediation ($SBMediation$) assumes that the consumers in the system show their *intentions*, denoted by vector $\overrightarrow{CI}$, to the mediator, while the providers keep them private. Instead, providers bid on queries, which is a means to reflect their *intentions*. Considering the allocation of some query q, the providers in P_q

bid on q. The bids are represented by a vector $\overrightarrow{B}$, with $\overrightarrow{B}[p] \in \mathbb{R}$ for all $p \in P_q$. If it is positive, the higher the bid is, the more p wants to be allocated q. If it is negative, the lower it is the less p wants to treat q. Intuitively, provider p's bid reflects its *intention* to perform q. This should lead to the providers' satisfaction. However, if only bids are considered, the consumer may be dissatisfied either because its intentions w.r.t. providers are not considered (when it gets answers from providers it doesn't want) or because some queries are not performed (because no provider wants to treat them). Hence, to satisfy the consumer, $SBMediation$: *(i)* directly considers the consumer's *intentions* ($\overrightarrow{CI}_c^q$); *(ii) imposes* the query when not enough providers want to perform it [9]. Processing $SBMediation$ for some query q, amounts to computing *(i)* $All\overrightarrow{oc}_q$, and *(ii)* $Tr\overrightarrow{ans}_q$, which defines all the "monetary" transfers that occur among the providers in P_q. In both steps, bids and consumer's *intentions* have to be balanced to ensure satisfaction.

4.1 Query Allocation

Query q is allocated to the $min(n, N_q)$ "best" providers, which are given by vector of ranking $\overrightarrow{R}$, where $\overrightarrow{R}[1] = p$ iff p is the best ranked, $\overrightarrow{R}[2]$ stands for the second best ranked and so on. Hence, $All\overrightarrow{oc}_q[p] = 1$ iff $\exists i, \overrightarrow{R}[i] = p$ and $i \leq min(n, N_q)$. Vector $\overrightarrow{R}$ is computed with the providers' levels $\overrightarrow{L}$ (Definition 3).

Definition 3. *Vector of providers' levels* $\forall p \in P_q$, *with* $\omega \in [0..1]$,

$$\overrightarrow{L}[p] = \begin{cases} (\overrightarrow{B}[p]+1)^{\omega} \times (\overrightarrow{CI}_c^q[p]+1)^{1-\omega} & if\ \overrightarrow{B}[p] \geq 0 \\ -(-\overrightarrow{B}[p]+1)^{\omega} \times (\overrightarrow{CI}_c^q[p]+1)^{\omega-1} & otherwise \end{cases}$$

Parameter ω reflects the relative importance the mediator gives to the *intentions* or the bids. A competition occurs when the number of providers with a positive bid is equal or higher than n, or else there is an imposition.

4.2 Monetary Transfers

Unlike an usual auction mechanism, bids cannot be directly compared, because of the consumer's *intentions*. Thus, to calculate the payment we define the *theoretical bid* ($\overrightarrow{B}^{Th}(p,l)$), which corresponds to the amount that p should bid for reaching level l. With $\omega \neq 0$ and $\alpha = 1$ if $l \geq 0$, and $\alpha = -1$ otherwise, $\overrightarrow{B}^{Th}(p,l)$ is given by Formula 2. In a competition case, the payments are calculated in the spirit of a generalized Vickrey auction, except that the selected providers pay the amount they should bid to reach the level of the first unselected provider. In an imposition case, theoretical bids are used to define what each of the providers owe due to the imposition of some of them.

$$\overrightarrow{B}^{Th}(p,l) = \alpha max(((\alpha \times l)^{\frac{1}{\omega}} (\overrightarrow{CI}_c^q[p]+1)^{\frac{\alpha(\omega-1)}{\omega}} - 1), 0) \quad (2)$$

The money used in the whole system is *purely virtual*. We could speak of tokens as well, which are used to regulate the system and which are not linked to any particular business model. Also, for the system to be correctly regulated, we must define how the money circulates. In fact, the providers spend and earn money through the mediator only by *(i)*bidding on queries and *(ii)* paying to compensate other providers that have been imposed. They earn money when they are imposed by the process. The mediator never looses money and even tends to accumulate some, thus making the providers poorer and poorer [1]. This could distort the mediation process or even block the system. Hence, this piled up money is regularly redistributed to the providers, in a equitable way. From the providers point of view, this is another, regular, way of earning money.

4.3 The Providers' Bidding Strategies

Once a provider p obtains its *intention* to perform query q (denoted by function $PI_p(q)$) [6], it proceeds to work out its bid to perform q. Intuitively, the bid is the product of its *intention* by its current money balance, denoted by bal_p. Nonetheless, such a simple procedure may lead p to spend all, or almost all, its money on only one query. Thus, to avoid so, p offers at most only a pre-defined percentage of bal_p, denoted by the constant $0 < c_0 \leq 1$. We formally define the providers' bid in Definition 4, where constant c_1 is set to the initial money balance of a provider.

Definition 4. *A provider's Bid*

$$B_p(q) = \begin{cases} PI_p(q) \cdot bal_p \cdot c_0 & if\ PI_p(q) > 0 \\ PI_p(q) \cdot c_1 & otherwise \end{cases}$$

The idea behind above definition is that a provider always sets positive bids when it desires to perform queries and it is not *overutilized*, otherwise it sets a negative bid. This allows a provider to preserve its *preferences* while good response times are also ensured to consumers.

5. VALIDATION

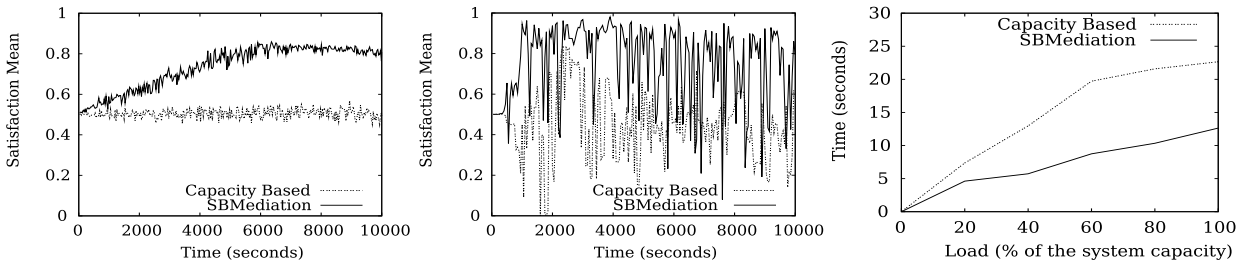
Our main objective is to evaluate, from a satisfaction point of view, how well $SBMediation$ operates.

5.1 Setup

We built a Java-based simulator that models a *mono-mediator* distributed information system following the mediation system architecture presented in [1]. We compare the $SBMediation$ process to *Capacity based* one [3, 8], which is a well-known approach, in distributed information systems, to balance queries among providers. *Capacity based* allocates queries to those providers that have the most available capacity amongst the set P_q of providers. For $SBMediation$ we set $\omega = 0.5$, which means that the consumer's and providers' interests are given the same importance for allocating queries. In all the simulations, the number of consumer and provider sites is 200 and 400 respectively. In our experiments, the consumer's *preferences* denote their *intentions*, while the provider's *intentions* are computed as defined in Section 4.3. To simulate high autonomy in our experiments, we randomly obtain the consumers' preferences between 0 and 1, and the providers' preferences between -1 and 1. We assume that providers decide to leave the system if their *satisfaction* is equal or smaller than 0.3, which is a very low threshold.

5.2 Results

We analyze $SBMediation$ from two points of view: *satisfaction* and *performance*. We also validated $SBMediation$ regarding QLB, which results show that, for *workloads* from 10% to 60% of the total system capacity (i.e. the aggregate capacity of all providers), $SBMediation$ approach is under and so worse than *Capacity based* one. Nonetheless, we observed that, for *workloads* from 60% to 100% of



(a) Consumers' satisfaction for a query load of 100%

(b) Providers' satisfaction for a query load of 100%

(c) Response time with autonomous participants

Figure 1: Satisfaction and performance results.

the total system capacity, *SBMediation* almost ensures the same QLB than *Capacity based*. This is because, for high *workloads*, providers start to pay more attention to their *utilization*. However, we do not present neither discuss further such results by lack of space in this paper.

Concerning *satisfaction*, we observed through our experiments that participants' *satisfaction* is almost the same for different query arrival rates (from 10% to 100% of the total system capacity). By this fact and space reasons, we only present the *satisfaction* results for a query arrival rate of 100% of the total system capacity. On the one hand, consumers benefit from more paid attention to their *intentions* by *SBMediation* than *Capacity based* (see Figure 1(a)). Consumers are always more satisfied with *SBMediation* because *Capacity based* proceeds in a blind way as far as this point is concerned. On the other hand, providers are also always more satisfied with *SBMediation* than *Capacity based* (see Figure 1(b)). This demonstrated that *SBMediation* gives in average interesting queries to providers and *Capacity based* punishes them with uninteresting ones. Notice that, the values of the providers' *satisfaction* suffer from greater oscillations than those of consumers, because of natural competition of providers for performing queries.

Now, we proceed to study the impact on performance of the providers' autonomy. We observed during our experimentations that while *Capacity based* looses in average 60% of providers for all query arrival rates, *SBMediation* looses only a 27% of providers! Indeed, such provider's departures are reflected on the ensured response time[1]. We observe in Figure 1(c) that *SBMediation* significantly outperforms *Capacity based*, which cannot ensure good response times. Therefore, *SBMediation* can scale up when confronted to autonomous providers while *Capacity based* cannot.

6. CONCLUSION

In this paper, we addressed the query allocation problem in distributed information systems from a new point of view, by considering not only query load but also participants' *satisfaction*. Our work brings several contributions.

First, we proposed the *satisfaction* notion that reflects, in the long run, whether participants' expectations are met by the query allocation method. This definition is original since it is independent of how participants compute their *intentions* and how the mediation process considers them.

[1]As is conventional, it is defined as the elapsed time from the moment that a query q is issued to the moment that the $q.c$ site receives the response of q.

Second, we propose a mediation process, called *SBMediation*, that considers the participants' *intentions* while allocating queries. We discussed how query allocation and invoicing steps leads to participants' *satisfaction*. The originality of *SBMediation* is to satisfy both participants' expectations and query demand.

Finally, we evaluated and compared, through experimentation, the behavior of *SBMediation* to the behavior of *Capacity based*. We demonstrated that *SBMediation* significantly outperforms *Capacity based*. We showed that consumers and providers are, in general, very satisfied with *SBMediation*. This is not the case for *Capacity based* which suffers from several providers' departures due to dissatisfaction. Furthermore, the results demonstrate that *SBMediation* can scale up in these systems while *Capacity based* cannot. Finally, since *SBMediation* considers the consumers' *intentions* and providers' bids without any consideration about how they are computed, *SBMediation* is self-adaptable to the changes in their expectations.

7. REFERENCES

[1] P. Lamarre, S. Cazalens, S. Lemp, and P. Valduriez. A Flexible Mediation Process for Large Distributed Information Systems. In *Procs. of CoopIS*, 2004.

[2] L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Procs. of the WWW Conf.*, 2003.

[3] R. Mirchandaney, D. Towsley, and J. Stankovic. Adaptive Load Sharing in Heterogeneous Distributed Systems. *Parallel and Distributed Computing*, 9(4), 1990.

[4] M. H. Nodine, W. Bohrer, and A. H. Ngu. Semantic Brokering over Dynamic Heterogeneous Data Sources in InfoSleuth. In *Procs. of the ICDE Conf.*, 1999.

[5] T. Özsu and P. Valduriez. *Principles of Distributed Database Systems (2nd ed.)*. Prentice-Hall, 1999.

[6] J.-A. Quiané-Ruiz, P. Lamarre, and P. Valduriez. SQLB: A Query Allocation Framework for Autonomous Consumers and Providers. In *Procs. of the VLDB Conf.*, 2007.

[7] T. W. Sandholm. *Multiagent Systems, a modern approach to Distributed Artificial Intelligence*, chapter Distributed Rational Decision Making. The MIT Press, 2001.

[8] N. G. Shivaratri, P. Krueger, and M. Singhal. Load Distributing for Locally Distributed Systems. *IEEE Computer*, 1992.

[9] Y. Shoham and M. Tennenholtz. Fair Imposition. In *Procs. of IJCAI*, 2001.