Comparison of Nash Bargaining and Myopic Equilibrium for Resources Allocation in Cloud Computing

Giovanni Perin, Gianluca Fighera, and Leonardo Badia
Department of Information Engineering, University of Padova
Via Gradenigo, 6/b, 35131 Padova, Italy
Email: {giovanni.perin.2,gianluca.fighera}@studenti.unipd.it, badia@dei.unipd.it

***Abstract*—Distributed (cloud, cluster, grid) computing systems are becoming popular due to the huge amount of data available nowadays and the complexity of the computations required to handle them. An efficient allocation of computational resource is key to guarantee service quality in terms of execution time and cost. However, the inherent distributed character of these scenarios prevents them from adopting centralized allocation strategies and suggests that approaches inspired or related to game theory can be used instead. However, most solutions available in the literature propose simple techniques based on static allocation scenarios subsequently finding their outcome as a plain Nash equilibrium, which seems to leave some room for improvement. In this paper, we address this issue by considering instead a Nash bargaining solution obtaining a Pareto optimal solution of the allocation problem. We compare the results of this approach with those of a "myopic" strategy that pursues a Nash equilibrium, and we determine that, while both allocation strategies fully utilize the entire system capacity, a Nash bargaining achieves significantly better performance in terms of time spent by the users in the system. This gives evidence for a high Price of Anarchy of the myopic allocation and points out the need for a better allocation policy that makes a more efficient use of the available resources.**

***Index Terms*—Cloud computing, Game theory, Nash bargaining, Pareto efficiency, Integer Programming**

I. Introduction

Cloud computing makes computing resources, especially computational power and storage capability, available online on demand. Internet users are able to exploit this paradigm without the need for their direct management [1]. The physical devices providing such a service often form networks spread around the world. A cloud computing system is related to but also different from cluster computing, where instead computers are connected in a Local Area Network (LAN), and grid computing, which is a decentralized network of resources owned by multiple parties.

In any event, cloud computing and all similar distributed computing systems require provisioning of services of more and more advanced importance and also the underlying infrastructures are becoming increasingly complex. It would be nonsensical to resort to a centralized approach to manage a distributed computation system, and for this reason the idea of using a game-theoretic approach is well justified [2], [3].

Game theory is a branch of applied mathematics that was originally developed in the past century to economic scenarios but around the 2000s has become more and more popular for wireless and communication networks as well as distributed computing systems [4], [5]. It is particularly suitable for describing system interactions where individual players are guided by rationality and selfishness criteria; thus, it is even more suitable to be used by computing platforms whose capability of making correct decisions in the field of abstract reasoning is becoming really powerful.

For this reason, in the scientific literature we can find many examples of resources allocation algorithms for cloud computing that use a game-theoretic approach [2], [6]–[8]. However, a common trend of the majority of these papers is that they simply resort to the immediate analysis of game theory, i.e., to find an allocation strategy reaching a Nash equilibrium. On the one hand, this is indeed a correct choice, since a Nash equilibrium corresponds to a situation in which no user has any incentive to deviate from its strategy, and this then qualifies as a self-sustainable allocation in a distributed environment. Nevertheless, Nash equilibria are often sub-optimal in a global welfare sense. Game theory in reality proposes many advanced directions to solve this issue. It seems that, being its application of distributed computing platform in its infancy, it just scratched the surface without obtaining ultimately efficient proposals.

In this paper, we address the issue to quantify if game theory approaches exhibit room for improvement in resource allocation problems for cloud computing, and if so, how large. To this end, similar to many aforementioned references, we consider a game theoretic formulation where multiple tasks to be executed over a cloud computing platform are players in a game, so that they act in a fully uncoordinated and distributed fashion. Their strategies in the game are the choice of where, among the available computing servers, these tasks can be executed. We extend the scenario of [2] by adding the possibility of subtasks migrations between servers if the system will benefit from them. Instead of resorting to traditional Nash equlibrium solutions, we use a bargaining approach [9]. In particular, we discuss Nash barganing, which, despite being also proposed by John Nash, is actually an axiomatic approach and does not involve the Nash equilibrium as the solution concept, but rather can be seen as a physically achievable Pareto efficient allocation.

Our contribution gives the following advancements over the existing literature. First, we give a general characterization of the player's payoff (actually their cost function) in the case of tasks that can be divided in parallel subtasks, which allows a representation transcending the static allocation and considering also scenarios where servers are able to communicate. Then, we solve the allocation problem through the Nash bargaining solution and compare the allocation found with a solution resulting from a trivial and myopic Nash Equilibrium. Such a comparison determines a lower bound on the Price of Anarchy of the system, that is, the loss in efficiency due to players being just driven by selfish objective. We perform our evaluation also simulating the quality of service of the resulting allocation in a distributed computing scenario with typical parameters of a working environment.

From our simple comparison, it turns out that, even though the myopic allocation strategy achieves full resource utilization (which is relatively obvious, since unused resources would immediately attract distributed usage by some players), its adequacy is rather questionable as the overall efficiency is around 50% of the Nash bargaining solution. This highlights that game theoretic proposals for resource allocation in cloud computing are still in need of some strategical advancements to improve their performance.

The rest of the paper is organized as follows: in Section II we present in more depth the game theoretic approaches used by the already cited works, as well as other solutions found in other recent works. In Section III we describe in mathematical terms how we set the cost function of the system and in Section IV we give a game-theoretic interpretation of the two solutions that we present in the paper. In Section V we define the settings of our simulation and show the results obtained, with a special consideration to the Price of Anarchy. Finally, in Section VI we resume the important findings of our paper and suggest possible future expansions.

II. Related Work

Various solutions have been proposed in the literature to tackle the relevant problem of resource allocation in cloud computing services. Here, we just focus on those inspired by game theory, which is motivated by the desire for a rigorously founded and scalable distributed allocation approach.

In particular, one inspirational paper for the present analysis is [2], which focuses on a scenario similar to ours, where tasks can be divided over multiple subtasks and allocated to different servers. The problem is subsequently framed as a game and, following the concepts of (i) single and (ii) global participant efficiency loss of a reallocation, three different allocation algorithms are proposed. They all guarantee convergence towards a Nash Equilibrium of the allocation, but this is only used to justify the implementability of the proposal and the local optimality, without considering whether the allocation is also globally efficient.

In [6], instead, a minimax game approach with utility maximization is proposed to build an adaptive virtual machine allocation system. Another approach, employed by [7], [10] and [11], is the use of auctions, respectively in a cloud computing and grid computing scenario. In particular, the latter solves the task from the point of view of the service providers, with some constraints to meet user satisfaction, as we also do in this work. Likewise, the exploitation of the uncertainty principle of game theory has been proposed in [8], reaching a solution that leads to coalition-formation of the physical machines of the cloud. A number of optimal and suboptimal policies based on heuristics are proposed by [1], [12] and by [13], where four scheduling policies for virtual clusters with queuing systems and estimation of the resources required are presented.

In [14], a novel approach for resource allocation to make a reasonable application for task migration is developed: they build an imperfect information Stackelberg game using Hidden Markov Models (HMM) to predict the service provider's current bid, using historical data. Their model is shown to simultaneously increase the profits of both service providers and infrastructure suppliers.

In this work, we approach the problem of resource allocation through Nash bargaining. Such a solution concept has been employed in many other scenarios of allocation problems with good results, see [15] for more details. It falls within the number of axiomatic solution approaches to the allocation of constrained resources, as proposed by Nash in his seminal paper [16] (notably, there are other similar approaches most importantly the one by Kalai and Smorodinsky in 1975 [17]). Indeed, other papers have studied Nash bargaining solutions to cloud allocations, for example the allocation of multimedia cloud tasks subject to some constraints is proposed in [9]. A similar approach is also used for example in [18] to allocate power resources in a wireless powered relay system. Instead, we apply Nash bargaining to the assignment of virtual machines to physical machines, based on realistic constraints that can be considered typical of cloud computing.

Our objective is also to prove whether allocations based on a myopic strategy (which is basically the common element of all previously discussed papers) can be really considered efficient or not. As we will see, although Nash equilibrium solutions always utilize the entire cloud resources, they are not Pareto efficient and therefore pay what in game theory jargon is called the "Price of Anarchy." Certainly, the reduced system occupation time by our proposed Nash bargaining solution shows that there is room for improvement and possibly setting up more efficient allocation policies, still based on a distributed rationale, but exploiting more advanced game theoretic instruments.

III. Model

This section is dedicated to the mathematical modeling of the cost function of the system. We consider two factors that affect the QoS of our cloud computing physical system: execution time and cost. Let $\mathcal{S} = (S_1, \ldots, S_N)$ be the set of the N active tasks in the system, each one identifying a Virtual Machine and $\mathcal{R} = (R_1, \ldots, R_M)$ be the set of the M physical resources available. As done in [2], we assume that each task can be divided in k_i parallel subtasks, which for the sake of simplicity have all the same computational complexity. To account for a realistic parallel computation over multiple resources, we do not allow multiple subtasks of the same task to be allocated on the same resource. Therefore, a binary allocation matrix $A_{N\times M}$ can be defined, having a 1 in position i, j if a subtask of task S_i is assigned to the resource R_j. The matrix A is therefore the optimization variable of our problem, subject to the constraints

$$\sum_{j=1}^{M} a_{i,j} = k_i \qquad \forall i \in [N] \tag{1}$$

The two factors that affect the optimal decision for A can be represented by the matrices T for the time and E for the expense. The entries $t_{i,j}$ are the turnaround time it takes for resource R_j to complete $a_{i,j}$ subtask of the task S_i. Therefore, since subtasks are executed in parallel, the total execution time for task S_i is:

$$T_i = \max_{t_{i,j}\in t_i} t_{i,j} \qquad \forall i \in [N] \tag{2}$$

The entries $e_{i,j}$ are instead the expenses S_i pays for resource R_j to complete $a_{i,j}$ subtask. So, the price of task S_i is:

$$E_i = \sum_{j=1}^{M} e_{i,j} \qquad \forall i \in [N] \tag{3}$$

Slightly modifying what done in [2], we define a cost function which is the weighted average of the cost due to the execution time t and the cost due to the expense e of the resources. Moreover, we allow migrations between physical machines, and this affects the time cost by a certain amount $m_i(a, b)$ where a, b represent the physical source and destination of the migration. Thus, the total cost becomes:

$$\begin{aligned} J_i(t, e) = w_i \cdot \left(\max_{t_{i,j}\in t_i} t_{i,j} + m_i(a, b) \right) \\ + (1 - w_i) \cdot \sum_{j=1}^{M} e_{i,j} \end{aligned} \tag{4}$$

Both t and e depend on the optimization variable A. In our setting, a dynamic game is performed: given some constraints on the set of physical machines $\mathcal{R}$, tasks arrive with a certain rate at each time slot, having a bias towards time or price efficiency and some other characteristics. In particular, we need a price vector $\mathbf{c} = [c_1, \ldots, c_M]^T$, a transmission rate vector $\mathbf{r} = [r_1, \ldots, r_M]^T$ and a power vector $\mathbf{p} = [p_1, \ldots, p_M]^T$ with an entry for each physical resource. Moreover, we also need a computation vector $\mathbf{z} = [z_1, \ldots z_N]^T$ and a data vector $\mathbf{d} = [d_1, \ldots d_N]^T$ describing the amount of computation and memory required for each task. Given these parameters, we compute the matrix $\hat{T}_{N\times M}$, whose entries $\hat{t}_{i,j}$ represent the amount of time that resource R_j would employ to resolve subtask $a_{i,j}$ solely, as:

$$\hat{T} = \mathbf{z} \cdot \left(\mathbf{p}^{-1}\right)^T \tag{5}$$

From this matrix and from the other parameters, we can rewrite (4) in the form

$$\begin{aligned} J_i = w_i \cdot \left(\max\left(\hat{t}_{i,j} \sum_{i=1}^{N} a_{i,j} \right) + m_i(a, b) \right) \\ + (1 - w_i) \cdot \left(c_j + \frac{r_j}{R_m} \right) \sum_{j=1}^{M} (a_{i,j}\hat{t}_{i,j}) \end{aligned} \tag{6}$$

$$m_i(a, b) = \sum_{(R_a, R_b)\in\mathcal{R}} \left(\frac{d_i}{\min(r_a, r_b)} + \gamma \frac{d_i}{p_a} \right) \cdot \delta(a, b) \tag{7}$$

where $t_{i,j}$ and $e_{i,j}$ are translated according to [2], and J_i is then related to the values of the $a_{i,j}$s.

The formulation of (6) is explained as follows. The execution time is actually affected by the parallel executions of the other tasks at that time slot and the power of the CPUs is assumed to be divided equally between all the running tasks. The expense is proportional to both the price and the rate (normalized to the unit of money) of the employed resources and of course to the execution time. Since we add the possibility to move a subtask from a resource to another if it is convenient, we must take care of an additional term, $m_i(a, b)$, as per (7). Here d_i is the amount of data that is to be transmitted at a rate being the minimum of the two communicating servers. The second term is again proportional to the size of data and inversely proportional to the power of the source. This is the time needed to save the data that were modified by the executing program and γ is just a correction term. Finally, $\delta(a, b)$ is the indicator function which has value 1 if between (a, b) the transmission is active, 0 otherwise. When the optimal allocation at time slot t causes one or more migrations, we update the matrix $\hat{T}$, adding to the execution time left the fraction of time spent in the transmission and the backup of data. Also, at each time slot, we update $\hat{T}$ removing the fraction of $\mathbf{z}$ that was completed during that period.

IV. Game-Theoretic Approaches

In our work, we compare two different game-theory-based solutions to the problem of cloud computing resource allocation. First, we introduce the game theory notation applied to our problem. Given a set of players, in our case tasks, $\mathcal{S} = (S_1, \ldots, S_N)$, the set of their possible strategies is $\mathcal{A} = (a_1, \ldots, a_N)$, where a_i denote the allocation chosen by task S_i, which is row i of the binary matrix A. The choice of the allocation a_i for the task S_i will give to the task a payoff computed through a utility function

$$u_i : \mathcal{A} \longmapsto \mathbb{R} \tag{8}$$

which depends on the chosen allocation a_i but also on the strategies played by all other tasks, which, for brevity, are usually denoted a_{-i}. Therefore, the payoff for S_i will be the real number $u_i(a_i, a_{-i})$.

A. Nash Bargaining Solution

The first approach we employ was developed by John Nash in 1950, who proposed an axiomatic solution to the bargaining problem. As such, there is no real game in the Nash bargaining, and the solution is guaranteed to belong to the frontier of Pareto efficient allocations under certain assumptions described in the seminal paper [16]. Importantly, despite the similar names, the Nash bargaining solution does not represent a Nash Equilibrium of the game. In our scenario, the Nash bargaining approach is relevant because the service provider itself can force the system to guarantee the best resource allocation scheme leveraging users' preferences. This prevents users from being egoistic, ending up in sub-optimal solutions.

The Nash bargaining solution (NBS) for N players is defined as the solution of the problem:

$$\max \prod_{i=1}^{N} G_i(a_i, a_{-i}) \tag{9}$$

$$G_i(a_i, a_{-i}) = |\, u_i(a_i, a_{-i}) - u_i(d)\, | \tag{10}$$

where d is the disagreement point. If we define the utility function $u_i(\cdot)$ for each user as:

$$u_i(a_i, a_{-i}) = u_i(d) + \exp\left(J_i(a_i, a_{-i})^{-1}\right) \tag{11}$$

with $J_i(a_i, a_{-i})$ being the cost function described in Section III, the problem of (9) is reduced to:

$$\max \prod_{i=1}^{N} G_i(a_i, a_{-i}) = \max \prod_{i=1}^{N} \exp\left(J_i(a_i, a_{-i})^{-1}\right) \tag{12}$$

$$= \max \exp\left(\sum_{i=1}^{N} J_i(a_i, a_{-i})^{-1}\right) \tag{13}$$

$$= \min \sum_{i=1}^{N} J_i(a_i, a_{-i}) \tag{14}$$

Cost function $G_i(a_i, a_{-i})$ is chosen as the exponential function, which is common in the literature [9] for the sake of simpler computation, even though in a QoS problem the payoff has more probably a sigmoid shape, with some kind of saturation. In the following numerical results, we will stick to this commonly made choice. Any other choice would just make the numerical derivation of the maxima more complex, without much additional insight for the problem at hand, and for our discussion here it will just lead to minor numerical approximations with the exponential case.

It is also worth noting that, while the aforementioned formulation starts from requiring in principle a centralized arbitrator, the final derivation can still be efficiently approximated by distributed implementations since we end up in the minimization of the sum of the cost of each user, which can be seen as the total cost of the system.

B. Myopic Best Response Solution

We compare the solution described in Section IV-A with the one in which each task entering the system is allowed to choose its favourite allocation for itself. Therefore, each player computes privately

$$BR_i\left(a_i, a_{-i}\right) = \arg\max\, u_i(a_i, a_{-i}) \tag{15}$$

to find their best response to the strategies of the other players and play accordingly. In our setting, there is no simultaneous game: a task entering the system is supposed to have full knowledge about the allocation of the tasks already allocated. However, at the same time, no knowledge is given on the future arriving tasks. The latter assumption is a strong one, since it may be assumed that tasks at least know something about the statistics of arrivals in the system. Since the setting is unchanged, the best response can be computed as:

$$BR_i\left(a_i, a_{-i}\right) = \arg\min\, J_i(a_i, a_{-i}) \tag{16}$$

Due to the assumptions we made, this solution leads to a Nash Equilibrium, because each player is playing his best response to other players in the game and no prior is given about the future. Therefore, players cannot have regrets if their strategy is no more a best response after some other player played. Because of this and also since players are not allowed to change their play, we call this strategy Myopic Best Response.

In both the versions of Section IV-A and Section IV-B, the solutions are Binary Integer Programming (BIP) optimization problems. This is because the variable A is restricted to have values in $\{0, 1\}$.

V. Simulation and Results

In this section, the parameters of the implemented model are shown, together with the results of the simulation.

In our experiment, we considered a fixed number of resources $M = 5$. The computational power of each resource is expressed in MIPS (million instructions per second), while the resources cost is expressed in a generic money unit. This choice is due to the fact that, in our experiment, we were not interested in the exact cost in terms of money, but we just wanted to model that the more powerful is a resource, the higher it costs. The parameters are listed in Tab. I. The communication rate between each resource is chosen as $r = 0.1$ GB/s and it is equal for all resources for the sake of simplicity.

For what concerns the tasks, which are the players in our system, their arrival is modelled as a Poisson process with arrival rate fixed to $\lambda = 0.075$ min^{-1}. Each player is also characterized by a type w_i that is a random variable with uniform distribution in $[0, 1]$ and it indicates the weight that the player gives to the execution time over the expenses. A player with $w_i = 1$ will only seek to minimize the time spent in the system, regardless the cost, whereas a player with $w_i = 0$ will be only interested in minimizing the expense, no matter how long it spends in the system. In general, we say that the execution time t_i and the expense e_i paid by user i are weighted as $w_i t_i + (1 - w_i)e_i$, as per (4).

M (Nr. of Physical Machines)	5
Δt (Time slot)	10 [min]
N (Nr. of tasks in execution)	$\mathcal{P}(\lambda) \quad \lambda = 0.75$ [10 min]$^{-1}$
k_i (Nr. of subtasks per task)	$p(1) = p(5) = 0.1$ $p(2) = p(4) = 0.25 \quad p(3) = 0.3$
w_i (Weigth for time cost)	$\mathcal{U}([0, 1])$
z_i (Nr. of ops per subtask)	$\mathcal{Z}(3, 10) \times 10^{15}$ [MIPS · s]
d_i (Size of data)	$\mathcal{Z}(0.07, 10)$ [GB]
$\mathbf{c}$ (Cost of the resources)	$[10, 7, 3, 2, 1]^T$ [money / 10 min]
$\mathbf{p}$ (Power of the CPUs)	$[15, 9, 6, 3, 2]^T \times 10^{12}$ [MIPS]
$\mathbf{r}$ (Transmission rate)	$[0.1, 0.1, 0.1, 0.1, 0.1]^T$ [GB/s]
γ (Correction term for storage)	30 [GB]$^{-1}$

TABLE I: Resume of the values chosen for the simulation

Each task in the system is composed of an integer number of subtasks to be executed sampled from a truncated Gaussian distribution in [1,5] with average $\mu = 3$ and variance $\sigma^2 = 10.6$. Different subtasks of the same task have the same size in terms of number of operations to be executed, randomly picked from a truncated Gaussian distribution with $\mu = 3$ and $\sigma^2 = 10$, denoted as $\mathcal{Z}(3, 10)$. The size is expressed in a unit of measure corresponding to 10^{15} MIPS. Finally, subtasks of the same task are characterized by a specific size of data to be exchanged by resources in case of reallocations, again sampled from a truncated Gaussian distribution with $\mu = 0.07$ and $\sigma^2 = 10$ and expressed in GB. In our experiments, the arrivals occur during a 24h interval and then the system keeps executing subtasks until all tasks are terminated. This simulation is run 1000 times and results are averaged.

We perform a comparison between the NBS and the myopic strategy of just playing a best response (BR). Four evaluation metrics are considered: the number of tasks in the system, the number of active resources, the average cost of the system, and the average total time spent in the system by the tasks. Fig. 1 shows the number of tasks in the system as a function of time. Both NBS and BR approaches show a rapid increase in the initial part, where the system is loaded with tasks, and reach their steady-state values after about 10 hours. In this situation, the number of users in the system is more or less stable since the arrival rate equals the exiting rate. After the last time interval of the day, there are no more arrivals and the system empties with a rate that is almost equal for the two solutions considered. After $2-3$ hours the system has completed all subtasks and is empty. The steady-state value for the NBS approach is about 4 tasks, while those for BR approach is 8 tasks. This means that a system implementing the NBS policy in the management of incoming users is able to half its load factor with respect to a system implementing a BR approach, without losing anything both in term of disposal time at the end of the 24 hours and in term of time for system loading at the beginning. The vertical dashed green line at $x = 24$ indicates the limit time for arrivals. In Fig. 2 a comparison between the two approaches is shown, the measures being normalized with respect to the maximum one. The first considered measure is the average number of active resources, i.e., resources that are processing at least one subtask. The value for the NBS approach is slightly lower, indicating the presence of some time intervals in which the optimal allocation of subtasks allows some resources to be shut down. The second measure is the average cost from the system perspective, computed as the sum of the costs for the resources to be active and to exchange data between them. As it can be seen, there is neither waste nor gain of money of the NBS approach with respect to the BR solution. The third measurement is the average time spent in the system by a task, computed using Little's law. Indeed, since we know

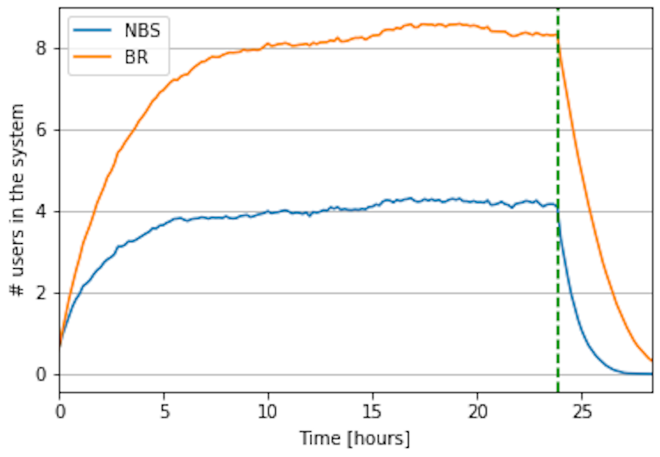

Fig. 1: Time-evolution of the system load.

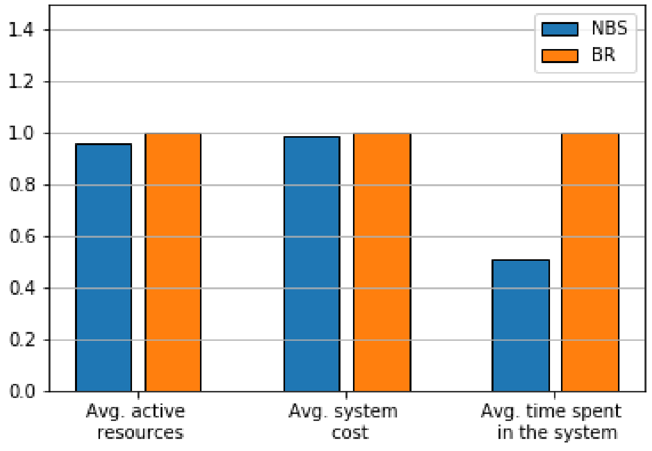

Fig. 2: Comparison between Nash bargaining solution and Best response approaches.

the average number of users in the system, denoted as $\bar{X}$, and the arrival rate λ, the average time $\bar{S}$ spent in the system is computed through

$$\bar{X} = \lambda \cdot \bar{S} \tag{17}$$

As expected from the fact that the steady-state number of users in the system in NBS is about half of BR, the time spent in the system implementing NBS policy is also halved. This difference in performance between the Pareto optimal solution of the Nash bargaining and the Nash Equilibrium of the myopic strategy can be seen as a lower bound to the Price of Anarchy of the system. In other words, if players are acting selfishly, i.e., only driven by their myopic best interest, they obtain a less efficient allocation, not in that some resources are underutilized (they are indeed fully exploited anyways) but rather by an increased congestion, which results in their permanence in the system for a time that is at least doubled with respect to the best Pareto efficient allocation. We remark that, while the specific numerical result of a halved permanence time in the system through the NBS can of course depend on the choice of parameters for the evaluation (which have been selected with in mind a realistic platform, anyways), the order of magnitude of this comparison confirms that there is a significant improvement margin from myopic best-response equilibria.

VI. Concluding Remarks

In this paper, a Nash Bargaining Solution has been proposed for the optimal allocation of subtasks among resources in a cloud computing system. The same system has been tested also with a different allocation policy in which every player entering the system played its best response. The two different approaches were tested in terms of number of players in the system, average resources utilization, average system cost and average time spent by tasks in the system. The most remarkable result is that our allocation policy is able to halve the average number of tasks in the system without paying anything in terms of cost and number of active resources, making this algorithm suitable for this type of situations.

Future work can be done by considering different arrival rates for tasks within 24 hours, since during the day it is more likely for the system to receive tasks to execute than during night. Moreover, we have decided to limit the number of subtasks that can be executed on a single resource, hence each task has a number of subtasks limited to 5. A queue can be considered for each resource, allowing for the execution only of an upper bounded number of subtasks by the same physical machine. For what concerns the computational power of the resources, we set to 2×10^{12} MIPS the power of the less capable resource: this corresponds to the power of a good laptop. Other choices can be done considering for example a network of machines with different physical specifications. A more precise analysis can be performed computing the price paid and the time spent in the system by each task individually to assess with how much accuracy the system meets specific quality of service requirements.

References

[1] M. A. Sharkh, A. Shami, and A. Ouda, "Optimal and suboptimal resource allocation techniques in cloud computing data centers," *Journal of Cloud Computing*, vol. 6, no. 1, p. 6, 2017.

[2] G. Wei, A. V. Vasilakos, Y. Zheng, and N. Xiong, "A game-theoretic method of fair resource allocation for cloud computing services," *The journal of supercomputing*, vol. 54, no. 2, pp. 252–269, 2010.

[3] A. V. Guglielmi, M. Levorato, and L. Badia, "A Bayesian game theoretic approach to task offloading in edge and cloud computing," in *Proc. IEEE Globecom*, 2018.

[4] Z. Han, D. Niyato, W. Saad, T. Ba?ar, and A. Hjø rungnes, *Game theory in wireless and communication networks: theory, models, and applications.* Cambridge university press, 2012.

[5] I. Abraham, L. Alvisi, and J. Y. Halpern, "Distributed computing meets game theory: combining insights from two fields," *ACM Sigact News*, vol. 42, no. 2, pp. 69–76, 2011.

[6] K. Srinivasa, S. Srinidhi, K. S. Kumar, V. Shenvi, U. S. Kaushik, and K. Mishra, "Game theoretic resource allocation in cloud computing," in *Proc. Int. Conf. Appl. Dig. Information and Web Technologies (ICADIWT)*, pp. 36–42, IEEE, 2014.

[7] A. Nezarat and G. Dastghaibifard, "Efficient Nash equilibrium resource allocation based on game theory mechanism in cloud computing by using auction," *PloS one*, vol. 10, no. 10, p. e0138424, 2015.

[8] P. S. Pillai and S. Rao, "Resource allocation in cloud computing using the uncertainty principle of game theory," *IEEE Systems Journal*, vol. 10, no. 2, pp. 637–648, 2016.

[9] M. M. Hassan and A. Alamri, "Virtual machine resource allocation for multimedia cloud: a Nash bargaining approach," *Procedia Computer Science*, vol. 34(Supplement C), pp. 571–576, 2014.

[10] S. Sequeira and P. Karthikeyan, "A survey on auction based resource allocation in cloud computing," *International Journal of Research in Computer Applications and Robotics*, vol. 1, no. 9, pp. 96–102, 2013.

[11] A. R. Gahrouei and M. Ghatee, "Auction-based approximate algorithm for grid system scheduling under resource provider strategies," *arXiv preprint arXiv:1803.04385*, 2018.

[12] M. B. Gawali and S. K. Shinde, "Task scheduling and resource allocation in cloud computing using a heuristic approach," *Journal of Cloud Computing*, vol. 7, no. 1, p. 4, 2018.

[13] T. J. Hacker and K. Mahadik, "Flexible resource allocation for reliable virtual cluster computing systems," in *Proc. Int. Conf. High Perf. Comput., Networking, Storage and Anal.*, p. 48, ACM, 2011.

[14] W. Wei, X. Fan, H. Song, X. Fan, and J. Yang, "Imperfect information dynamic stackelberg game based resource allocation using hidden markov for cloud computing," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 78–89, 2018.

[15] H. Park and M. van der Schaar, "Bargaining strategies for networked multimedia resource management," *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3496–3511, 2007.

[16] J. F. Nash Jr, "The bargaining problem," *Econometrica*, pp. 155–162, 1950.

[17] E. Kalai and M. Smorodinsky, "Other solutions to Nash's bargaining problem," *Econometrica*, pp. 513–518, 1975.

[18] Z. Zheng, L. Song, D. Niyato, and Z. Han, "Resource allocation in wireless powered relay networks: A bargaining game approach," *IEEE Trans. Veh. Tech.*, vol. 66, no. 7, pp. 6310–6323, 2017.