SQLB: A Query Allocation Framework for Autonomous Consumers and Providers

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

Jorge-Arnulfo Quiané-Ruiz, Philippe Lamarre, and Patrick Valduriez
Atlas group, INRIA and LINA – Université de Nantes

VLDB Conference
September 27, 2007

Roadmap

1 Motivation and Problem Definition

2 Satisfaction Model

3 SQLB Framework

4 Validation

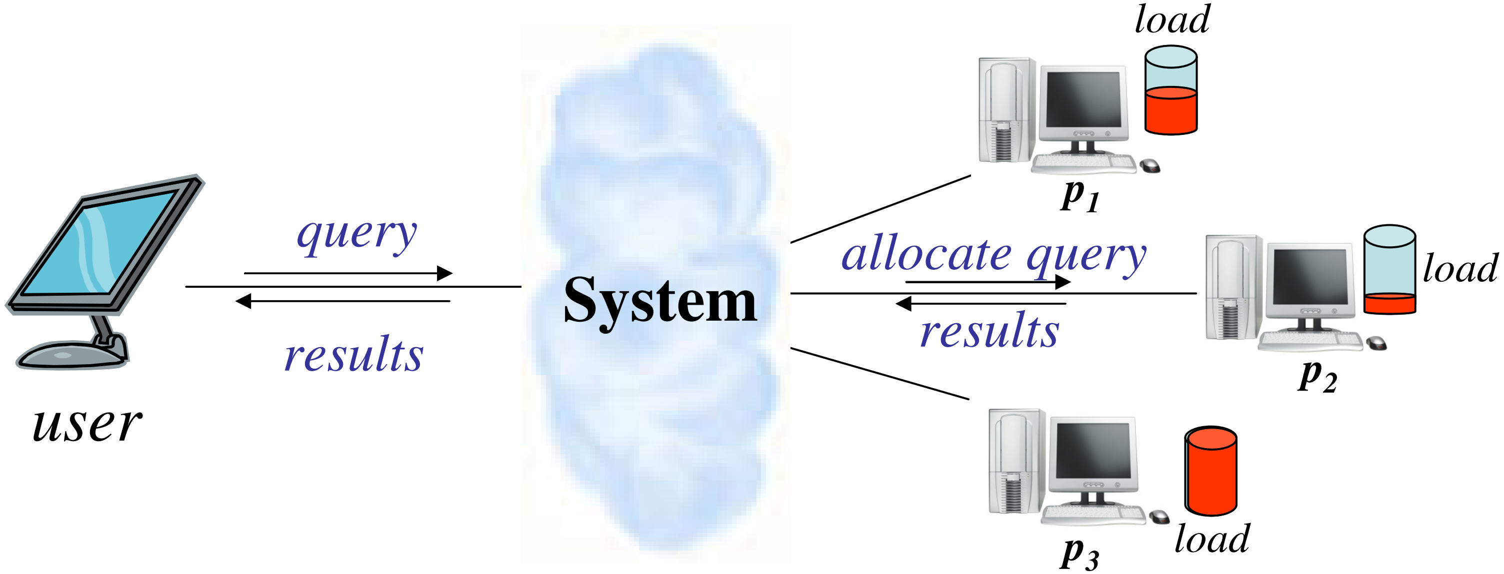
5 Conclusion

Context

- **Large-scale D**istributed **I**nformation **S**ystems (DIS)
- **Autonomous** participants (consumers and providers)
  - May **join** and **leave** the system **at will**
  - Have **interests** towards providers and queries
- Focus on **Query Allocation**

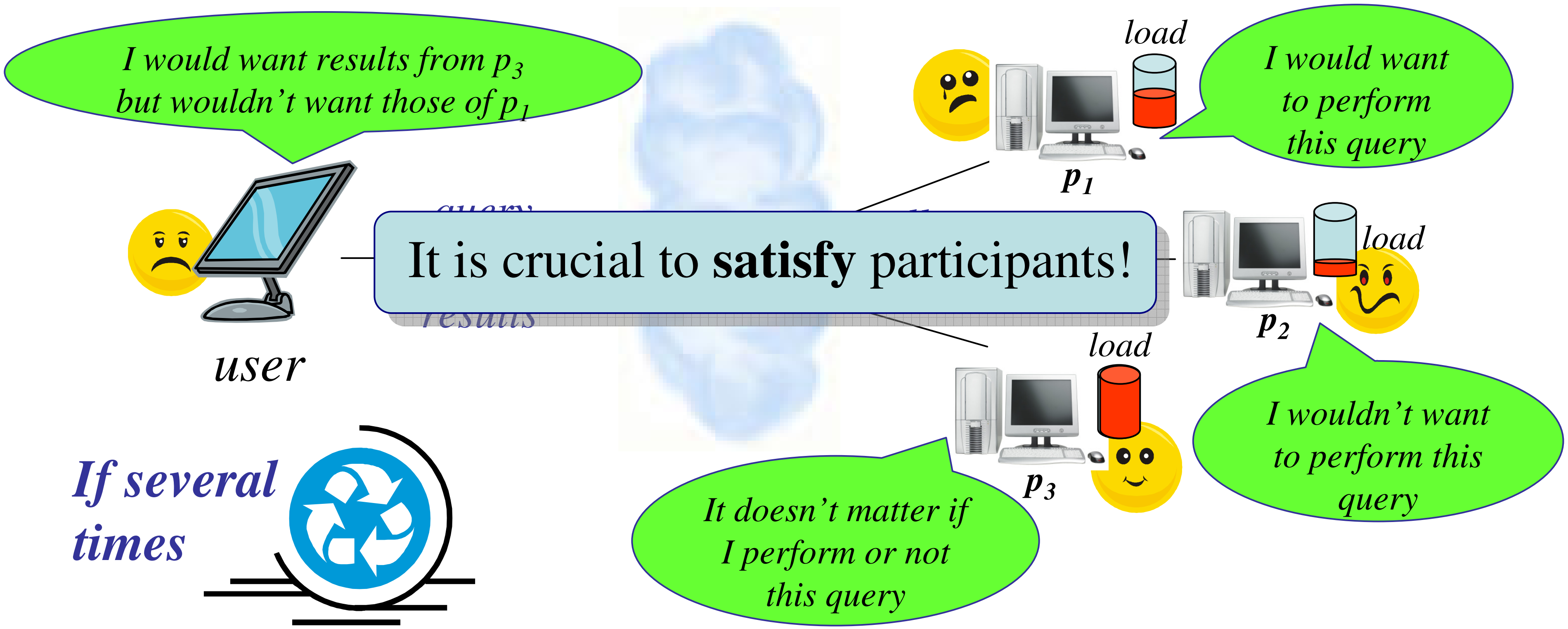
Query Allocation

Query **l**oad **b**alancing (QLB) : **maximize** overall system **performance** (throughput and response time)



Problem Overview

However, participants may have certain **expectations** (***intentions***) that are **not only performance-related**



It is crucial to **satisfy** participants!

Problem Statement

Assumptions:

- **Large-scale** and **heterogeneous** DIS
- **Autonomous** participants
- Queries **must** be treated whenever possible

Let:

- $\boldsymbol{q} = \boldsymbol{< c, d, n >}$ be an incoming query
- $\boldsymbol{P_q}$ be the set of providers that are able to deal with $\boldsymbol{q}$

Problem:

- **Allocate** each $\boldsymbol{q}$ to a set $\boldsymbol{P_q}$ such that **good response time** and **participants' satisfaction** are **ensured**

Challenge

Query allocation is **hard** because:

- Query **demand** should be **satisfied**
- Participants should **be satisfied** to some (which?) extent
- Participants' **expectations** may be **contradictory**

Our Contributions

SQLB Model

A **model** to characterize the participants' **expectations** in the **long-run**

SQLB Framework

A **framework** to allocate queries **based on** the participants' **satisfaction**

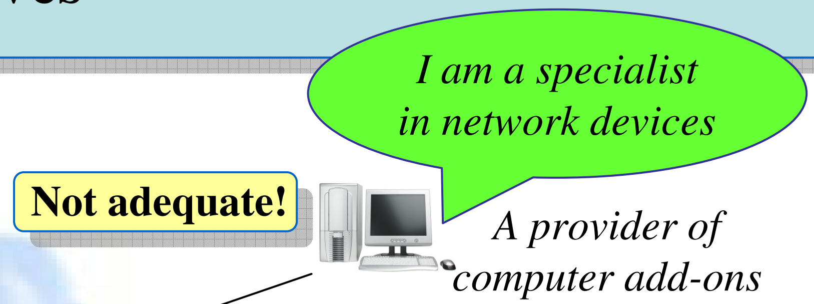
Roadmap

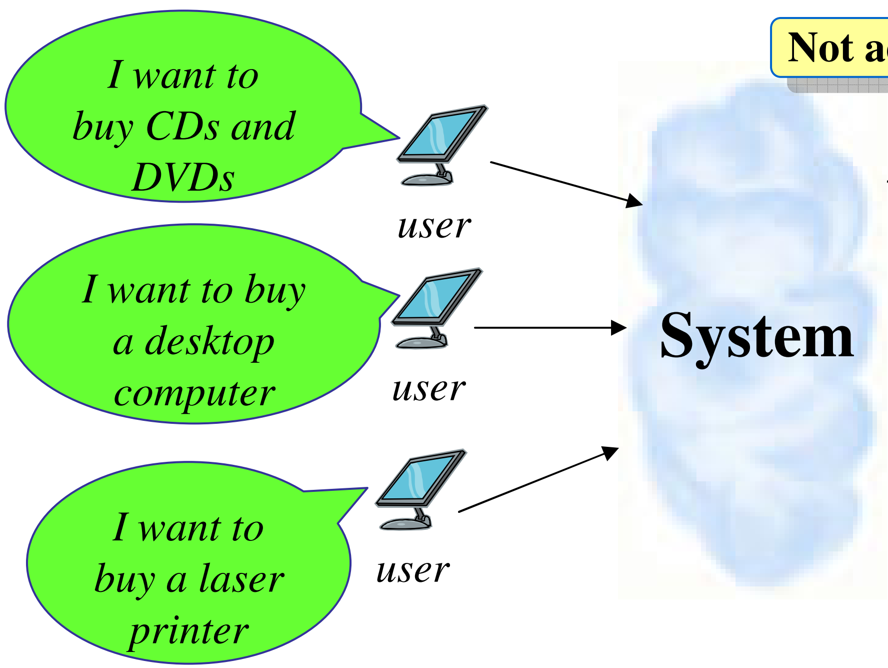
Satisfaction Model

- Captures how well the system **meets** the participants' **expectations**,
- Three notions:
  - **Adequation**
  - **Satisfaction**
  - **Allocation Satisfaction**
- They are **based on** the ***k*** last participants' **interactions** with the system

Participant Characterization (1/3)

Adequation: enables a participant to **know** whether it can **reach** its **objectives**





The Math

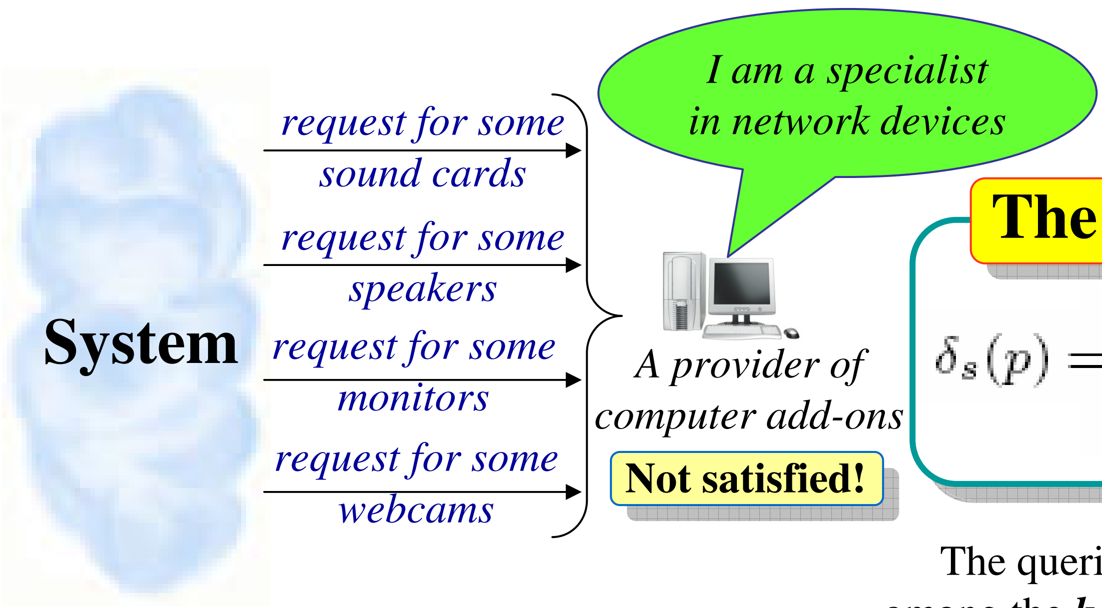
$$\delta_a(p) = \left(\left(\frac{1}{||PQ_p^k||}\sum_{q\in PQ_p^k} \overrightarrow{PPI_p}[q]\right)+1\right)\Big/\,2$$

The ***k*** last proposed queries by the system to ***p***

p's desire to perform query ***q***

Participant Characterization (2/3)

Satisfaction: enables a participant to **know** whether it is **fulfilling** its **objectives**



The Math

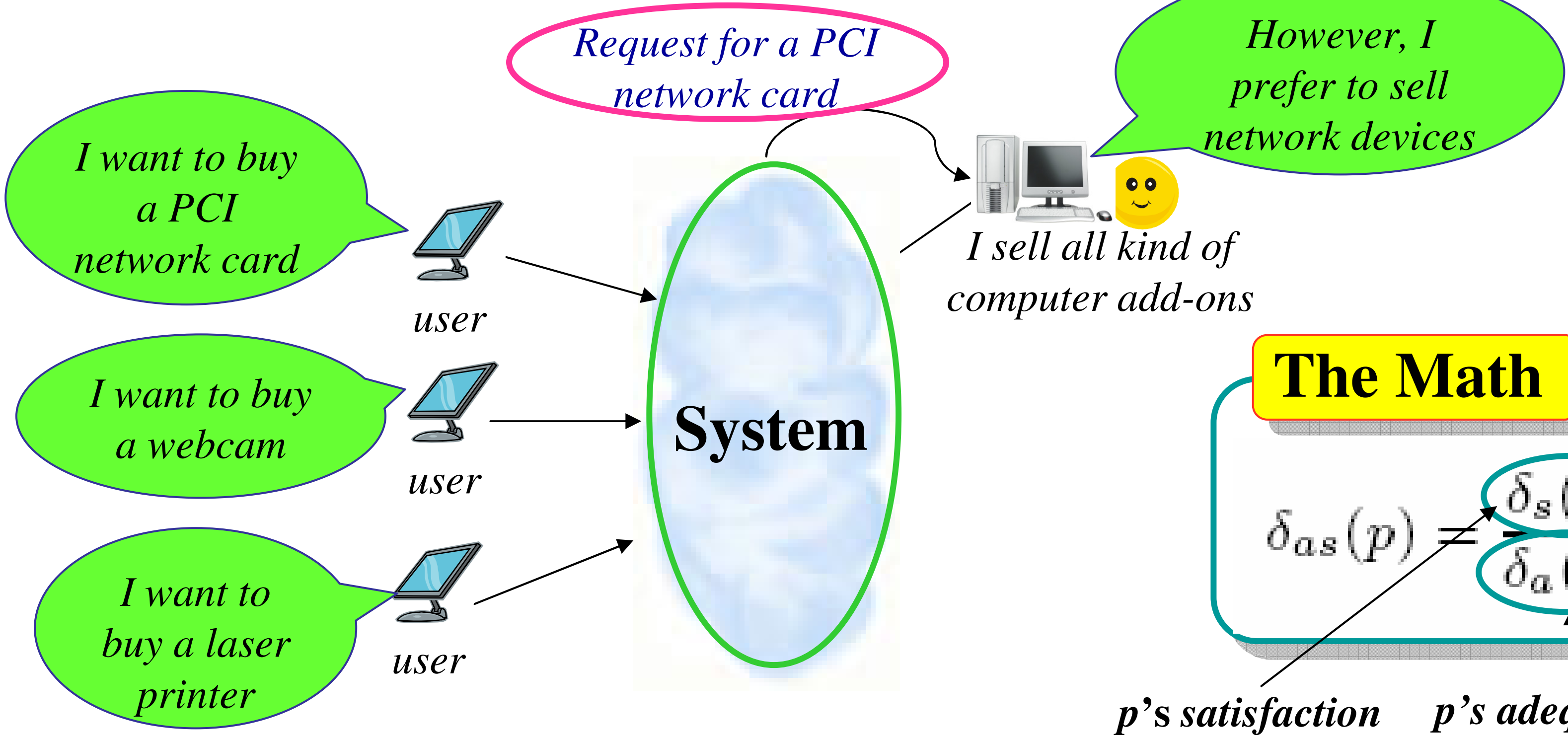
$$\delta_s(p) = \left(\frac{1}{||SQ_p^k||} \sum_{q \in SQ_p^k} \overrightarrow{PPI_p}[q] + 1 \right) / \, 2$$

The queries that ***p*** performed among the ***k*** last queries the system proposed to it ($SQ_p^k \subseteq PQ_p^k$)

p's desire to perform query ***q***

Participant Characterization (3/3)

Allocation Satisfaction: enables a participant to **know** the **reason** of its **dissatisfaction** or **satisfaction**



The Math

$$\delta_{as}(p) = \frac{\delta_s(p)}{\delta_a(p)}$$

p's satisfaction *p's adequation*

Roadmap

1 Motivation and Problem Definition

2 Satisfaction Model

3 SQLB Framework

4 Validation

5 Conclusion

Query Allocation Objectives

- **Guarantee** good system performance
- Be **self-adaptable** to the participants' **expectations**
- Give **interesting sources** to **consumers** and **interesting queries** to **providers**
  - **To do so,** participants are required to express their **intentions**

Consumer Side: Intention

- Defines the consumer's **desire** to see a given provider performing its query
- Is the result of merging consumer's **preferences** with the provider's **reputation**

The Math

c's preference to allocate ***q*** to ***p***

p's reputation

$$ci_c(q,p) = \begin{cases} prf_c(q,p)^{v} \times rep(p)^{1-v} & if\ prf_c(q,p) > 0 \wedge \\ & \wedge\, rep(p) > 0 \\ -\left((1 - prf_c(q,p) + \epsilon)^{v} \times (1 - rep(p) + \epsilon)^{1-v} \right) & else \end{cases}$$

Intention of a consumer ***c*** to allocate its query ***q*** to a provider ***p***

Balance in accordance to ***c***'s past experiences with ***p***

Prevents the intention from taking zero values

Provider Side: Intention

- Defines the provider's **desire** to perform a given query
- Is the result of merging provider's **preferences** with the provider's **utilization**

The Math

p's preference to perform *q*

p's utilization

$$pi_p(q) = \begin{cases} (prf_p(q))^{1-\delta_s(p)}(1-U_t(p))^{\delta_s(p)}, & if\ prf_p(q) > 0 \wedge \\ & \wedge U_t(p) < 1 \\ -\left((1-prf_p(q)+\epsilon)^{1-\delta_s(p)} \times (U_t(p)+\epsilon)^{\delta_s(p)}\right) & else \end{cases}$$

Intention of a provider *p* to perform a query *q*

Balance in accordance to *p*'s satisfaction

It prevents the intention from taking zero values

Mediator Side: Providers' Score

- Defines the provider's **importance** to be allocated a given query
- Is the result of merging the consumer's and provider's **intention**

The Math

p's intention to perform *q*

q.c's intention to allocate *q* to *p*

$$scr_q(p) = \begin{cases} \left(\overrightarrow{PI}_q[p]\right)^{\omega} \left(\overrightarrow{CI}_q[p]\right)^{1-\omega} & if\ \overrightarrow{PI}_q[p] > 0 \wedge \\ & \wedge\ \overrightarrow{CI}_q[p] > 0 \\ -\left(\left(1 - \overrightarrow{PI}_q[p] + \epsilon\right)^{\omega} \left(1 - \overrightarrow{CI}_q[p] + \epsilon\right)^{1-\omega}\right) & else \end{cases}$$

Score of a provider *p* given a query *q*

Balance in accordance to *q.c*'s and *p*'s satisfaction

It prevents the score from taking zero values

$$\omega = \left(\left(\delta_s(c) - \delta_s(p)\right) + 1\right) / \, 2$$

Mediator Side: Query Allocation

Algorithm 1: QueryAllocation

```
Input  : q, P_q
Output: Alloc_q
begin
    // Consumer's intentions
    fork ask for q.c's intentions;
    // Providers' intention
    foreach p ∈ P_q do
        fork ask for p's intention w.r.t. q ;
    waituntil CI_q and PI_q be calculated or timeout;
    // Scoring and ranking providers
    foreach p ∈ P_q do
        compute p's score concerning CI_q[p] & PI_q[p] ;
    rank the set P_q of providers, R^q, regarding scr_p(q) ;
    // Query Allocation
    for i = 1 to min(n, N_q) do Alloc[R^q[i]] ← 1 ;
    for j = min(n, N_q) + 1 to N do Alloc[R^q[j]] ← 0 ;
end
```

Input: q, P_q
Output: $\overrightarrow{Alloc}_q$

input

Consumer's and providers' intention w.r.t. q

we compute $scr_q(p)$

where $\overrightarrow{R}^q[1]$ **is the best scored provider and** $\overrightarrow{R}^q[N]$ **is the worst**

$$\overrightarrow{Alloc}[p] = \begin{cases} 1 & \text{if } p \text{ gets the query} \\ 0 & \text{otherwise} \end{cases}$$

Roadmap

1 Motivation and Problem Definition

2 Satisfaction Model

3 SQLB Framework

4 Validation

5 Conclusion

Validation

Objectives

- **Evaluate** if participants are **satisfied** with the query **allocation** process
- **Evaluate** the **impact on** performance of the participants' **departure**

Tested methods

- **Capacity based** (QLB approach)
- **Mariposa-like** (economic approach)
- **SQLB** (our proposal)

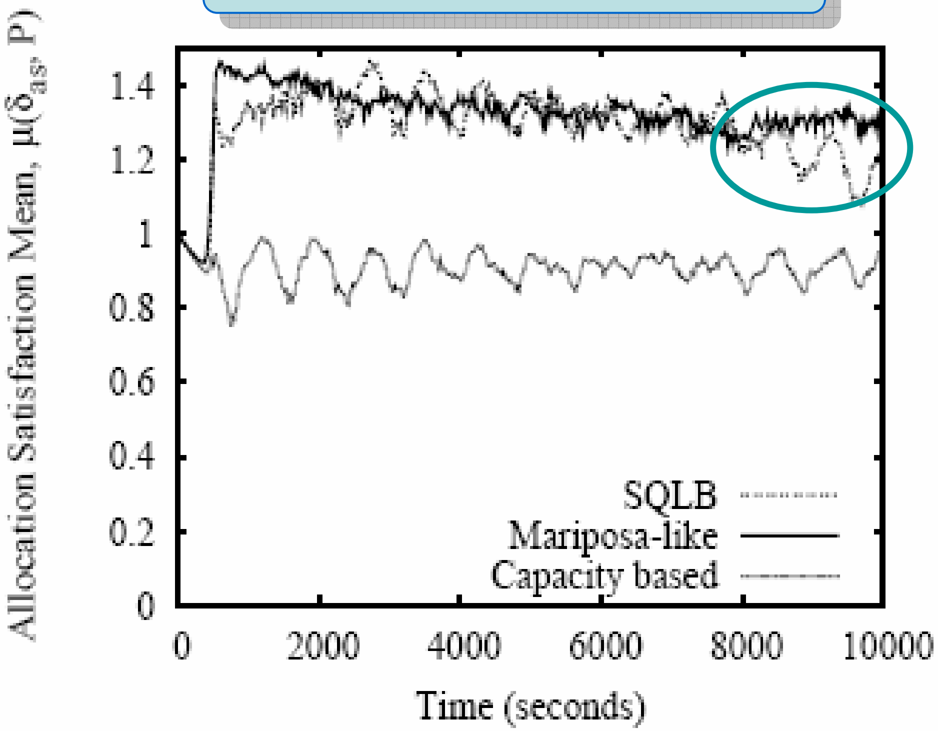
Setup

Parameter	Value
Number of consumers	200
Number of providers	400
Number of mediators	1
Query distribution	Poisson
k size for consumers	200
k size for providers	500

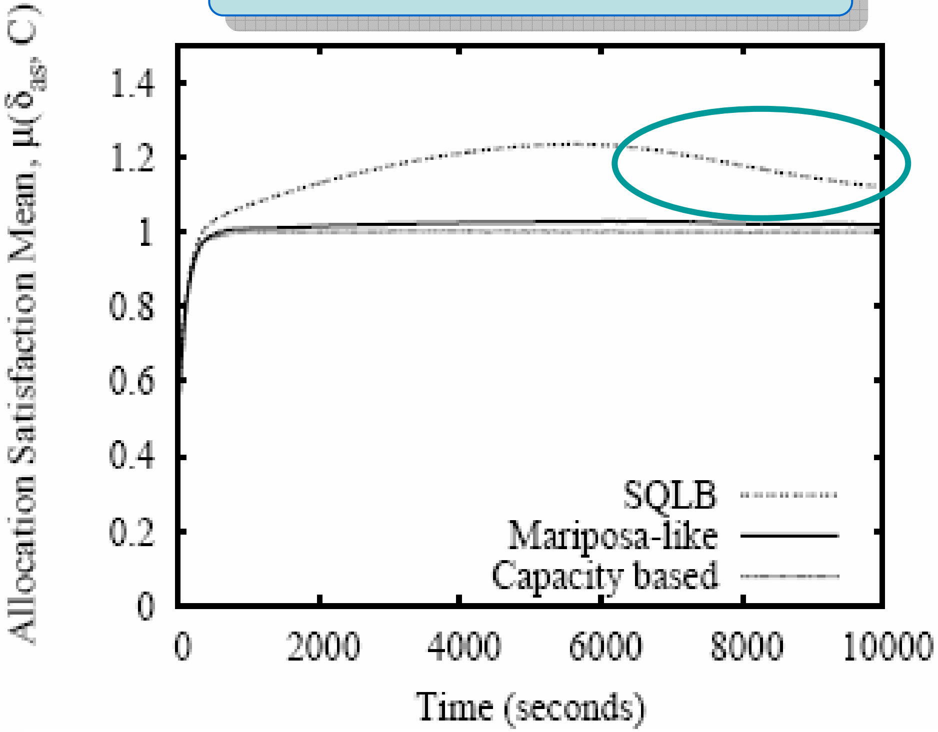
We **implemented** our algorithms in **Java** and used **SimJava** to simulate the **network** communication

Satisfaction Results

Providers' allocation satisfaction



Consumers' allocation satisfaction

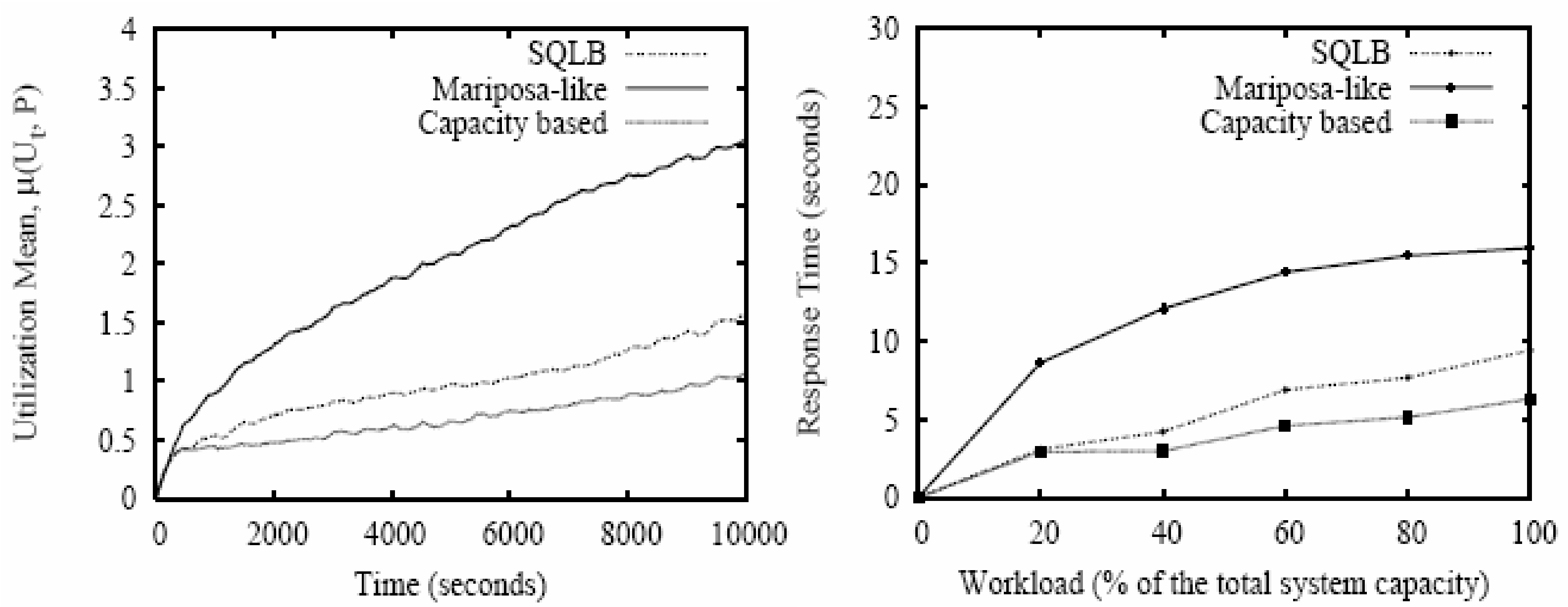


SQLB has the ***same*** performance than ***Mariposa-like*** while ***Capacity based*** penalizes providers

Consumers are **satisfied only** with the ***SQLB*** approach

Performance Results (1/2)

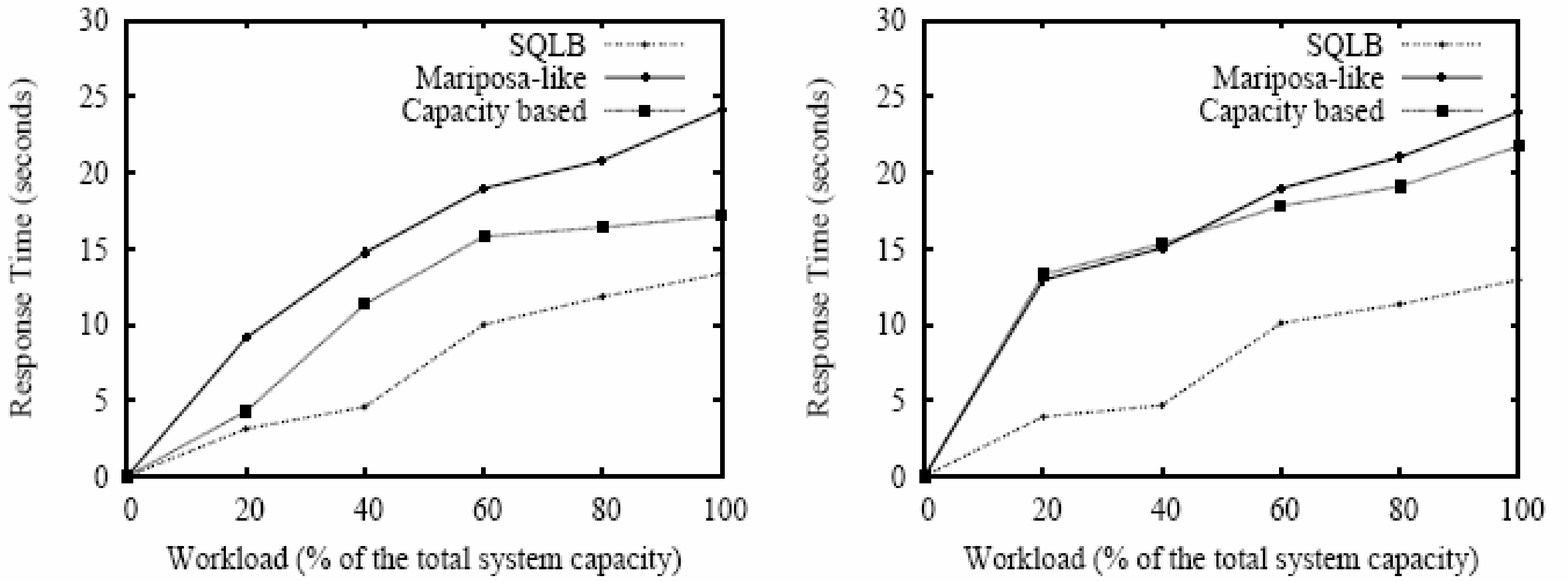
Captive participants: they are not allowed to leave the system



Even if **not** designed for **captive** environments, ***SQLB*** ensures **quite good** response times

Performance Results (2/2)

Autonomous providers: they may leave the system at will



(a) Providers may leave by *dissatisfaction* or *starvation*. (b) Providers may leave by *dissatisfaction*, *starvation*, or *overutilization*.

SQLB significantly outperforms ***Capacity based*** and ***Mariposa-like*** by a factor of **2** in average

Roadmap

1 Motivation and Problem Definition

2 Satisfaction Model

3 SQLB Framework

4 Validation

5 Conclusion

Summary

- *SQLB* **Model**
  - **Characterizes** the participants' **expectations**
  - **Allows** to **design** and **evaluate** query allocation methods for **autonomous** environments

- *SQLB* **framework**
  - **Allows** trading **consumers'** intentions for **providers'** intentions in accordance to their **satisfaction**
  - **Avoids** query **starvation**

- **Future work**
  - Develop an **economical** version of our approach
  - Consider **super-peer** and **unstructured** P2P systems

Danke!

Questions ?

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