Nature-inspired Self-organization in P2P Ad hoc Grids

Tariq Abdullah, Koen Bertels,
CE Laboratory, EEMCS, Faculty, Delft University of Technology,
Mekelweg 4, 2624 CD, Delft, The Netherlands
{m.t.abdullah, k.l.m.bertels}@tudelft.nl

***Abstract*—Resource availability fluctuates over time in ad hoc grids and other similar distributed systems due to their dynamism and complexity. These changes require adaptation of the system to the new system state. This paper will present an ant colony based self-managing mechanism in a P2P ad hoc grid. In this paper, we apply ant colony optimization for self-management of a P2P ad hoc grid. The proposed mechanism will enable the ad hoc grid participants to self-organize themselves and form/de-form virtual resource clusters according to their changing resource availability/requirements. We also investigate how can ants form a dynamic cluster/ring of specialized resources. The experimental results show that the proposed mechanism work better than the previously proposed mechanisms.**

***Keywords*–** *Self-organization; natural intelligence; ad hoc grids*

I. Introduction

There are a large number of naturally existing Complex Adaptive Systems (CAS) like fish swim in schools, geese fly in organized V-shaped flocks, immune system, sand dune ripples and ant foraging. These CAS are dynamic, highly decentralized networks and consist of many participating agents and are characterized by decentralized control, emergent behavior, robustness and self-organization. The participating agents in these systems interact according to simple local rules which result in self-organization and complex behavior.

Ad hoc grids and similar computational distributed systems are inherently dynamic and complex systems. Resource availability fluctuates over time in ad hoc grids. These changes require adoption of the system to a new system state by applying some self-organizing mechanism.

We aim to study the different adaptation mechanisms that can be used to introduce self-organization in ad hoc grids (or similar distributed systems) in the context of GRAPPA project [1]. We envision Ant Colony Optimization (ACO) [2] as one way of introducing self-organization in distributed systems under varying resource availability of the autonomous participating nodes of the distributed systems. We are also interested in understanding the effect of ACO based mechanism on the infrastructural spectrum.

We proposed a dynamic, self-organizing mechanism for resource discovery in ad hoc grids in our previous work [3], where, we focused on developing dynamic and scalable mechanisms for resource discovery that could self-organize according to the work load of the resource manager (referred to as *matchmaker* hereafter).

In this paper, we propose an ant colony inspired mechanism for self-organization in ad hoc grid. The proposed self-organizing mechanism also focuses on the dynamic formation of resource cluster(s) according to resource demands. We apply the meta-heuristic methods from ACO for dynamic cluster formation/de-formation and resource virtualization in an ad hoc grid. In these meta-heuristic methods a complex, adaptive behavior emerges from the local interactions of the participating ants. Each ant (a participating node of the ad hoc grid) is looking for its required resources without any centralized control.

The rest of the paper is organized as follows. Related work is presented in Section II. Section III summarizes the micro-economic based modified ACO algorithm and formulas for pheromone calculation presented in our previous work [4]. Proposed nature inspired self-management mechanism in explained in Section IV. Experimental setup & experimental results discussion are in Section V. Whereas, Section VI concludes the paper with some future research directions.

II. Related Work

Jaeger et al. [5] applied bloom filter for self-organizing broker topologies in publish/subscribe systems. Their work is closely related to the work presented in this paper. The main problem with their work is they only considered the similarity of the notification messages in publish/subscribe system to reduce the total cost of forwarding and processing the notification messages. Ritchie et al. [6] proposed a hybrid ant colony optimization algorithm to select the appropriate scheduler in a heterogeneous computing environment. The proposed approach was only tested in solving a scheduling problem in a static environment for independent jobs. Deng et al. [7] proposed a resource discovery mechanism for Peer-to-Peer (P2P) grids inspired by ant colony optimization algorithm. Fidanova et al. [8] attempted searching for the best task scheduling for grid computing using ant colony optimization based algorithm. Zeng et al. [9] proposed a dynamic load balancing mechanism based the count of waiting jobs and the arrival rate of the new jobs in distributed systems. Andrzejak et al. [10] compared different algorithms, including ant colony optimization algorithms, for self-organization and adaptive service placement in dynamic distributed environments. Messor [11] is implemented on top of the Anthill framework [12]. An ant can be in Search-Max or Search-Min states. The ant

 ISBN: 978-1-61208-134-2

wander randomly in the environment it find an overloaded node. The same ant, then, changes its state to Search-Min and wanders randomly again in the environment, while looking for an underloaded node. After these state changes, the ant balances the underloaded and overloaded node. However, considering the dynamism of grid environments, this information may cause erroneous load balancing decision making.

The main contributions of this paper are that this paper proposes an ant colony inspired, micro-economic based, self-organizing mechanism in a P2P ad hoc grid. The proposed mechanism introduces self-organization in the ad hoc grid and enables it to re-organize and adapt to a stable status under varying network and work load conditions. The scalability and robustness of the proposed mechanism is tested on PlanetLab [13].

III. Micro-economic based ACO Algorithm

We presented and explained an ant colony inspired micro-economic based resource management mechanism for ad hoc grids in [4]. The algorithms, formulas and the mapping of an ACS to an ad hoc grid were explained in the perspective of a micro-economic domain. We provide an overview of the mechanism in this section for better understanding the results presented in this paper.

An ad hoc grid node acts like a consumer/producer node in the modified ACO algorithm, and the matchmaker(s) are treated like the *food sources*. Each consumer/producer node is capable of generating and sending *request/offer ants* to the food source. The *pheromone value* indicates the weight of the matchmaker in the ant system. A matchmaker with higher pheromone value indicates that it has a higher probability of finding a compatible resource offer for a submitted resource request and vice verse.

Each joining node in our ad hoc grid is under the responsibility of a matchmaker and sends its resource request/offer ants to its responsible matchmaker. The joining node gets the pheromone value of the food source from its responsible matchmaker. The pheromone value of a matchmaker is updated for each resource request or resource offer received from a consumer/producer node. A matchmaker exchanges periodically its pheromone value with its immediate neighboring matchmakers (the successor and the predecessor matchmakers). The updated pheromone value is then sent to the consumer/producer nodes with a matched message. The consumer/producer node uses the pheromone value as an indicator of matchmaker's matchmaking performance. The consumer/producer node sends its next request/offer ant to a food source with the highest pheromone value. The pheromone value of a matchmaker is calculated periodically according to the following formula [4] given below:

$$\tau_{new} = \begin{cases} \alpha * \tau_{old} + (1-\alpha) * \triangle\tau & if\ \triangle\tau > 0 \\ (1-\alpha) * \tau_{old} + \alpha * \triangle\tau & if\ \triangle\tau < 0 \end{cases} \quad (1)$$

The parameter α represents the pheromone evaporation rate. The value of α varies between 0 and 1. τ_{old} represents the pheromone value during time interval $T1 = [t_{s_1}, t_{e_1}]$. Whereas, $\triangle\tau$ is the change in the pheromone value between the time interval $T1 = [t_{s_1}, t_{e_1}]$ & $T2 = [t_{s_2}, t_{e_2}]$. The start time of both intervals is represented by t_{s_1} & t_{s_2} and t_{e_1} & t_{e_2} represent the end time of both the intervals, such that $T2 > T1$ & $t_{s_2} = t_{e_1}$. Value of $\triangle\tau$ is calculated as:

$$\triangle\tau = \sum_{i=1}^{n} \tau(i)/_N \quad (2)$$

where N is the total number of messages received by the matchmaker and $\tau(i)$ is the pheromone value contributed by an individual ant. $\tau(i)$ for a consumer agent is calculated as:

$$\tau(i) = Perform(MM) * UPrice_{consumer} \quad (3)$$

$\tau(i)$ for a producer agent is calculated as:

$$\tau(i) = Perform(MM) * UPrice_{producer} \quad (4)$$

where $Perform(MM)$ represents the performance of a matchmaker and $UPrice$ represents the unit price of a requested or offered computational resource by an ant. The above formulas are explained in [4].

IV. ACO based Self-managing Ad Hoc Grid Segmentation/De-segmentation

This section explains the proposed ACO based self-managed segmentation and de-segmentation approach in an ad hoc grid. This self-managed, dynamic segmentation and de-segmentation process is based on the dynamic changing resource requirements and resource availability in an ad hoc grid.

In the proposed mechanism, there can be as many segments of the ad hoc grid as many needed. These segments are created when needed and are removed, when not needed. The proposed approach also considers the ad hoc grid segmentation for specialized resource requirements of the participating ad hoc grid nodes. These resources may include a pool of specialized hardware for a video rendering application or a pool of software resources for a data processing application or a pool of resources for remote collaboration on a scientific experiment.

A consumer/producer node knows about its resource category at the time of joining the ad hoc grid. The consumer/producer node joins the ad hoc grid by contacting one of the existing nodes in the ad hoc grid, referred to as a bootstrap node. A consumer/producer node discovers a responsible matchmaker for its resource category. The algorithms for node joining and finding a responsible matchmaker are detailed in [14]. All of the newly joining consumer/producer nodes discover their responsible matchmakers and the nodes of a similar resource category are under the responsibility of one matchmaker. An ad hoc grid node can change its resource category or its resource availability/demand status (being a consumer or producer of resources) at any time during its life span. A consumer/producer node generates and sends a request/offer ant after finding its responsible matchmaker.

 ISBN: 978-1-61208-134-2

Algorithm 1 Discovering the right food source.

1:IF (($Ant_{resCategory}$ == $FS_{resCategory}$) AND ($FS_{phValue}$ is highest)) THEN
2: CALL Find matching request/offer algorithm
3: IF (no match found) THEN
4: IF ($Ant_{resCategory}$ == $pFS/sFS_{resCategory}$) THEN
5: Ant visit pFS/sFS
6: GO TO Step 1
7: END IF
8: END IF
9: ELSE IF ($Ant_{resCategory}$!= $FS_{resCategory}$) THEN
10: Ant visit $successorFS$
11: GO TO Step 1
12: END IF
13:END IF

Algorithm 2 Find matching request/offer.

1:Store request/offer in request/offer repositories
2:Match request parameters with offer parameters
3:IF ($Offer_{parameters}$ match $Request_{parameters}$) THEN
4: Send matched message to consumer
5: Send matched message to producer
6: Update pheromone value
7:END IF

The proposed mechanism comprises the algorithms for *discovering the right food source (matchmaker)*, *finding the matching request/offer* and *changing the segment*. A food source determines whether it is the right food source to process the received resource request/offer ants by executing the algorithm discovering the right food source (Section IV-A). The *right food source* processes the received request/offer ant and attempts finding a matching offer/request ant by executing the algorithm find matching request/offer (Section IV-B). The participating consumer/producer nodes are autonomous and can change their resource category at any time. When they change their resource category then they join the new resource segment. The steps for changing the resource segment of a consumer/producer node are performed by executing the algorithm change segment (Section IV-C).

A. Discovering the Right Food Source

A matchmaker receives a request/offer ant, and first checks whether it is the right food source for processing the received request/offer ant or not. This process is listed in Algorithm 1 and is performed as follows:

Before explaining Algorithm 1, we first introduce the notations used in this algorithm. The resource category of the received request/offer ant is denoted by $Ant_{resCategory}$, resource category of the current matchmaker is denoted by ($FS_{resCategory}$). Whereas, $pFS_{resCategory}$ and $pFS_{resCategory}$ denote the resource category of the successor and predecessor matchmaker resource category respectively. The pheromone value of the current matchmaker is denoted by $FS_{phValue}$.

After receiving a request/offer ant, the matchmaker first checks whether the resource category of the received request/offer ant ($Ant_{resCategory}$) is similar to its resource category ($FS_{resCategory}$). Secondly, the matchmaker also checks whether the predecessor's resource category ($pFS_{resCategory}$) and/or the successor's resource category ($sFS_{resCategory}$) is similar to its resource category. In case of a matched resource category, the matchmaker also checks that it has the highest pheromone value ($FS_{phValue}$).

In case the matchmaker does not have the matching offer/request then the request/offer ant is forwarded to the predecessor/successor matchmaker with the second highest pheromone value. The predecessor/successor matchmaker node processes the received request/offer ant in the same way as explained above. When the resource category of the request/offer ant ($Ant_{resCategory}$) is different from the resource category of the matchmaker ($FS_{resCategory}$) and its immediate neighboring matchmaker nodes (successor and predecessor matchmakers), then the request/offer ant is forwarded to the successor matchmaker node. The successor matchmaker node again performs the steps in Algorithm 1.

B. Find Matching Request/Offer

After determining from its local knowledge that it is the right matchmaker/food source to process the received request/offer ant, the matchmaker attempts finding a matching offer/request for the received request/offer ant by following the steps listed in Algorithm 2. The matchmaker first stores the received request/offer ants in it request/offer repositories. Then the matchmaker/ food source compares the offer parameters ($Offer_{parameters}$) with the constraints/ parameters of received request ($Request_{parameters}$). If a resource offer satisfies the resource request parameters then a matching resource request/offer pair is declared by the matchmaker and the matching consumer/producer nodes are directly notified by the matchmaker about the match[3].

After finding a successful match, the pheromone strength of that matchmaker increases. The pheromone value of a matchmaker is based on its matchmaking efficiency. The pheromone value of a matchmaker decreases, when it is unable to find a match. The matchmaker periodically notifies its immediate neighboring matchmaker nodes (successor and predecessor matchmaker nodes) about the change in its pheromone value. The consumer/producer nodes communicate directly with each other for task execution after receiving a matched request/offer message from the matchmaker. The producer node returns the results back to the consumer node after executing the task.

C. Change Segment

The process of changing a node's segment is triggered when a node changes its resource category. This process is listed in Algorithm 3 and is performed as follows: Whenever a consumer/producer node changes its resource category, it receives a matched message reply from a new matchmaker ($matchMsgSender_{FSID}$). A consumer/producer node

Algorithm 3 Change segment.

```
IF (CPNode_FSID != matchMsgSender_FSID) THEN
    CPNode_FSID == matchMsgSender_FSID
    Join new virtual segment
    Send request/offer ant to CPNode_FSID
ELSE
    No change in virtual segment
    Send request/offer to old CPNode_FSID
END IF
```

changes its matchmaker node ($CPNode_{FSID}$) and sends its next request/offer ant to the new matchmaker. In this way, a consumer/producer node continues sending its subsequent request/offer ants to the matchmaker that sent a matched message reply. Whenever, an ad hoc grid node changes its resource category, it leaves its current virtual segment and becomes a member of another virtual segment. Therefore, the ad hoc grid keeps on changing its infrastructure and is divided into specialized resource segments. The consumer/producer nodes are not bound to a specific ad hoc grid segment. The nodes can dynamically leave one virtual segment and join another according to their changed resource category.

V. Experimental Results and Discussion

Both the experimental setup used for evaluating the proposed mechanism and the experimental results are discussed in this section.

A. Experimental Setup

The modified ACO algorithm is implemented by extending Pastry [15], a structured overlay network. Pastry forms an overlay network among the ad hoc grid nodes and performs the basic tasks required for maintaining an overlay network. The experimental results reported here are obtained by executing the experiments on PlanetLab [13].

These experiments are executed in a different network condition including balanced network (BN) condition, resource intensive network (RIN) condition and task intensive network (TIN) condition. The consumer-producer ratio is approximately $50-50$ in BN condition. Whereas, the consumer-producer ratio is $20-80$ and $80-20$ in RIN and TIN conditions, respectively. The number of participating nodes is varied from $15-650$. The number of different resource categories is 3 in the first set of experiments and number of matchmakers is 5 in the second set of experiments. Different parameters of resource request/offer like task execution time and resource quantity are randomly generated from a prespecified range. The validity period (TTL) of request/offer message is set to 10000 milliseconds for accommodating delays observed in PlanetLab. The results of the BN condition are discussed in detail.

Each ant's pheromone is initialized by 1. In nature, each ant wanders randomly without any initial pheromone value as used in these experiments. However, if the initial pheromone value is set to 0, then the new pheromone value will always be zero.

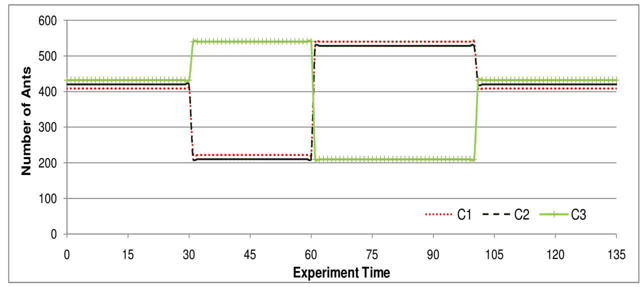


Figure 1: Workload distribution of different resource categories.

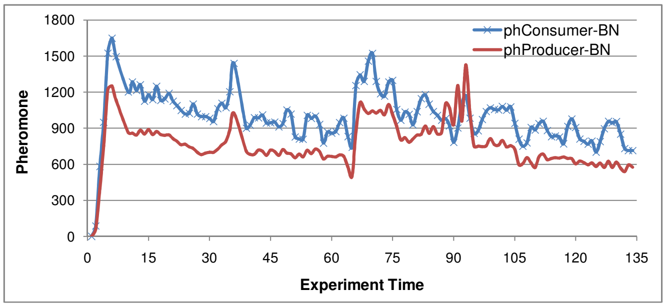
The value of α (rate of pheromone evaporation) is set to 0.8 in these experiments. The workload distribution among different resource categories is shown in Figure 1. In this figure, the experiment time is represented on the horizontal axis, and the number of ants of each resource category is shown on the vertical axis.

The analysis parameters are pheromone evaluation, consumer/producer utilization, response time, and the average ask/bid price of the participating producer/consumer nodes. The formulas for calculating the pheromone value are described in Section III. The consumer utilization and producer utilization are calculated according to the following equation: $(MatchedMessage/N)*100$, where $MatchedMessage$ represents the count of matched messages and N denotes the total number of request/offer ants processed by the matchmaker(s) in a unit time interval. The *response time* represents the time interval between receiving a request/offer message and finding a matching offer/request by the matchmaker. The response time is calculated as:$RT = T_{match} - T_{receive}$, where RT represents the response time, T_{match} is the time when the matchmaker found a matching offer/request for the received request/offer and $T_{receive}$ is the receiving time of the received request/offer. Following simplifying assumptions are in place for the experimental results reported in this paper and will be relaxed in the future work: (1) There exists at least one food source for a resource category. (2) Each consumer/producer node knows about or is under the responsibility of one food source at any given time. (3) The successor/predecessor food sources of a food source node (aka matchmaker node) update the current matchmaker node after updating their pheromone value and vice verse.

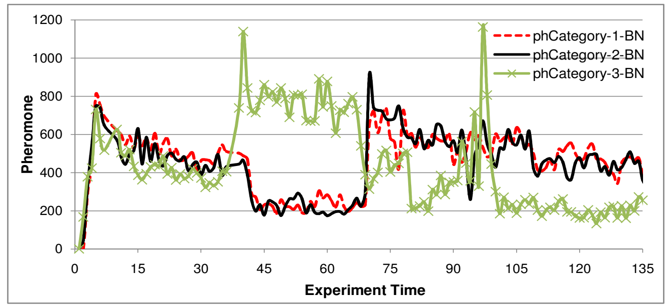
B. Experimental Results Discussion

The proposed ACO-based self-organizing mechanism is compared with different matchmaking schemes. These schemes include the simplest and less compute intensive first come, first served and a micro-economic based, compute intensive continuous double auction scheme.

The overall pheromone evolution for each resource category (represented as $phCategory-1-BN$, $phCategory-2-BN$ and $phCategory-3-BN$) during the simulation in a BN condition is depicted in Figure 2b. The experiment time is represented on the horizontal axis and the pheromone value

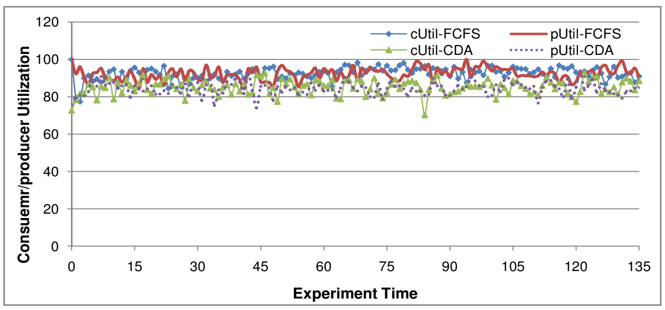


(a)

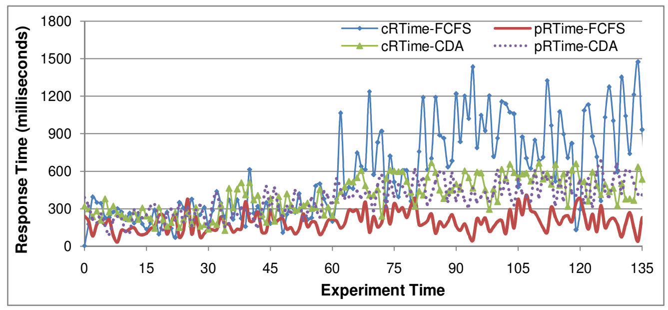


(b)

Figure 2: **(a)** Consumer/producer pheromone evolution in BN condition of the ad hoc grid. **(b)** Individual category pheromone evolution in an ad hoc grid.



(a)



(b)

Figure 3: **(a)** Consumer/producer utilization **(b)** Consumer/producer response time of the in the ad hoc grid in BN condition.

is on the vertical axis. The pheromone of each category evolves according to the workload distribution of the respective resource category (Figure 1). Whenever the distribution of ants in different resource categories is changed, a change in the respective pheromone concentration is observed. The ad hoc grid self-organizes itself after each change. A stable pheromone value is observed before the next disturbance in the ant's distribution during the simulation.

The pheromone behavior of the individual consumer/producer nodes in BN condition in represented in Figure 2a. The horizontal axis of both these figures represent the experiment time and the vertical axis represents the pheromone value of the participating consumer/producer nodes of the ad hoc grid. The initial increase of the pheromone value for the consumer/producer nodes is followed by a decreasing trend that leads to a stable status of the ad hoc grid. Similar to the individual category pheromone value evolution, the overall consumer/producer pheromone also evolves similarly. The pheromone pattern is disturbed after a change of the ant's distribution in different resource categories. The proposed algorithm enables the ad hoc grid to re-structure itself into different virtual resource and in attaining a stable state.

The average consumer/producer utilization of the participating consumer/producer nodes in an ad hoc grid with different matchmaking mechanisms is depicted in Figure 3a. In-spite of changing workload of ants in different resource categories, the consumer utilization ($cUtil - CDA$, $cUtil - FCFS$) and the producer utilization ($pUtil - CDA$, $pUtil - FCFS$) in CDA and FCFS schemes under a BN condition remains above 80%. The fluctuations in the consumer/producer utilization refer to the activity in the ad hoc grid. This behavior implies that the compute intensive nature of CDA does not affect the matchmaking capacity of the ad hoc grid. It can be concluded from the above discussion that, in spite of being compute intensive, the consumer/producer utilization in CDA is as good as in FCFS. Whereas, the effect of an overload condition on a matchmaker in the proposed mechanism and a solution for avoiding the overload condition was discussed in [4].

The proposed self-organizing mechanism can also be analyzed from the matchmaker response time for the consumer/producer nodes. The consumer response time and the producer response time for continuous double auction scheme and first come, first served scheme in a balanced network condition is represented in Figure 3b. The response time shows a stable behavior and is not affected by the resource category change of the participating consumer/producer nodes. The consumer response time ($cRTime{-}CDA$, $cRTime{-}FCFS$) and the producer response time ($pRTime{-}CDA$, $pRTime{-}FCFS$) for both the schemes in BN condition are initially low due to the equal number of ants of different categories. When the consumer/producer nodes change their resource category, the ad hoc grid becomes unstable. The consumer/producer nodes have to wait longer for getting a matched response. This longer wait time results in an increasing trend of the consumer/producer response time. The response time shows a stable behavior once the ad hoc grid attains back the stable behavior.

The consumer response time for FCFS ($cRTime - FCFS$) is higher than that of the consumer response time for CDA ($cRTime - CDA$) in a BN condition. The lower consumer response time in CDA can be understood by understanding the requests/offers handling process in the matchmaker request/offer repositories. The matchmaker stores requests in descending order and offers in ascending order of the price in its request/offer repositories. The consumer response time is less in CDA scheme as compared to the consumer response time in FCFS scheme, due to the sorted placement of requests/offers in the matchmaker repositories.

It can be concluded from the above discussion that a nature inspired ACO-based, self-organizing mechanism with CDA scheme is preferred over an ACO-based mechanism with FCFS scheme. CDA based mechanism performs as good as the FCFS mechanism in terms of consumer/producer utilization, response time and pheromone value. CDA based mechanism enables the individual consumer/producer nodes to value their resource requests and resource offers according to their previous experiences from the ad hoc grid. The consumer/producer nodes can increase/decrease their bid/ask prices according to the resource demand/availability in the ad hoc gird. Thus, a CDA based ACO mechanism enables the node level self-organization along with the system level self-organization.

VI. Concluding Remarks

An ACO inspired, micro-economic based resource management approach for the ad hoc grids is presented in this paper. We used matchmaking performance as the basic factor for calculating the pheromone value. The proposed ACO inspired CDA based approach enables node level as well as system level self-organization and supports resource specialization in an ad hoc grid. From the experimental results it can be concluded that the proposed mechanism gives a stable behavior of the system in resource management, and shows better load balancing.

References

[1] http://ce.et.tudelft.nl/grappa/, GRAPPA Project, last visited March-2011.

[2] M. Dorigo and C. Blum, “Ant colony optimization theory: A survey,” *Theoretical Computer Science*, vol. 344, pp. 243–278, 2005.

[3] T. Abdullah, V. Sokolov, B. Pourebrahimi, and K. Bertels, “Self-organizing dynamic ad hoc grids,” in *Proceedings of the 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, Venice, October 2008, pp. 202–207.

[4] T. Abdullah, K. Bertels, and L. O. Alima, “Ant colony inspired microeconomic based resource management in ad hoc grids,” in *Proceedings of the 4th International Conference on Grid and Pervasive Computing*, Geneva, May 2009, pp. 189–198.

[5] M. A. Jaeger, H. Parzyjegla, G. Muhl, and K. Herrmann, “Self-organizing broker topologies for publish/subscribe systems,” in *Proceedings of the ACM symposium on Applied computing (SAC’07)*, 2007, pp. 543–550.

[6] G. Ritchie and J. Levine, “A hybrid ant algorithm for scheduling independent jobs in heterogeneous computing environments,” in *Proceedings of the 23rd workshop of the UK planning and scheduling special interest group*, Cork, Ireland, 2004, pp. 1–7.

[7] Y. Deng, F. Weng, and A. Ciura, “Ant colony optimization inspired resource discovery in P2P Grid systems,” *The Journal of Supercomputing*, vol. 49, no. 1, pp. 4–21, 2008.

[8] S. Fidanova and M. Durchova, “Ant algorithm for grid scheduling problem,” in *Proceedings of the 5th International Conference on Large-Scale Scientific Computations (LSCC ’05)*, 2006, pp. 405–412.

[9] Z. Zeng and B. Veeravalli, “Design and performance evaluation of queue-and-rate-adjustment dynamic load balancing policies for distributed networks,” *IEEE Transactions on Computer*, vol. 55, no. 11, pp. 1410–1422, 2006.

[10] A. Andrzejak, S. Graupner, V. Kotov, and H. Trinks, “Algorithms for self-organization and adaptive service placement in dynamic distributed systems,” HP laboratories Palo Alto, Tech. Rep. HPL-2002-259, 2002.

[11] A. Montresor, H. Meling, and O. Babaoglu, “Messor: Load-balancing through a swarm of autonomous agents,” University of Bologna, Tech. Rep., May 2002.

[12] Özalp Babaoglu, H. Meling, and A. Montresor, “Anthill: A framework for the development of agent-based peer-to-peer systems,” in *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems (ICDCS ’02)*, 2002, pp. 15–22.

[13] https://www.planet-lab.org/, PlanetLab Online, last visited March-2011.

[14] T. Abdullah, L. O. Alima, V. Sokolov, D. Calomme, and K. Bertels, “Hybrid resource discovery mechanism in ad hoc grid using strucutred overlay,” in *Proceedings of the 22nd International Conference on Architecture of Computing Systems*, March 2009, pp. 108–119.

[15] A. Rowstron and P. Druschel, “Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems,” in *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware ’01)*, vol. 2218, 2001, pp. 329–350.

 ISBN: 978-1-61208-134-2