Reinicke, Michael; Streitberger, Werner; Eymann, Torsten

Working Paper

A scalability analysis of grid allocation mechanisms

Bayreuther Arbeitspapiere zur Wirtschaftsinformatik, No. 31

Provided in Cooperation with:
University of Bayreuth, Chair of Information Systems Management

Suggested Citation: Reinicke, Michael; Streitberger, Werner; Eymann, Torsten (2008) : A scalability analysis of grid allocation mechanisms, Bayreuther Arbeitspapiere zur Wirtschaftsinformatik, No. 31, Universität Bayreuth, Lehrstuhl für Wirtschaftsinformatik, Bayreuth, https://nbn-resolving.de/urn:nbn:de:bvb:703-opus-3875

This Version is available at:
https://hdl.handle.net/10419/52638

Lehrstuhl für
Wirtschaftsinformatik
Information Systems
Management

No. 31
Februar 2008

Bayreuther Arbeitspapiere zur Wirtschaftsinformatik

Michael Reinicke / Werner Streitberger / Torsten Eymann

A Scalability Analysis of Grid Allocation Mechanisms

Bayreuth Reports on Information Systems Management

UNIVERSITÄT
BAYREUTH

ISSN 1864-9300

Die Arbeitspapiere des Lehrstuhls für Wirtschaftsinformatik dienen der Darstellung vorläufiger Ergebnisse, die i. d. R. noch für spätere Veröffentlichungen überarbeitet werden. Die Autoren sind deshalb für kritische Hinweise dankbar.

The Bayreuth Reports on Information Systems Management comprise preliminary results which will usually be revised for subsequent publications. Critical comments would be appreciated by the authors.

Authors:

Michael Reinicke (University of Bayreuth)
Werner Streitberger (University of Bayreuth)
Torsten Eymann (University of Bayreuth)

Information Systems and Management Working Paper Series

Edited by:

Prof. Dr. Torsten Eymann

Managing Assistant and Contact:

Raimund Matros
Universität Bayreuth
Lehrstuhl für Wirtschaftsinformatik (BWL VII)
Prof. Dr. Torsten Eymann
Universitätsstrasse 30
95447 Bayreuth
Germany

Email: raimund.matros@uni-bayreuth.de

ISSN 1864-9300

A Scalability Analysis of Grid Allocation Mechanisms

Michael Reinicke*, Werner Streitberger* and Torsten Eymann*

*Chair of Information Systems Management
University of Bayreuth, Germany
Email: {reinicke,streitberger,eymann}@uni-bayreuth.de

***Abstract*—Grid Computing is a paradigmatic application for the requirements associated with the exponentially growing complexity in the engineering, operation and maintenance of today's information systems. Grid Computing comprises ever-growing global communication infrastructures and millions of possible system elements into the picture. Self-organizing, agent-based Grids share many properties and research questions with Autonomic Computing. In both concepts, the procedure of matching service provisioning to service demand is often centrally (geographically) realized by a dedicated resource broker instance. However, one of the prerequisites of this resource broker would be that it scales with the increasing size of the system. This article examines the broker's behavior with regard to a varying number of participating nodes and shows that incremental losses have to be accepted in central resource allocation when introducing new nodes. In comparison, a self-organizing, decentralized, bilateral bargaining approach shows more evenly behavior in response times and allocation efficiency.**

***Index Terms*—Grid Computing, Resource Allocation, Resource Broker, Scalability Analysis**

I. Resource Brokerage in Self-Organizing Grid Networks

Since several years, companies like Sun, IBM and HP are working on software architectures that allow clients to obtain computing services on demand from anywhere in a computer network [1] [2] [3].

This so-called *Service Oriented Computing* paradigm allows bundling remote resources from diverse providers in the net, creating economies of scale as well as economies of scope. The term *Grid Computing* [4] describes global, Internet-based bundling and on-demand access of computational resources like processor power or data storage. Aggregated processor power worldwide combines to a virtual computer with an immense amount of capacity, enabling users to process computation-intense jobs. Being connected via powerful network links, it is irrelevant where this accomplishment will be fulfilled. Usual examples are systems for number-crunching applications, like computations in climate change, cancer therapy or decoding the human genome [5].

The *Grid* network forms a virtual organization of service providers and clients and thus possesses a system identity. Every service provider knows whether it belongs to the system or not, and the client addresses its requests "to the Grid network" rather than to a bunch of unknown resources.

As the demand and offer situation in the network continuously and (assumably unpredictably) varies, the Grid system reconfigures these resources such that the objectives of availability and resource usage are sufficiently met. On the application level, this variation might be caused by nodes entering or leaving the network (like in P2P networks); on a technical level, network connections might disappear or get overly congested. This requires self-adaptation and learning mechanisms, which lead to a self-healing of the system and to continuous self-optimization of its processes.

This paper focuses on a specific functionality of an Grid system, which can be either solved by a centralized or a self-organizing solution. Out of a list of candidate service provider instances, a matchmaker (or resource broker) usually selects a match between client and provider which satisfies both parties. The selection process will, in the likely case of multiple, redundant service instances, need to order available alternatives according to some optimization criterion. These can be of technical nature, like fastest response time, or economic, like overall allocation efficiency.

Grid Computing systems show characteristics that highly affect the performance of a chosen service selection method and resource broker instance implementation:

- **Dynamicity**: Networks are changing environments and there is a continuous need for adaptation.
- **Large scale**: Grids contain a large number of nodes, possibly tens of thousands, and it is necessary to use mechanisms that can scale up.
- **Partial knowledge**: It is not possible to determine at any point in time (or a priori) the complete state of the system.
- **Evolutionary**: The system is evolving in directions that cannot be foreseen in the initial setup.

The remainder of this article is structured as follows. Section II presents some related work concerning service selection. Section III develops a formalization of the simulated centralized service selection approach and section VI draws a comparison to a novel decentralized selection strategy. Section V shows a simulation, analyzing the behavior of a generalized economic allocation mechanism. This will be followed by an evaluation showing the scalability behavior of both mechanisms. Section VI finally presents a conclusion and draws an outlook for further research.

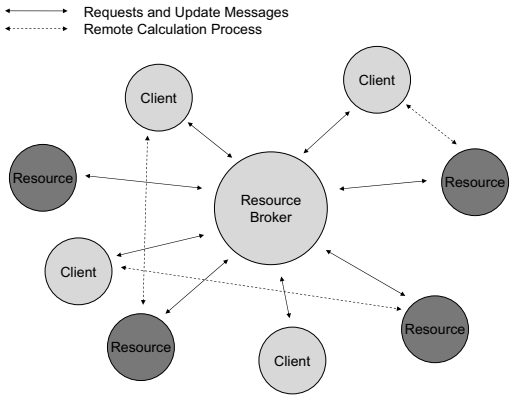

Fig. 1. Centralized Service Discovery and Service Selection

II. Related Work

The most common implementation for service selection is that of a centralized matchmaker or resource broker. The candidate list is evaluated centrally by the matchmaker instance, and the requesting client receives only the resulting match (see Figure 1). Clients and service providers update the centralized resource broker in synchronized rounds about service requests and effective availability. Receiving and storing this data, the resource broker periodically selects – after collecting a certain amount of incoming information – providers and clients to commit a temporary relationship. After notifying the clients and resources about the respective allocation, it is up to the clients to claim the resource directly.

Existing matchmaking approaches are realized in Condor-G [6], Darwin [7], Globus [8], which rely on using a centralized instance which implements both service selection and service discovery procedures.

The centralized approach has several drawbacks, some of which cannot be solved by incremental development: One prerequisite is that the coordinator needs to have actual, global knowledge on the state of the network. This is mostly achieved by calculating the time steps such that actual status information from all nodes arrives safely at the coordination instance. However, if the diameter of the network is too large, this approach leads to long latency times for the nodes. Furthermore, during this time the environment must not change in order for the optimization mechanism to produce results applicable to the problem set.

Grid computing networks, however, are very dynamic and fast changing systems, service demands and communication paths changes are frequent [9], and new different services can be created and composed continuously [10]. Thus, dynamic networks need a continuously updating coordination mechanism, which reflects the changes in the environment. When increasing the number of participants in the network that rely on the centralized matchmaker to allocate the jobs to the resources, issues of scalability arise. Due to the increase in complexity, a centralized resource broker will supposedly not be able to cope with an increasing number of connected nodes and resources.

III. Formalization of a Centralized Grid Market Scenario

This section formalizes a simple Grid Computing market scenario to analyze different matchmaking procedures. Two types of agents are created. Sellers provide the resource broker with service offers, Buyers send bids to request a proper job allocation. The seller represents both a physical resource and a software instance (software agent) running on top of it, analogous to an autonomic computing resource and an autonomic manager instance, respectively. This "service copy" instance can exclusively process one transaction at a time.

A. Obtaining Bids

T is assumed to be a fixed time duration, during which the resource broker collects bids by opening an auction. The auction begins when a buyer k_1 requires the desired good at a certain time t_0 and offers a maximum price K_1 for that good (*bid*). The sellers receive a message requesting to propose a counteroffer (*ask*). A busy seller, whose service copy is occupied at the time of reception of the buyers' offer, immediately rejects the offer by sending a *reject* to the resource broker. Other sellers send counter*offers* during the interval $[t_0, t_0 + T]$. When this time span ends, the resource broker selects a winner out of the received bids and determines a – at least for this round – valid market price.

B. Determining the Market Price

The buyers that have issued an offer to the resource broker in time interval $[t_0, t_0 + T]$ are denoted by

$$k_1, \ldots, k_N$$

and the corresponding bids are

$$K_1, \ldots, K_N.$$

Analogous are

$$v_1, \ldots, v_M$$

the sellers, who have issued the offers

$$V_1, \ldots, V_M$$

in the same time interval.

For any set $A \subset \mathbb{R}$ let the function

$$I_A : \mathbb{R} \longrightarrow \{0; 1\}, \qquad I_A : p \mapsto \begin{cases} 1, & \text{if} \quad p \in A \\ 0, & \text{if} \quad p \notin A \end{cases}$$

be the indicator function of A.

A buyer having bit K_i will buy the good at price p, when

$$p \leq K_i$$

and buyer k_i contributes to the aggregated demand curve

$$p \mapsto I_{[0;K_i]}(p) \ .$$

Thus, an aggregated demand curve against the price p results:

$$D : p \mapsto \sum_{i=1}^{N} I_{[0;K_i]}(p) \ .$$

Accordingly,

$$S : p \mapsto \sum_{j=1}^{M} I_{[V_j;\infty[}(p)$$

is the cumulated supply function against price p.

Now, let

$$p_1 \leq p_2 \leq p_3 \leq \cdots \leq p_{N+M}$$

be the ordered bids $K_1, \ldots, K_N, V_1, \ldots, V_M$ (index by size).

To determine the equilibrium price, the *Surplus*

$$\Delta(p_i) = S(p_i) - D(p_i)$$

is calculated for each price p_i and $i \in \{1, \ldots, N+M\}$. $\Delta(p_i)$ describes the surplus of sellers respectively buyers at price P_i. As the demand and supply curves are not continuous, generally no market clearing price p^* could be determined where $\Delta(p^*) = 0$ applies. The value

$$\Delta^* = \min\{|\Delta(p_1)|, \ldots, |\Delta(p_{N+M})|\}$$

describes the minimum of this surplus at a price p^* and

$$\Delta^* = |\Delta(p^*)|$$

maximizes the transactional volume.

As D and S are monotonous functions, and p_{i+1} for $i \in \{1, \ldots, N+M-1\}$ is a jump discontinuity of D respectively S finds that

$$D(p_i) > D(p_{i+1})$$

or

$$S(p_i) < S(p_{i+1})$$

hold. Therefore is

$$S(p_i) - D(p_i) < S(p_{i+1}) - D(p_{i+1}) \ ,$$

i.e.

$$\Delta(p_i) < \Delta(p_{i+1}) \qquad \forall\, i \in \{1, \ldots, N+M-1\} \ .$$

So, only two cases are possible:

- First case: It exists only one $p^* \in \{p_1, \ldots, p_{N+M}\}$ with

  $$\Delta^* = |\Delta(p^*)| \ .$$

  In this case the price p^* will be determined as market price by the resource broker.
- Second case: Two prices exist, $p_1^*, p_2^* \in \{p_1, \ldots, p_{N+M}\}$, both satisfying the equation

  $$\Delta^* = |\Delta(p)| \qquad \text{for} \quad p \in \{p_1, \ldots, p_{N+M}\}.$$

  In this case additionally $\Delta(p_1^*) = -\Delta(p_2^*)$ holds.
  The resource broker then determines the price $p^* \in \{p_1^*, p_2^*\}$ as a market price, obtaining

  $$\Delta(p^*) > 0 \ .$$

C. Allocation of Buyers and Sellers

In this model buyers and sellers reside at fixed, but different locations in the network. The distance between the agents will be identified by node reference points that have been passed by the messages. This will be an integer number and is denoted for agents a_1, a_2 by

$$d(a_1, a_2) \in \mathbb{N}.$$

Now let $\{k_{i_1}, \ldots, k_{i_r}\}$ be the set of buyers for which holds

$$K_i \geq p^* \ .$$

These are the buyers, that are willing to pay the market price. Furthermore $\{v_{j_1}, \ldots, v_{j_l}\}$ is the set of the sellers for that holds

$$V_j \leq p^* \ .$$

These are the sellers that are willing to accept the market price.

The buyers $\{k_{i_1}, \ldots, k_{i_r}\}$ now select the available sellers according to the lowest distance to their location. Therefore, the mapping

$$\tau : \{i_1, \ldots, i_r\} \longrightarrow \{0, j_1, \ldots, j_l\}$$

will be defined as follows:

- $\tau(i_h) = j_f$ means that buyers k_{i_h} buy the concerning good from seller v_{j_f}.
- $\tau(i_h) = 0$ means that for buyer k_{i_h} no seller is remaining. This might happen when the surplus at price p^* is negative and means that a buyer surplus exists.
- $\tau(i_h) \neq j_f \ \ \forall\, i_h \in \{i_1, \ldots, i_r\}$ means that for seller , v_{j_f} no buyers are remaining to that the good could be sold. This might happen if the surplus at price p^* is positive, thus it exists a seller surplus.

The allocation of a seller to a buyer k_{i_h} occurs as follows:

- First case: $\{j_1, \ldots, j_l\} \setminus \{\tau(i_1), \ldots, \tau(i_{h-1})\} \neq \emptyset$, means that at least one seller is remaining.
  Now is $\tau(i_h) \in \{j_1, \ldots, j_l\} \setminus \{\tau(i_1), \ldots, \tau(i_{h-1})\}$, so that

  $$d(k_{i_h}, v_{\tau(i_h)}) = \min\big\{d(k_{i_h}, v) \ \big|\ v \in \{v_{j_1}, \ldots, v_{j_l}\}$$

  $$\setminus \{v_{\tau(i_1)}, \ldots, v_{\tau(i_{h-1})}\}\big\} \ .$$

  Yet, the set of available sellers is

  $$v \in \{v_{j_1}, \ldots, v_{j_l}\} \setminus \{v_{\tau(i_1)}, \ldots, v_{\tau(i_{h-1})}\} \ ,$$

  among these the buyer k_{i_h} will be selected who shows the lowest distance to the seller.
- Second case: $\{j_1, \ldots, j_l\} \setminus \{\tau(i_1), \ldots, \tau(i_{h-1})\} = \emptyset$, i.e. no seller is remaining.
  In this case holds

  $$\tau(i_h) = 0 \ .$$

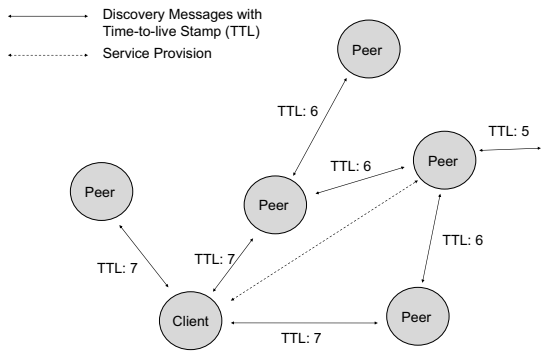

Fig. 2. Decentralized Service Discovery

IV. The Catallaxy as an Alternative Decentralized Approach

Having defined a formal model for using a central matchmaker in a Grid network, this section describes an alternative, decentralized model. This decentralized matchmaking mechanism implements the selection decision in the requesting client itself. Related realizations of decentralized matchmaking are found in P2P Networks, where Gnutella [11] is a typical example.

An optimization of network performance is out of the scope of the clients behavior; in contrast, the selfish conduct of each peer leads to performance and congestion problems in the P2P network, which are principally hard to solve [11]. Gnutella uses a flooding algorithm for service discovery (see Figure 2).

P2P algorithms are thus giving no guarantees whether a service is available at all, and their use of bandwidth is highly inefficient, as they were in 2004 responsible for approx. 56% of all network traffic measured by ISPs and 20% of the backbone traffic, counting only the search requests [12]. Innovative P2P applications like Chord, Pastry or CAN avoid this message abundance by introducing an overlay structure [13].

In decentralized matchmaking models, agents communicate directly with each other, decide on their own, and do not take the system state into account. In the Edgeworth process [14], economic subjects trade bilaterally with each other only if their utility is supposed to increase after the barter. In that case, the sum of all utilities increases after each successful barter; the final state is Pareto-optimal and has maximum system utility.

A theoretical fundament for how dynamic market processes, heterogeneous agents and choice under incomplete information works together, can be found in Neo-Austrian Economics, in particular in Friedrich August von Hayeks Catallaxy concept [15]. Catallaxy describes a state of spontaneous order, which comes into existence by the community members communicating (bartering) with each other and thus achieving a community goal that no single user has planned for.

The implementation of Catallaxy, described in this paper, uses efforts from both agent technology and economics, notably agent-based computational economics [16]. Autonomous software agents negotiate with each other using an alternating offers protocol [17] and adapt their negotiation strategies using feedback learning algorithms (evolutionary algorithms, Numerical optimization e. g. Nelder/Meads simplex method [18], hybrid methods e.g. Brenners VID model [19]). Ongoing communication by using price signalling leads to constant adaptation of the system as a whole and propagates changes in the scarcity of resources throughout the system. The resulting patterns are comparable to those witnessed in human market negotiation experiments [20] [21] [22].

A. Setup and Variables Definition

While the notation for buyers, sellers and goods is the same as in the centralized matchmaker case, we need to add some definitions for the decision-making process (the strategy) of the agents. The negotiation strategy described here is based on the AVALANCHE strategy [23] [24]. The strategy consists of 5 basic parameters, which define the individual behavior (genotype) of each agent.

For every tradeable good there are two types of agents, buyer and seller. Let agent k be a buyer and agent v a seller of a tradeable good.

Let i_k be the number of negotiations, agent k has started and i_v the number of negotiations, agent v has started. It is irrelevant how many negotiations were finished successfully.

A genotype defines the behavior of the agents in the negotiation strategy. Let the genotype of agent k during his negotiation i_k be

$$G_k^{i_k} \in [0;1]^5$$

with

$$G_k^{i_k} = (G_{k,1}^{i_k}, \ldots, G_{k,5}^{i_k})^\tau = (a_k^{i_k}, s_k^{i_k}, t_k^{i_k}, b_k^{i_k}, w_k^{i_k})^\tau$$

where

$a_k^{i_k}$	acquisitiveness
$s_k^{i_k}$	satisfaction
$t_k^{i_k}$	priceStep
$b_k^{i_k}$	priceNext
$w_k^{i_k}$	weightMemory.

Acquisitiveness defines the probability of sticking with the last offer made, and not to make an unilateral concession in the following negotiation step. The value interval is between 0 and 1, and will be challenged by a stochastic probe in every negotiation step. A value of 0.7 means a probability of 70% that the agent will not make a concession – a highly competitive strategy. An agent with acquisitiveness value 1.0 will never change his price and an agent with *acquisitiveness* value 0.0 will always make an unilateral concession. If the probe succeeds, a buyer agent will rise his offer, a seller agent will lower his price.

The exact change of the bid value is defined by the concession level (*priceStep*). The concession level is

represented by a percentage of the difference between the initial starting prices. A value of *priceStep* = 0.25 means a computation of the concession level as 1/4 of the first stated difference. If both opponents are homogenously negotiating and always concede, they meet each other on the half way in the third negotiation round under the assumption of no negotiation abortion.

Obviously, with an *Acquisitiveness* level set high, and a *priceStep* set low enough, the opponents might never reach an agreement. The *Satisfaction* parameter determines if an agent will drop out from an ongoing negotiation. The more steps the negotiation takes, or the more excessive the partner's offers are, the sooner the negotiation will be discontinued. Effectively, this parameter creates time pressure. Like for *Acquisitiveness*, it does this by doing a stochastic probe against a set value between 0 and 1. A *satisfaction* value of 0.75 means, that the agent has a chance of 75% to continue the negotiation process. An agent with *satisfaction* = 0.0 will abort all negotiation at once and an agent with *satisfaction* = 1.0 will never abort.

The last piece of the strategy is an expression of selfishness. Behind each successful negotiation lies a future opportunity for gaining more of the utility share, by negotiating harder. *priceNext* thus modifies the starting bid. A successful seller will increase his offer price, a successful bidder will start with a lower bid next time.

For a viable strategy, the participants will have a close eye on what others deem to be the market price. If not, they risk being tagged as "excessive" and their bids will fail the *satisfaction* probe. They thus weigh current price information and historic price information in a specified ratio *weightMemory*, balancing short-time price fluctuation and longer-term opportunities.

In a formal representation, the genotype of an agent v during his negotiation i_v is

$$G_v^{i_v} \in [0;1]^5$$

with

$$G_v^{i_v} = (G_{v,1}^{i_v}, \ldots, G_{v,5}^{i_v})^\tau = (a_v^{i_v}, s_v^{i_v}, t_v^{i_v}, b_v^{i_v}, w_v^{i_v})^\tau \ .$$

At the beginning of the simulation the genes $G_{*,j}^{i_*}$ for $* = k, v$ and $j \in \{1, \ldots, 5\}$ are distributed according to the probabilities:

$$\text{Ufo}\big([m_j - \delta_j \,; m_j + \delta_j]\big)$$

Thereby, the constants m_j and δ_j for $j \in \{1, \ldots, 5\}$ are defined so that $[m_j - \delta_j \,; m_j + \delta_j] \subset [0;1]$.

Additionally, the agent k has the following variables:

$M_k^{i_k}$	is the market price, which is estimated by agent k during his negotiation i_k.
$P_k^{i_k}$	is the price of the the last *successful* negotiation $1, 2, \ldots, i_k$ of agent k.
$O_k^{i_k}$	is the last offer, which the negotiation opponent has made in negotiation number i_k the agent k before the negotiation ended.
$p_k^{i_k}$	is the number of stored plumages of agent k direct after his negotiation i_k.

The related variables of agent v are defined in the same way. These variables build the basis for decision making during a negotiation.

B. The Negotiation Strategy

When agent k and agent v negotiate, agent k is the buyer and agent v the seller. The sequence $(P_j)_{j \in \mathbb{N}_0} \subset [0, \infty[$ constitutes the offer in chronological order. The buyer always makes the first offer. This means, all offers

$$P_{2m} \qquad \forall\, m \in \mathbb{N}_0$$

originate from the buyer and the offers

$$P_{2m+1} \qquad \forall\, m \in \mathbb{N}_0$$

come from the seller.

At the beginning of a negotiation the buyer k determines his initial price $\underline{K}$ and his maximum price $\overline{K}$:

$$\underline{K} = M_k^{i_k} \cdot (1 - b_k^{i_k}) \,, \qquad \overline{K} = M_k^{i_k}$$

The seller v determines his starting price $\overline{V}$ and his minimum price $\underline{V}$:

$$\overline{V} = M_v^{i_v} \cdot (1 + b_v^{i_v}) \,, \qquad \underline{V} = M_v^{i_v}$$

The buyer starts with the first bid:

$$P_0 = \underline{K}$$

First Case: $\underline{K} \geq \overline{V}$
Then v offers also

$$P_1 = \underline{K}$$

and the negotiation will be closed successfully to the price P_1.

Second Case: $\underline{K} < \overline{V}$
Then v offers his initial price

$$P_1 = \overline{V} \ .$$

Both agents determine now her steps $\delta_*^{j_*}$ for price concessions:

$$\delta_*^{i_*} = (\overline{V} - \underline{K}) \cdot t_*^{i_*} \qquad \text{for} \quad * = k, v$$

In the subsequent negotiation rounds, let $A_1, A_2, A_3, \ldots$ and $S_1, S_2, S_3, \ldots$ be stochastic independent random variables with the following binomial distributions:

$$A_{2m} = \begin{cases} 1 & \text{with probability } a_k^{i_k} \\ 0 & \text{with probability } 1 - a_k^{i_k} \end{cases} \qquad \forall\, m \in \mathbb{N}$$

$$A_{2m+1} = \begin{cases} 1 & \text{with probability } a_v^{i_v} \\ 0 & \text{with probability } 1 - a_v^{i_v} \end{cases} \qquad \forall\, m \in \mathbb{N}$$

$$S_{2m} = \begin{cases} 1 & \text{with probability } s_k^{i_k} \\ 0 & \text{with probability } 1 - s_k^{i_k} \end{cases} \qquad \forall\, m \in \mathbb{N}$$

$$S_{2m+1} = \begin{cases} 1 & \text{with probability } s_v^{i_v} \\ 0 & \text{with probability } 1 - s_v^{i_v} \end{cases} \qquad \forall\, m \in \mathbb{N}$$

• Offer number $2m$; it is the buyer's k turn:

If $S_{2m} = 0$ and $P_{2m-1} \geq P_{2(m-1)-1}$ with $m \neq 1$, then the buyer k cancels the negotiation. This means, $O_k^{i_k} = P_{2m-1}$ and $O_v^{i_v} = P_{2(m-1)}$.

Otherwise, the buyer k makes the following offer:

$$P_{2m} = \Big(\max\left\{\overline{K}, (P_{2(m-1)} + \delta_k^{i_k}), P_{2m-1}\right\}\Big)^{A_{2m}} \cdot \Big(P_{2(m-1)}\Big)^{1-A_{2m}}$$

• Bid number $2m+1$; it is the seller's v turn: If $S_{2m+1} = 0$ and $P_{2m} \leq P_{2(m-1)}$, then the seller v cancels the negotiation. That means, $O_v^{i_v} = P_{2m}$ and $O_k^{i_k} = P_{2(m-1)+1}$.

Otherwise the seller v makes the following offer:

$$P_{2m+1} = \Big(\min\left\{\underline{V},\ (P_{2(m-1)+1} - \delta_v^{i_v}),\ P_{2m}\right\}\Big)^{A_{2m+1}} \cdot \Big(P_{2(m-1)}\Big)^{1-A_{2m+1}}$$

The negotiation ends if either one of the agents cancels the negotiation or the negotiation ends successfully with

$$P_j = P_{j+1}$$

for a $j \in \mathbb{N}$. In this case, it holds $O_k^{i_k} = P_j = O_v^{i_v}$. With the end of a *successful* negotiation to the price P_j the negotiation compute their estimated *profit*

$$\Pi_k^{i_k} = M_k^{i_k} - P_j \qquad \text{respectively} \qquad \Pi_v^{i_v} = P_j - M_v^{i_v} \ . \quad (1)$$

Additionally, both agents update after *every* negotiation their estimated market price using

$$M_k^{i_k+1} = w_k^{i_k} \cdot O_k^{i_k} + (1 - w_k^{i_k}) \cdot M_k^{i_k}$$

respectively

$$M_v^{i_v+1} = w_v^{i_v} \cdot O_v^{i_v} + (1 - w_v^{i_v}) \cdot M_v^{i_v} \ .$$

This last step is independent of the success of a negotiation.

C. Gossip Learning

The learning concept used in this simulation is derived from so-called gossip learning. This means that the agents learn from received information about other transactions in the market. This information may not be accurate or complete, but serves as an indication about the gross direction of the market. In our implementation, this gossip information is created and broadcast by a successful agent, in analogy to issuing an ad-hoc information in stock market periodicals.

Let n be an agent and

$$g_1, \ldots, g_d$$

the tradeable goods. The agent n has finished his negotiation i_n successfully with an estimated profit of $\Pi_n^{i_n}(g)$ for the good $g \in \{g_1, \ldots, g_d\}$. A *learning* step according to the learning algorithm (see subsection IV-D) is performed by agent n last time at the end of his negotiation j_k. This means

$$G_n^{j_n+1} = G_n^{j_n+2} = \cdots = G_n^{i_n}.$$

If agent n with the negotiation numbers

$$j_n + 1, j_n + 2, \ldots, i_n$$

has successfully completed at last 10 negotiations for every good, he sends a *Plumage*

$$(G_n^{i_n}, F_n^{i_n})$$

to all other agents of his type. Then, his updated *fitness* is $F_n^{i_n}$, which is computed like the following:

(a) For every good $g_j \in \{g_1, \ldots, g_d\}$ the next profit value $\Pi(g_j)$ is determined t: Let be

$$\Pi_1(g_j), \ldots, \Pi_{10}(g_j)$$

the estimated profits of the last 10 successful negotiations of agent n for the good g_j. Then, the fitness is

$$F_n^{i_n}(g_j) = \frac{1}{10}\Big(\Pi_1(g_j) + \cdots + \Pi_{10}(g_j)\Big) \ .$$

(b) The updated fitness $F_n^{i_n}$ finally is

$$F_n^{i_n} = \frac{1}{d}\Big(\Pi(g_1) + \cdots + \Pi(g_d)\Big) \ .$$

The agents used in the simulations for this paper are only able to negotiate one type of good ($d = 1$).

D. The Learning Algorithm

After having received some gossip information message, the agent may modify his own strategy. Comparing his own results with those of the strategy received, can lead to recognizing that the other strategy is much better than his own. In this case, the agent will try to cross both strategies to gain competitive advantage. In practice, out of a list of received genotype/performance-tuples, the agent will choose the best performing external genotype, and then mix, cross and mutate with his own genotype.

Let be n an arbitrary agent at the end of his negotiation i_n and let be $p_n^{i_n}$ the number of plumages, the agent n has stored directly after his negotiation i_n. The last *learning* step was performed by agent n after his negotiation j_k. Let be $e_n^{i_n}$ the number of negotiations, an agent n of the negotiation numbers

$$j_n + 1, j_n + 2, \ldots, i_n$$

has successfully finished.

Let be

$$p = 1 \ . \quad (2)$$

If

$$p_n^{i_n} < p \qquad \text{or} \qquad e_n^{i_n} < 10$$

applies for agent n after his negotiation i_n, no *learning* step will be performed. This means, his genotype will not change:

$$G_n^{i_n+1} = G_n^{i_n} \ .$$

Hence, if

$$p_n^{i_n} \geq p \qquad \text{and} \qquad e_n^{i_n} \geq 10$$

applies, the agent n performs a *learning* step. The genotype of agent n changes like the following:

First, the stored plumage of agent n with the highest fitness is selected. Let be

$$G_f = (G_{f,1}, \ldots, G_{f,5})^\tau = (a_f, s_f, t_f, b_f, w_f)^\tau$$

the related genotype. Second, a *crossover* is performed. In doing so, a new genotype $\tilde{G}_n^{i_n+1}$ is created, which contains a random mixture of genes of the genotypes $G_n^{i_n}$ and G_f. This process follows a *mutation* step third: Using the genotype $\tilde{G}_n^{i_n+1}$ and changing its genes slightly will result in the genotype $G_n^{i_n+1}$.

1) Crossover: Let be $C_1, \ldots, C_5$ stochastic independent random variables with the following binomial distribution:

$$C_j \quad = \quad \begin{cases} 1 & \text{with probability} \quad 0,5 \\ 0 & \text{with probability} \quad 0,5 \end{cases} \qquad \forall\, j \in \{1, \ldots, 5\}$$

Then it is imperative

$$\tilde{G}_{n,j}^{i_n+1} = (1 - C_j) \cdot G_{n,j}^{i_n} + C_j \cdot G_{f,j} \qquad \forall\, j \in \{1, \ldots, 5\}\,.$$

2) Mutation: Let be $M_1, \ldots, M_5, X_1, \ldots, X_5$ stochastic independent random variables with the following distributions:

$$M_j \quad = \quad \begin{cases} 1 & \text{with probability} \quad 0,05 \\ 0 & \text{with probability} \quad 0,05 \end{cases} \quad \forall\, j \in \{1, \ldots, 5\}$$
$$X_j \quad \sim \quad \mathcal{N}(0\,,\,1) \qquad \forall\, j \in \{1, \ldots, 5\}$$

That means, X_j is $\forall\, j \in \{1, \ldots, 5\}$ standard normal distributed.
Then, it holds

$$G_{n,j}^{i_n+1} = \max\Big\{0\,;\, \min\big\{\tilde{G}_{n,j}^{i_n+1} + M_j \cdot \big((\frac{1}{10} X_j) \bmod(1)\big)\,;\, 1\big\}\Big\}$$

$\forall\, j \in \{1, \ldots, 5\}\,.$

V. Simulation and Evaluation

Having now a formalized description of both a centralized and decentralized allocation mechanism, this section evaluates their performance in a simulation environment. The simulation runs in terms of varying numbers of participants (requesters and resources) and computing requests.

The fundament of the simulation environment is the network simulator J-Sim. J-Sim is an object oriented, components-based package for the discrete simulation of freely definable topologies [25]. The network attributes (like protocols, bandwidths, loss rates etc.) can be freely determined. In our case, messaging is based on UDP/IP and software agents are attached to the nodes to simulate clients, nodes and services. The configuration is easily described using Tcl/Tk-scripts, which are generated dynamically.

This simulation environment used has been implemented first in the CatNet project [26] and is consequently enhanced and extended. To evaluate the simulation findings, we first need to create a metrics reference framework, which is the topic of the following subsection.

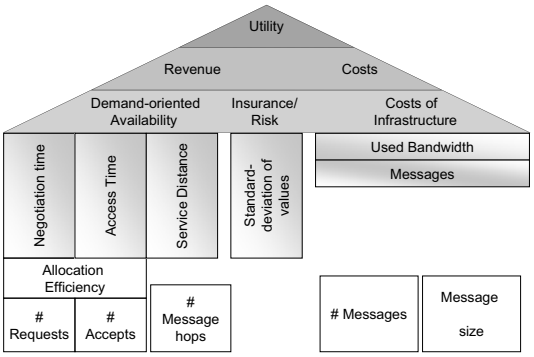

Fig. 3. Matching technical and economical metrics

A. Inferring Economic from Technical Metrics

The behavior of a matchmaking mechanism could principally be evaluated by technical and economic metrics, or a combination of both.The economic criteria are mostly inferred from the technical parameters. Directly measurable metrics are

- the resource allocation rate, which counts the number of accomplished contracts in relation to launched requests,
- costs of second source management,
- the time duration between the launch of the initial request and the successful allocation of the requested resource, and
- the amount of necessary messages for the allocation process (infrastructure costs).

In the considered context, these technical metrics are transferable to economic metrics:

- An increased allocation rate leads to a higher number of contracts and thus to an increase in the requestors' welfare utility, as the needs of the agents are satisfied more often.
- The allocation efficiency describes the ratio of satisfied to initiated requests. This metric can be associated with the opportunity costs of failure of the network infrastructure, in particular the risk for not getting a service in time.
- The longer a client has to wait until it can access a service, the higher is the risk that a desired schedule will be missed. Insofar, lower waiting times decrease opportunity costs and thus add to the client's welfare.

Exogenous shocks like failures of resources or usage peaks negatively impact the performance of the system, which also shows in economic terms. A system that recovers faster allows the expectation of higher profits in comparison to a system that does not recover rapidly. If sufficient values of the economic metrics can be guaranteed and a system can be regarded as more reliable in performance, this will reduce the costs for risk management, both on the side of the service providers and the clients.

Figure 3 presents a metrics pyramid that combines the technical and economic metrics. Utility is expressed

by computing $revenue$ - $costs$. These are affected by infrastructure costs and by changing levels of resource availability.

Infrastructure costs can be roughly measured by counting the number of messages multiplied by the message size, which determines the bandwidth consumption.

Revenue is closely related to availability; service providers being open for business. Availability is impacted by long negotiation times and sluggish access times. Responsible for these can be either network problems, leading to message loss, or negotiation problems, marked by non-matching bids and offers. The occurrence of message loss is assumed to be stochastically related to the distance between provider and client, quantifiable by the number of message hops. Non-matching bids and offers also lead to long negotiation duration, measurable by the ratio of request to accept communication primitives and defined as allocation efficiency.

The risk of non-availability can principally be mitigated through an insurance. Insurance and risk costs are influenced by the standard deviations of the technical metrics.

The exact setup of the pyramid is still part of on-going research, so in the remainder of this article, only the technical metrics at the lowest level are taken into account.

B. Simulation Setup of Network Attributes

For the simulation, the matchmaker instance was embedded in a service network consisting of 106 nodes, with 75 clients on the edges, and 300 redundant service/resource instances, randomly distributed in the network (see Figure 4). The network topology has been arbitrarily chosen. We now simulate five different physical resource concentration setups and five degrees of service availability (see Table 1 and Table 2). In total, we thus have 25 scenario combinations to experiment with, to pitch 2 resource allocation methods against each other – 50 simulation runs for one network with a fixed number of participants.

Some of the scenarios mirror existing, real-world networks. In Content Distribution Networks or Edge Networks, Web sites are replicated among few servers, but requested by millions of web clients, like *Akamai* [27]. In the simulation, this scenario is represented by density level 0. In Ubiquitous Computing scenarios, representing huge numbers of small, mobile devices with only a small number of services on each, resources are often widely distributed and on each site/node just one computing resource is located. Density level 4 represents this scenario.

Levels 1 to 4 then depict the transition from concentration to complete dispersion. Dynamicity level 0 defines total stability - reliable nodes and links of a service network, almost guaranteeing the availability of resources. A real-world example are LANs, where the owners can maintain the availability through organizational actions (e.g. enterprise networks). Dynamicity level 4 constitutes a highly variable network, in which nodes and links often fail. This behavior can be noticed in peer-to-peer (P2P) and file sharing applications, with nodes frequently leaving and re-entering the network.

To scale the simulation to an increasing number of participants in the network, we can add a second and a third ring (see Figure 4). Within one experiment, the 75 clients issue 2000 requests for service instances. When using two rings, 150 clients launch 4000 requests and so on. The duration of the resource usage is fixed to 50ms. The interval between the requests is 75ms, so that the system is always under load (sensitivity experiments can be found at [26]).

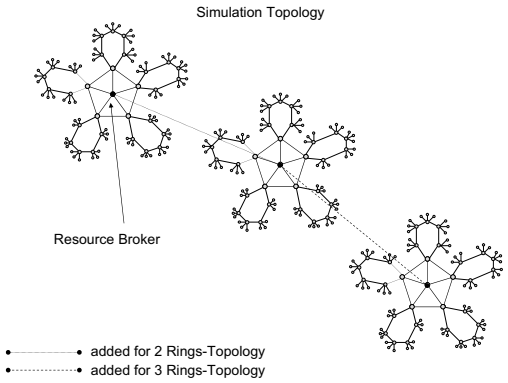

Fig. 4. Simulation network topology

TABLE I
DENSITY DISPERSION OF THE 300 SERVICE INSTANCES PER RING

Level of Density	Number of nodes containing amount of services
0	6 Nodes each with 50 services
1	15 Nodes each with 20 services
2	25 Nodes each with 12 services
3	50 Nodes each with 6 services
4	75 Nodes each with 4 services

C. Simulation Results for the Centralized Approach

This section presents the results for the two metrics allocation rate (RAE) and response time (REST).

1) Resource Allocation Efficiency: Figure 5 shows the allocation rate RAE, in dependence to the level of dynamicity and the number of rings, when fixing the resources to few locations in the network (Density 0). In a stable network (Dynamicity 0), we can reach an allocation rate up to 93%, which decreases significantly when increasing dynamicity. The reason for this shortfall of successful allocations is the increasing problem of the resource broker to produce a still

TABLE II
DYNAMICITY PROBABILITY OF SERVICE FAILURE (TO BE MEASURED EVERY 200MS)

Level of Dynamicity	Probability of failure
0	0 %
1	15 %
2	30 %
3	45 %
4	60 %

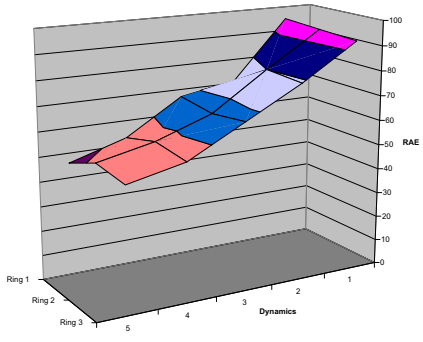

Fig. 5. Centralized: Resource Allocation in Density 0

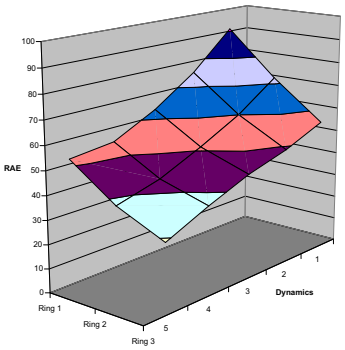

Fig. 6. Centralized: Resource Allocation in Density 4

applicable solution set for a changing problem. In the scenario Dynamicity 4 / Density 0, only 51% of the requests can be successfully allocated.

Obviously, increasing the number of participants by attaching ring 2 and 3 does not have a significant effect. The curve smoothes a little, which can be explained by a greater selection choice of the matchmaker; if a resource fails, he can easily select another resource that is still available.

The picture changes when distributing the resources over more network nodes. Figure 6 shows the allocation rate in density level 4 (The aspect of the diagram has changed to present a clearer view). Starting only 75 clients in ring 1 achieves a rate of more than 90%, while increasing the number of participants by attaching ring 2 and 3 is responsible for a clear decrease of the allocation rate. This is not surprising: the less system elements, the easier can the resource broker compute an optimum allocation. Both with increasing dynamicity and distribution of system elements, complexity increases. Increasing dynamicity shows a continuing loss of the rate down to 31% when running 225 clients in 3 rings.

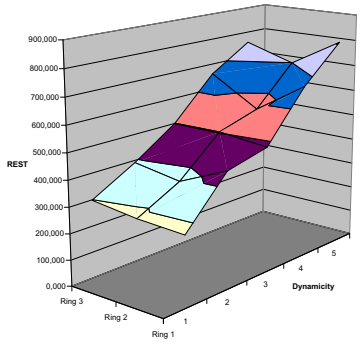

Fig. 7. Centralized: Response Time in Density 0

2) Response Time: The response time (REST) is an important metric from the client's perspective, as it represents the waiting time when requesting a desired computing resource. It is measured here from the initial request of the client until the final delivery of the service, and thus possibly spans over several failed attempts.
In a network with services and resources distributed over a few nodes (Density 0), the response time develops as shown in Figure 7. Changing the dynamicity has a significant impact on the response time. The explanation is that higher dynamics clearly lead to more failed allocations and therefore to time-consuming new requests and allocations. Adding more rings and clients seems not to have a great impact on the measurement, aside from a small valley for ring 2. This might be a result of two influences. First, the rising number of participants leads to more available service providers, enabling the selection algorithm to allocate more efficiently. However, an increasing network topology (ring 3) leads to higher response times, as misallocations increase and the distance between the agents increases.

Figure 8 shows the response time for density level 4, which means that resources are distributed all over the network. The picture is now very different, compared to Figure 7. The response time is very sensitive to the number of network elements, but not to the dynamics. Higher distribution in the network also means more alternative routes between clients and services, so that one failure does not change much. Larger distances however do matter significantly, because there is a higher probability that any link in a given route fails, which then leads to new requests and a long time until the resource service is successfully enacted.

D. Simulation Results for the Decentralized Approach

1) Resource Allocation Efficiency: Figure 9 shows the graphical development of the allocation rate in density level 0 for a decentralized scenario without resource broker (note the changed aspect to present a clearer view of the diagram).

Still, the best results can be obtained for networks with a low level of dynamicity. Values over 85% can be achieved.

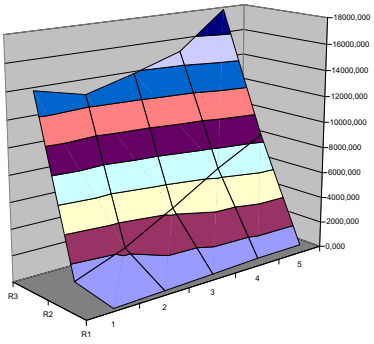

Fig. 8. Centralized: Response Time in Density 4

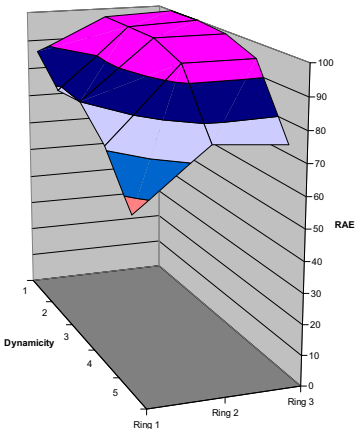

Fig. 9. Decentralized: Resource Allocation in Density 0

Increasing the level of dynamicity immediately shows a decrease in the allocation rate.

Given a failure probability of 60% (dynamics level 4), less than 55% of the requests are saturated. However, a noticeable increase is found when more rings are attached. With both ring 2 and 3 attached, nearly 99% of the service requests can be fulfilled in the lower dynamics scenarios.

The allocation rate increases about 20%, compared to the simulation run with only 1 ring and nearly 35% higher than with a centralized resource broker. When dynamicity is high, the rate drops down to 75% – which is still 23% more than with using a broker. As a caveat, this might be a result of the increased number of market participants which balances demand and supply on the market – a subject of future investigation.

Figure 10 shows the highest density level of 4. It is obvious that an increase of dynamicity affects the rate slightly from 81% to 74%. Attaching ring 2, the rate increases to 95% (dynamicity 0) and 88% (dynamicity 4), ameliorating the results of the centralized approach approximately 50% in dynamicity level 4. Running the decentralized strategy in a third ring, results seem to be similar to the ones obtained from two rings whilst in the centralized case results become even worse.
It seems that a higher number of market participants enables an equilibrium, allowing the system to reach a more stable allocation rate and cope easily with demand or supply fluctuations.

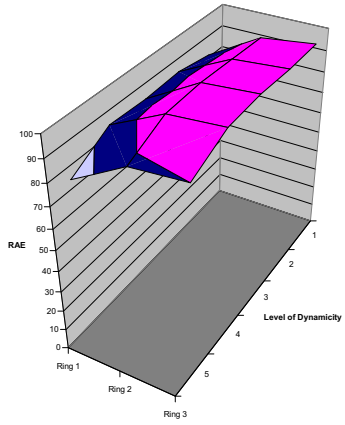

Fig. 10. Decentralized: Resource Allocation in Density 4

2) Response Time: Figure 11 shows the response time results of the decentralized approach with density = 0. Comparing to Figure 7 – the centralized approach – it is obvious that the data is completely different: The range of the results is wider in the centralized case. For dynamicity = 0, the centralized reponse times are about 300ms, which is unbeatable. Even in that static case, the decentralized approach will take longer because of the more complex negotiation strategy to be processed.

When increasing dynamicity, the centralized approach shows deteriorating results in Figure 7, as was expected. In the decentralized case, we see a slight minimum in the simulation runs with 2 rings, and much less increase in the 3-ring simulation. This is probably caused by a inverse scalability effect: more participants lead to averaging message latency, the more so because the average distance does not principally change, even when adding more rings. Dynamicity has thus a marginal effect on the response times.

This becomes even clearer when increasing density. The response time in density 4 (see Figure 12) seems to be lower in all ring topologies. Response times in the range of 800ms up to 1200 in density 0 are replaced by times between 400ms and 550ms in density 4. This is a consequence of the ubiquity of the services, that is that they are distributed widely.

The effect of message latency in dependence of the topology is nearly not visible: Response times decrease moderately and dynamicity does not affect the results much.

Overall, a REST of approx. 400ms is the maximum for the decentralized case here. This is probably a consequence of the search algorithm which has to flood the whole network to get an adequate number of replies. There is also a maximum visible at the higher density levels, which seems to be nearly

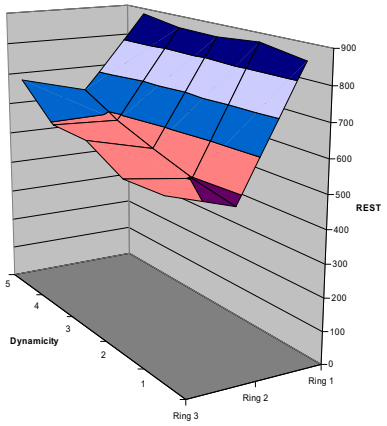

Fig. 11. Decentralized: Response Time in Density 0

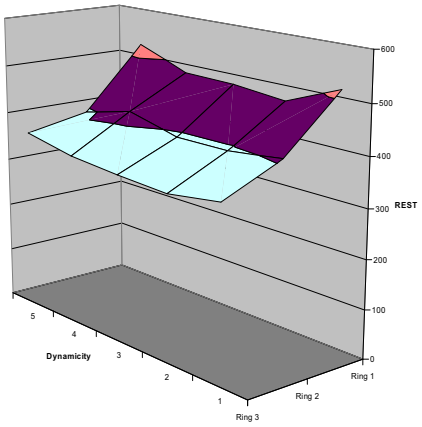

Fig. 12. Decentralized: Response Time in Density 4

independent of the dynamics level and lies below 520ms. In comparison to the centralized approach, these values are lower, more stable and therefore better. An improvement for the decentralized approach in low density cases might lie in a revised discovery algorithm (e. g. CAN, Chord).

VI. Conclusion and Outlook

This paper focuses on scalability issues of resource management mechanisms in Grid Computing systems. It can be shown how different resource allocation mechanisms are effected by varying dynamics and resource density in computing networks. In short, using a broker faces problems with scalability and a greater inability to cope with network dynamics, like link and node failures, when increasing the number of participants.

A self-organizing, decentralized "Catallactic" approach shows more robust behavior when varying input parameters. However, the simulation results need to be investigated in more detail. The modeling of the centralized matchmaker might be oversimplified, as the used allocation mechanism is rigid and does not take dynamics into account. Using decentralized mechanisms shows a good scalability and promises to provide stable allocation results.

Acknowledgements

This work has been partially funded by the European Commission under grant FP6-IST-003769 "CATNETS" [29]. The simulation environment was implemented first in the CatNet project under grant IST-2001-34030 [28].

References

[1] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef, "Web services on demand: Wsla-driven automated management," *IBM Syst. J.*, vol. 43, no. 1, pp. 136–158, 2004.

[2] M. A. Rappa, "The utility business model and the future of computing services," *IBM Syst. J.*, vol. 43, no. 1, pp. 32–42, 2004.

[3] A. Byde, M. Salle, and C. Bartolini, "Market-based resource allocation for utility data centers," 2003. [Online]. Available: citeseer.ist.psu.edu/byde03marketbased.html

[4] R. Buyya, J. Giddy, and D. Abramson, "A case for economy grid architecture for service-oriented grid computing," in *Proceedings of the 10th IEEE International Heterogeneous Computing Workshop (HCW 2001), in conjunction with IPDPS 2001, San Francisco, California, USA*, 2001.

[5] Rechenkraft.net, "http://www.rechenkraft.de, download on 2004-07-26."

[6] J. Frey, T. Tannenbaum, M. Livny, I. Foster, and S. Tuecke, "Condor-g: A computation management agent for multi-institutional grids," *Cluster Computing*, vol. 5, no. 3, pp. 237–246, 2002.

[7] P. Chandra, A. Fischer, C. Kosak, E. Ng, P. Steenkiste, E. Takahashi, and H. Zhang, "Darwin: Customizable resource management for value-added network services," in *ICNP '98: Proceedings of the Sixth International Conference on Network Protocols*, 1998.

[8] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *International Journal of Supercomputing Applications*, vol. 11, no. 2, pp. 115–129, 1997.

[9] A. Bestavros, "Demand-based dissemination for distributed multimedia application," in *Proceedings of the ACM/ISMM/IASTED International Conference on Distributed Multimedia Systems and Applications, Stanford, CA*, 1995.

[10] E. Amir, S. McCanne, and R. Katz, "An active service framework and its application to real-time multimedia transcoding," in *Proceedings of ACM SIGCOMM'98. Vancouver, Canada*, 1998.

[11] E. Adar and B. Huberman, "Free riding on gnutella," *First Monday*, vol. 5, no. 10, 2000.

[12] N. B. Azzouna and F. Guillemin, "Charcteristic of ip traffic in commercial wide area networks," in *Proceedings of the International Conference on Computing, Communcations and Control Technologies (CCCT'2004), Austin, Texas (TX)*, August 2004.

[13] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A scalable content-addressable network," Berkeley Press, Tech. Rep., 2001.

[14] H. R. Varian, *Mikrokonomie*. Oldenbourg, 1994.

[15] F. Hayek, W. Bartley, P. Klein, and B. Caldwell, "The collected works of f.a. hayek," *University of Chicago Press*, 1989.

[16] L. Tesfatsion, "How economists can get alife," in *The Economy as a Evolving Complex System II*. Arthur, W.B. and Durlauf, S. and Lane, D.A. (Hrsg.), 1997, pp. 533–564.

[17] G. Rosenschein, J. S.; Zlotkin, "Rules of encounter - designing conventions for automated negotiation among computers," *MIT Press, Cambridge*, 1994.

[18] W. H. Press and S. A. Teukolsky, *Numerical Recipes in C++ - The Art of Scientific Computing*. Cambridge, MA, Cambridge University Press, 2002.

[19] T. Brenner, "A behavioural learning approach to the dynamics of prices," *Computational Economics*, pp. 67–94, 2002.

[20] J. Kagel and A. Roth, *The handbook of experimental economics*. Princeton University Press, 1995.

[21] D. Pruitt, "Negotiation behavior," *Organizational and occupational psychology. New York: Academic Press*, 1981.

[22] V. Smith, "An experimental study of competitive market behavior," *Journal of Political Economy*, vol. 70, pp. 111–137, 1962.

[23] T. Eymann, D. Schoder, and B. Padovan, “Avalanche - an agent based value chain coordination experiment,” in *Workshop on Artificial Societies and Computational Markets (ASCMA’98)*, Minneapolis, 1998, pp. 48–53.

[24] T. Eymann, “Decentralized economic coordination in multi-agent systems,” in *Information Age Economy. Proceedings WI-2001.*, H.-U. Buhl, F. Huther, and A. Reitwiesner, Eds. Heidelberg: Physica Verlag, 2001, pp. 575–588.

[25] T. J.-S. web site, “http://www.j-sim.org.”

[26] T. Eymann, M. Reinicke, F. Freitag, L. Navarro, and O. Ardaiz, “Catnet project: Catallaxy evaluation report. report no. d3. barcelona,” UPC Barcelona and University Freiburg, Tech. Rep., 2003. [Online]. Available: http://research.ac.upc.es/catnet/pubs/D3.pdf

[27] A. Webseite, “http://www.akamai.com,” 2004.

[28] O. Ardaiz, T. Eymann, F. Freitag, L. Navarro, and M. Reinicke, “Catnet: Catallactic mechanisms for service control and resource allocation in large-scale application-layer networks.” in *Proceedings of the IEEE International Symposium on Cluster Computing and the Grid (CCGRID’2002)*. IEEE Computer Society, 2002, pp. 442–443.

[29] T. Eymann and M. Reinicke, “Catnets evaluating catallaxy for application layer networks,” 2003, project Proposal CATNETS for FET Open.

Grid Computing is a paradigmatic application for the requirements associated with the exponentially growing complexity in the engineering, operation and maintenance of today's information systems. Grid Computing comprises evergrowing global communication infrastructures and millions of possible system elements into the picture. This article examines the broker's behavior with regard to a varying number of participating nodes and shows that incremental losses have to be accepted in central resource allocation when introducing new nodes.

ISSN 1864-9300