Effort-Based Incentives for Resource Sharing in Collaborative Volunteer Applications

Davide Vega, Roc Meseguer, Felix Freitag
Computer Architecture Department
Universitat Politècnica de Catalunya
Barcelona, Spain
{dvega, meseguer, felix}@ac.upc.edu

Sergio F. Ochoa
Department of Computer Science
Universidad de Chile
Santiago, Chile
sochoa@dcc.uchile.cl

***Abstract*— Collaborative and volunteer applications need to implement incentive mechanisms to regulate resource sharing and encourage network nodes to contribute for reaching a certain goal. Typically, these incentive mechanisms assign resources to network node requests, based on the total amount of resources contributed by the requesting participant. This approach assumes that participants contributing more should also get back more resources from the collaborative environment. This assumption turns the system unfair to those participants with scarce resources, because they have just few resources to share. This paper proposes the use of an incentive strategy based on the contribution percentage of each node; i.e. an effort-based approach. This proposal is evaluated and compared to contribution-based strategies. The obtained results show that the proposed effort-based approach not only benefits participants that have scarce resources, but also it is able to satisfy the requests of the powerful nodes.**

Keywords*— *Collaborative computing, resource sharing, effort-based incentives, volunteer participation.

I. INTRODUCTION

Advances in wireless communication technology and mobile computing have transformed the physical scenario in a heterogeneous ecosystem where several device types can cooperate to reach individual or group goals. For example, an application running in a cellular phone can ask for help to other mobile computing devices in the neighborhood, to disseminate information to pedestrians in a certain area; e.g. the today's menu of restaurants located in a food court, or the offers of the day in a shopping mall. This type of collaborative applications usually utilizes volunteer approaches for information processing and dissemination, which also involve CPU, memory and energy of the devices participating in this process.

In collaborative volunteer computing [1,2], the participants share part of their hardware resources (e.g. CPU and memory) to help other participants to perform certain tasks. These participants are connected through a network. Therefore the collaborative environment can be seen as a heterogeneous mesh, where each one represents a network node. When a node requires more resources than those that it has locally available, it must borrow the extra resources to other nodes. As nodes request resources to other nodes and also provide their own resources to others, a sharing ecosystem is created, in which the participants interchange time slots of their resources. In order to have a sustainable ecosystem, participants must return the borrowed resources once they not needed anymore. A free and unrestricted access to the shared resources without the need to contribute with the ecosystem, will carry the sharing space to the point in which the resources are exhausted; therefore the ecosystem becomes useless. This problem, known as the "tragedy of the commons" [3], is caused by nodes doing free-riding [4].

Incentives mechanisms are typically used to regulate global and individual benefits, by encouraging nodes to collaborate and granting them a "fair" return for their contributions. Traditional methods to grade peers collaboration (i.e., to calculate their expected return) are based on absolute hardware contribution measurements. Some research works argues that nodes contributing more resources should receive in return a better service than those that contribute less [5, 6]. Such a contribution-based mechanism is simple and efficient in most cases. However, it has been demonstrated that when we apply contribution-based mechanisms in heterogeneous scenarios, the nodes tend to share and cooperate only between equals – in terms of resources [7]. Hence, nodes with scarce resources suffer discrimination because they have not enough resources to contribute.

This paper proposes and evaluates an effort-based mechanism for resource sharing in collaborative applications. This mechanism is based on the proposal of [8] that considers the nodes' effort, instead of taking into account only the absolute value of the nodes contributions. In order to evaluate the potential benefits for applications working in collaborative scenarios, we have simulated several test cases in which nodes share their resources (particularly, CPU slots) and make decisions about when and how collaborate, using an Iterative Prisoners' Dilemma game [9].

The obtained results show that when nodes use an effort-based strategy to make a collaboration decision, the process tends to be fair for every type of node. We have also observed that the percentage of tasks that nodes with scarce resources can perform in the system increases without harming the powerful nodes performance.

Next section presents a discussion on effort-based and contribution-based incentive strategies. It particularly shows the impact that the network scale and topology have on the collaboration process. Section III describes the framework used in the simulations conducted to try clarifying this issue. Section IV presents and discusses the obtained results. Finally, Section V presents the conclusions and the future work.

II. EXPLORATION OF EFFORT-BASED INCENTIVES

Rahman et al. [8] applied effort-based incentives to share resources in the BitTorrent protocol and also in a generic credit-based sharing ratio enforcement scheme. For the scenario, a BitTorrent-like system was simulated. The upload bandwidth was the metric considered to assess the peers reciprocal actions. The system simulated two types of peers: fast and slow peers, while the latter had half of the upload capacity of the first ones. The proportion of these node types in the BitTorrent swarm was varied in each experiment. The simulation results showed a greater utilization of the available resources when the effort-based incentives was used. Moreover, that strategy also contributed to reduce the download times of slow peers. It was observed that sometimes the fast peers had to sacrifice some of their performance.

Similarly, the BitTorrent simulation using a credit-based sharing ratio enforcement scheme was also done with two types of peers, one with high upload and download capacity, and the second one with low exchange capacity. The results showed that the download performance of the peers increased.

While the results of both incentive strategies of [8] are encouraging, the comparison between them leaves open questions regarding the impact of the network scale and of the topology on the collaboration process. We extend such a study by addressing the previously mentioned open issues. In our study the simulations involve more than 100 nodes and different network topologies. Moreover, we extend the binary classification of peer capacities towards a continuum of capacities that allow the nodes to make requests for different amounts of resources.

In an initial exploration of the effort-based approach, we performed four simulations on a peer-to-peer network with a small-world topology, and 125 nodes playing the Iterative Prisoner's Dilemma game. In the first three experiments, the nodes apply a tit-for-tat strategy to decide when to cooperate and with whom. In the fourth experiment the nodes apply an effort-based strategy to decide their next cooperation action. Figure 1 shows the cooperation coefficient – round by round - for all strategies. The results show that in the contribution-based tit-for-tat strategy without forgiveness, i.e. replying strictly reciprocal to the previous action of the other node, all nodes start to defect requests of other nodes after the first 50 transitional rounds.

When a forgiveness variant is applied, the nodes are able to recover from this situation and cooperate between 17% and 11% of the times. However, in the effort-based strategy, the nodes receive more contributions from their neighbors and thus they achieve a higher satisfaction.

Provided the differences in the nodes cooperation coefficient when using both approaches, we analyze in the study described in this paper in detail these differences in terms of global cooperation, resources distribution and locality satisfaction for both strategies (i.e. contribution-based and effort-based). For understanding these issues, several simulations were conducted using different scenarios, represented through multiple overlay topologies and resources distributions.

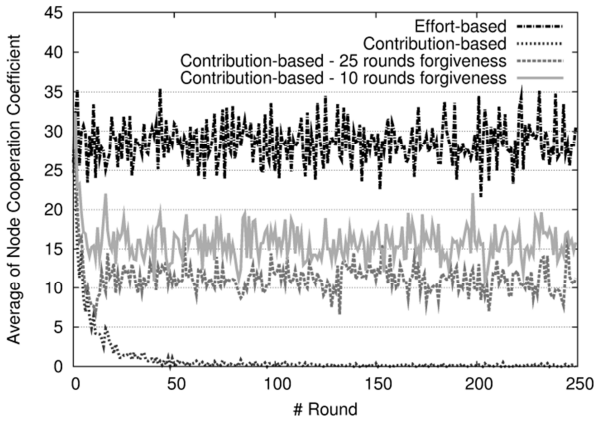

Figure 1. Nodes cooperation results, round by round

III. EXPERIMENTAL FRAMEWORK

All experiments performed in this study were done with a cooperative game simulator [10]. This tool allows configuring a large number of scenarios, reproducing the experiments and also isolating the variables to be studied. The simulator is also able to extract statistical results from the nodes behavior, when these nodes play a particular collaborative game.

We consider a stationary network scenario in order to focus our analysis on the impact produced by the network scale and topology. Links between nodes were considered homogeneous and stable; i.e. they do not change during the experiment. However, the nodes are heterogeneous (with a different number of CPU slots). These nodes play an Iterative Prisoner's Dilemma game with their neighboring nodes, upon the predefined network topology.

Every experiment in this study involved simulations performed over a discrete scenario with 250 rounds. The results of a previous work show that this number of rounds is enough to extract significant statistical conclusions after discarding the first fifty transitional rounds [10].

The first step in the simulation process is to load a certain network topology type, which is created by a topology generator. The network edges are undirected and non-weighted. The values for the variables representing the environmental conditions (i.e. CPU slots distribution, simulation time, nodes strategy) were predefined. Then, the simulations were executed with only one independent variable, assuring thus that the results reflect the impact of just that variable. In the next sections, we describe the network topologies, the process to model resources and nodes behavior, and the resources sharing strategies used in the simulations.

A. *Network topologies*

Our evaluation has been mostly conducted in networks of *small-world* topology, since social networked scenarios exhibit properties that are usually present in this topology [11, 12]. The small-world effect has also been considered for collaborative and peer-to-peer networking applications. However, we have also used a *torus* topology to assess if there is any dependence of the obtained simulation results on the network topology.

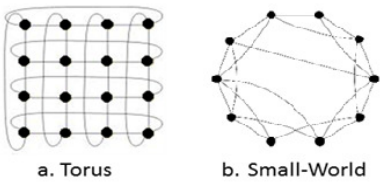

Figure 2. Network topologies used in the study

Torus: This topology involves an N-dimensional torus, defined as the Cartesian product of N rings. A symmetric torus is a regular topology that connects the head node with the tail node in each row and column. Thus this topology eliminates the edge effect. Consequently, the torus structures preserve all the topological properties; therefore the properties of one node are representative for the rest of the network nodes.

Small-world: Two characteristics distinguish the small-world topology from other kinds of networks: (1) there is a small average path length between each pair of nodes and (2) the topology has a high clustering coefficient that is independent of the network size.

B. Modeling resources and nodes behavior

Each network node represents a computing device (e.g. a handheld, desktop computer or router) with a fixed amount of resources available to be used by itself and other nodes. The nodes can share their resources with the neighboring nodes according to the network topology.

In our simulations, the resource type considered was just CPU slots. However any other type of resource can be shared in a real situation, e.g. the GPS, videocamera, memory and the network card. The CPU slots were randomly distributed among the nodes during each simulation initialization, using a discretized normal distribution with a lower bound of 1 CPU slot per node.

In each round, if a node completed its own tasks, it will generate a new task with 50% of probability. At this point, it does not matter if the node is running on its CPU slots other nodes' tasks. When a node decides to perform a task, it calculates the task cost in terms of needed CPU slots and time (rounds to perform it). The CPU requirements are randomly determined using a discrete normal distribution, with a mean of 6 slots and a variance of 2. The minimum number of CPU slots asked in a request is 1. The number of rounds needed to perform the task is randomly determined using a discrete uniform probability distribution, with a minimum value of 1 and a maximum of 3.

Whenever a node needs to perform a task, and after it has calculated the task cost, it first tries to use its own free CPU slots to complete it. If the node does not have enough free CPU slots, then it asks to its neighboring nodes for the CPU slots needed to accomplish the task. The decision on whether a node will cooperate or defect is modeled by the Iterative Prisoner's Dilemma.

Once the requesting node has achieved enough CPU slots to perform the task, these CPU slots will be blocked for the duration of the task. Any excess in the lent CPU slots will be released. If not enough slots are obtained for the task, all obtained slots are released; therefore they can be used to support other requesting nodes.

C. Sharing strategy

Each node plays a cooperative game with all of its neighboring nodes whenever the node needs more CPU slots. The Iterative Prisoner's Dilemma is the collaboration model we use to describe the relationships between the nodes. In this game, two players choose between cooperation or defection. The payoffs matrix of the two actions is shown in Table I. The relations between the different possible payoffs follow the rule $b > c > \varepsilon \rightarrow 0$, which poses the dilemma.

TABLE I. PAYOFF MATRIX OF PRISONER'S GAME

Player decision	Co-player Cooperate	Co-player Defection
Cooperate	b - c	c
Defection	b	$\varepsilon \rightarrow 0$

The decision of choosing one or another action can be based on the trust in others and their reciprocity. The consequences of the participants choices have a different impact on the global and local node community, and also on the own benefit of the nodes. In our study we evaluate these impacts when the decision is based on two approaches: the contribution-based and the effort-based strategy.

Contribution-based Incentive: This approach considers the absolute contribution of the requesting node, when the requested nodes have to make the decision about whether to cooperate or defect. A well-known strategy to maximize the payoff when using this approach is the tit-for-tat strategy. Using that strategy, a certain node A responds to the request of a node B by making the same decision (i.e. cooperation or defection) made by B during the last request from A. In the case that A and B have not met before, and B has free CPU slots, we consider in our simulation that node B is a cooperator with 50% probability.

Contribution-based Incentive with Forgiveness: It is based on a variant of tit-for-tat named naive pacemaker incentive, in which participants sometimes make peace by co-operating in lieu of defecting. The objective of this strategy is to avoid being highly harmed by single and/or random defection of other co-player. When node B has defected node A during a consecutive number of rounds, node A considers him as an unknown node; therefore node A will cooperate with B with 50% probability. This strategy has been used in the preliminary study of section II.

Effort-based Incentive: It is based on the nodes relative contribution or effort, defining the contribution as a fraction of the resources, instead of the absolute value of the contribution itself. In order to evaluate the cooperation process using this incentive, we have simulated the effort-based strategy as follows:

For a given round, a certain node A responds to the request of a node B assigning to each requested slot a probability of cooperation P. The probability P is calculated as the ratio between the amounts of CPU slots that B shared with A in the last request, and the total amount of CPU slots that B owns. Like in the contribution-based scenario, if A and B have not

met before, and B has free CPU slots, then A evaluates each slot with a 50% cooperation probability.

The idea behind effort-based incentive is to create a mechanism which does not harm nodes with few CPU slots in a contributory ecosystem. At the same time, the effort-based approach also aims to maintain a reciprocity policy similar to tit-for-tat.

The models of contribution-based and effort-based incentive strategies were implemented in our simulator. Then they were used to understand the advantages and drawbacks on cooperation, when nodes use both approaches to play the Iterative Prisoner's Dilemma game. Modifications to the effort-based approach can be made to take into account additional factors (e.g. resources optimization).

IV. Results and Discussion

The simulation results allow us to understand the impact of both incentives on a collaborative scenario, considering static network topologies and heterogeneous resources distribution. The next subsections describe the settings of the simulated experiments and discuss the obtained results.

Four different scenarios were used in the simulations. Table II shows their network topology and resource distribution. In addition the global results are also shown. The results are discussed in the following sections.

The scenarios are divided into two categories: *resource scarcity* and *resource abundance*. In scarcity scenarios, the amount of resources available is, in average, twice the amount requested per node and round. In resource abundance scenarios, the available resources are four times the requested amount of resources. We assume that these resources will never be 100% used, because regarding own tasks, each node can perform at most one during a maximum of three rounds.

A. Metrics used in the simulations

The simulation results were assessed using the following three metrics: *cooperation coefficient*, *percentage of tasks satisfied* and *node success percentage*. Next we briefly explain these metrics.

Node Cooperation Coefficient: It is calculated as the ratio between the amount of requested CPU slots and the number of positive answers – in term of CPU slots – obtained during the simulation, independently if the lent slots were used or not. Therefore, it is a measure of the nodes' willingness to collaborate with each other.

Tasks Fully Satisfied: It is calculated as the ratio between the number of tasks that all nodes want to perform during an experiment, and the number of tasks effectively completed. Therefore, it is a measure of how many tasks get the amount of resources needed for their completion.

Node Success Percentage (NSP): It is calculated as the ratio between the number of tasks that a given node wants to perform during an experiment, and the number of completed tasks. Therefore, it is a measure of the nodes' satisfaction. Notice that unlike the tasks fully satisfied metric, the NSP is calculated over each node individually.

In Table II, the cooperation coefficient shows high variations in the minimum and average values, considering the contribution-based (tit-for-tat) and effort-based incentives. The noticed variation is more important in resource scarcity scenarios. In our simulation, the reduction of a 50% in the available resources represented a reduction of positive responses over 8%. According to that observation, the nodes with effort-based incentives are more interested in cooperation than nodes with contribution-based incentives. It is important to notice that the cooperation coefficient of the nodes does not indicate that the whole amount of resources needed to complete a task will be acquired.

The results also show that the value of the tasks fully satisfied metric is high when using tit-for-tat strategies. The result is coherent with simulation games and strategies where nodes rather than requesting evenly to all of their partners, make decisions one by one.

In order to have a better understanding of these phenomena, we compute the *Node Success Percentage (NSP)* that is a direct measure of the nodes satisfaction. Table II shows the average, minimum and maximum NSP achieved by each of the 125 nodes in the four selected scenarios using each incentive. The results reflect the fact that when nodes use the contribution-based strategy, the average satisfaction – measured as the average Node Success Percentage – is a 6% over the satisfaction when they use a tit-for-tat strategy; no matter how many resources the system has or which are the topological properties of the network. These results are more consistent with the cooperation coefficient, and reveal that the task success percentage is not a consequence of a high cooperation among the nodes.

B. Impacts on resources utilization

The results of the first simulations show a trend difference between the percentage of *tasks satisfied* (see Table II) and both the *cooperation coefficient* and the *NSP* using a tit-for-tat strategy. Understanding the reason for this difference is key for understanding the effect of the contribution-based and effort-based incentives. A way to answer this question would be to study the impact of different strategies on resources utilization.

Figure 3 shows histogram with the average distribution of CPU slots in the four scenarios executed on a small-world topology. Four metrics are utilized: used, shared, free and requested CPU slots.

The amount of *used* CPU slots represents the number of slots that nodes assign to their own tasks, the *shared* slots indicate the slots devoted by a node to tasks of other nodes, and the number of *free* slots represents the slots that are not used. Finally, the *requested* slots show the amount of CPU slots that nodes intend to obtain from other nodes.

In *resource scarcity* scenarios, the experiment shows that when nodes use an effort-based instead of a tit-for-tat strategy, they dedicate more CPU slots to tasks belonging to other nodes at the expense of their own tasks. This solidarity helps other nodes to get all the required resources for their tasks, turning the competition on cooperation, and thus increasing the *NSP*. In *resource abundance* scenarios, this cooperation improvement is smaller, and it leads to a high reduction of

resources that nodes use for themselves. Therefore, it is not surprising to observe that the percentage of *requests per node and round* is about 7.5% lower. It is a clear evidence that nodes are self-provisioning themselves and do not need others' cooperation.

Finally, notice that in all cases there is a high amount of CPU slots underused by nodes which could contribute to increase the value of the metric *tasks fully satisfied*. That happens because both strategies in their decisions consider only the previous relationship between the nodes into account. Any other factor, such as resource usage or energy consumption is not considered at this time. However, it is important to point out that in our experiments one benefit of the contribution-based strategies compared to effort-based strategies is that they achieved a better resource usage (see non-free/free CPU slots distribution ratio in Table II).

C. Impact of node properties and network location

This simulation shows the results of evaluating the nodes behaviors from the locality point of view. The high range between the maximum and minimum *cooperation coefficient* and *NSP* in Table II, in addition to the high number of nodes' CPU dedication in Figure 3, suggest that there exist node conditions or topological properties that help some nodes to achieve a better satisfaction than others.

TABLE II. OBTAINED RESULTS

Strategy experiment	Topology	CPU Dist.		Cooperation Coefficient			Node Success Percentage			% Tasks satisfied	% Overall requests	CPU Usage dist.	
		Mean	*Var*	*Min*	*Max*	*Avg*	*Min*	*Max*	*Avg*			*Non free / Free*	*Own / Shared*
Tit-for-tat	Torus	6	4	0.00%	35.50%	1.59%	0.00%	47.59%	7.47%	87.15%	40.45%	0.518	16.793
Effort-based	Torus	6	4	21.76%	38.73%	30.34%	1.01%	71.05%	21.82%	73.23%	48.26%	0.267	0.767
Tit-for-tat	Torus	12	4	9.73%	41.40%	17.62%	0.00%	100%	33.96%	57.75%	53.97%	0.563	3.056
Effort-based	Torus	12	4	27.78%	46.74%	37.97%	24.62%	100%	46.02%	51.24%	61.57%	0.563	2.264
Tit-for-tat	SmallWorld	6	4	0.00%	35.49%	1.60%	0.00%	46.38%	7.04%	88.05%	41.37%	0.360	15.536
Pacemaker 25	SmallWorld	6	4	7.37%	29.33%	11.62%	0.00%	51.61%	15.97%	76.93%	45.96%	0.285	1.275
Pacemaker 15	SmallWorld	6	4	9.79%	26.32%	14.56%	0.63%	51.19%	17.32%	76.28%	46.56%	0.258	1.066
Effort-based	SmallWorld	6	4	21.58%	35.43%	28.55%	0.00%	86.57%	27.79%	68.30%	51.88%	0.167	0.646
Tit-for-tat	SmallWorld	12	4	2.79%	42.32%	9.57%	0.53%	100%	34.92%	57.26%	54.93%	0.509	3.680
Pacemaker 25	SmallWorld	12	4	12.92%	37.65%	19.27%	6.57%	100%	40.73%	53.13%	58.52%	0.540	2.757
Pacemaker 15	SmallWorld	12	4	13.37%	34.47%	18.66%	10.46%	100%	42.04%	54.29%	59.62%	0.518	1.975
Effort-based	SmallWorld	12	4	25.98%	43.59%	37.01%	24.29%	100%	49.92%	50.59%	64.22%	0.355	1.800

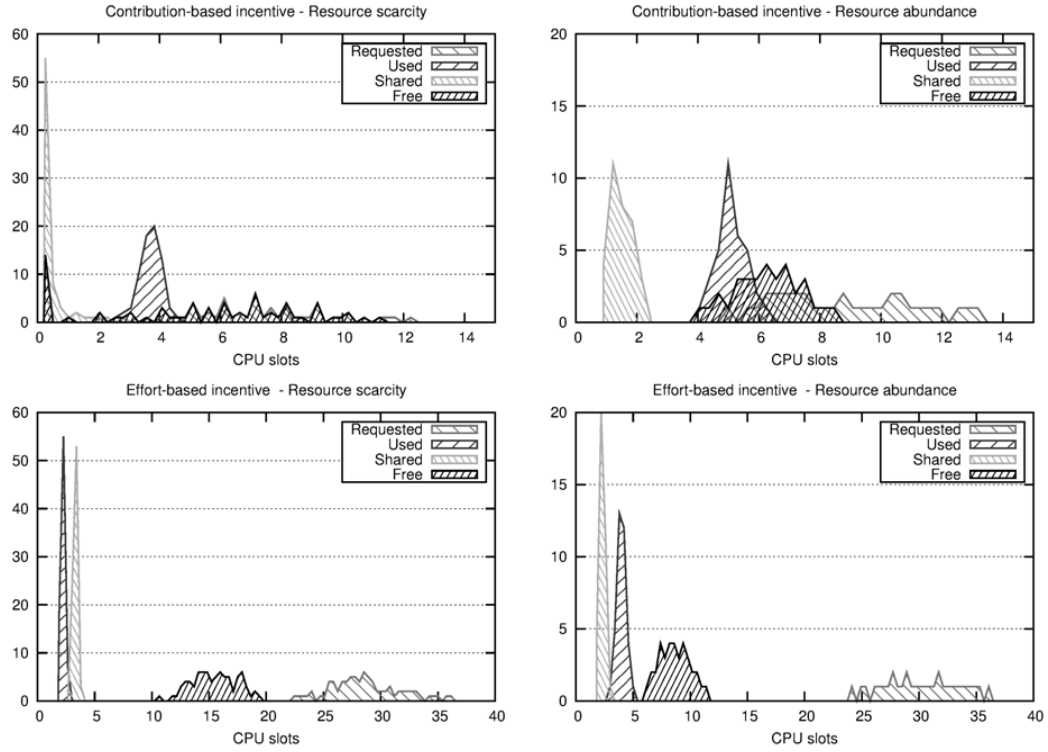

Figure 3. Histograms of average CPU slots requested, used, shared and free on different scenarios involving a small-world topology

Discarding the second reason – since the Torus topology shows the same behavior than small-world – we conducted a simulation that measures the individual nodes satisfaction in different scenarios. Figure 4 shows the NSP achieved by each node ordered according to their total amount of CPU slots. The experiment has been conducted using a small-world topology and 125 nodes in a *resource scarcity* scenario. In *resource abundance* the results show the same behavioral pattern. The results show also two different behaviors, one for nodes that have more than 8 CPU slots and another for nodes that have 8 or fewer CPU slots. In the first case it does not matter what strategy they play, because both strategies achieve the same NSP. However, when nodes have few slots, the contribution-based strategy is unfavorable, resulting in a NSP below 20%.

Contrarily, when using an effort-based strategy, the success percentage is between 3% and 70%. This shows that the effort-based strategy is fairer than contribution-based strategies, when they are used in heterogeneous resource sharing scenarios. The idea main is to avoid the discriminatory effect shown in Figure 1. Instead, measuring their *relative* cooperation willingness gives everyone the opportunity to share the same maximum value, up to 100%.

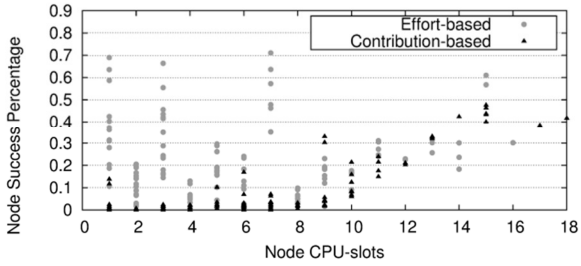

Figure 4. Node success percentage versus node CPU-slots

Finally, the NSP difference between the powerful and the powerless nodes using the effort-based strategy happens because the powerful nodes always carry out a task during the simulation. Therefore they can never share all their resources.

V. Conclusions and Further Work

The aim of this study is to understand different strategies for incentivizing a resource sharing ecosystem. We conducted a study based on simulations, which compared the impact of contribution-based and effort-based incentives on the resource sharing process in a scenario of hardware heterogeneity. Experiments on different scenarios show similar result trends regardless of the overlay network topology used. When an effort-based strategy was used, the nodes showed a higher willingness to cooperate, no matter how much resources they have. However when a contribution-based strategy was used, the nodes tended to cooperate only with their equals.

In global terms, the metric percentage of *tasks fully satisfied* is higher in the contribution-based strategy, because the nodes are self-providing the resources they need. This leads the system to have a lot of resources underused since the nodes do not have enough CPU to share with others. Hence, the resource sharing ecosystem becomes an inequality system where unique benefiters are powerful nodes that have enough resources to do their tasks and only occasionally collaborate with others. Collective satisfaction is achieved when most nodes are satisfied, although the percentage of success in the system is lower. It can only happen when all nodes can be equally graded like it occurs in the effort-based strategy. The differences between both strategies concern only participants with scarce resources. Nevertheless, in effort-based strategies the cooperation relationship is fairer, giving powerless nodes the opportunity to fulfill their resources needs.

These results provide a first understanding of the impact produced by both strategies on cooperation, when nodes play the same strategy. However, in peer-to-peer and volunteer computing applications, the participants typically use different strategies with different nodes, depending on the parameter that they want to optimize and the current situation. The study of the impact of system dynamics is part of the future work.

Another point to address as future work is the neighbor nodes information inference. In every experiment we have assumed that all participants know the maximum amount of resources, i.e. CPU slots of the node that their neighbors can share at maximum. This is a strong assumption since it is not ease to obtain in real world applications. It is however a critical value needed to measure the effort of each node, when it has to decide with whom to share some slots. Therefore, our future work will be also focus on finding a way to identify the maximum resources available in each participant.

Acknowledgment

This work was partially supported by Fondecyt (Chile) No. 1120207, LACCIR No. R1210LAC002 and Spanish MEC DELFIN TIN2010-20140-C03-01, and also CONFINE Project: FP7-288535 and Clommunity Project: FP7-317879.

References

[1] D. Anderson, “Boinc: A system for public-resource computing and storage,” in Proc. IEEE/ACM Workshop on Grid Comp., 2004, pp. 4-10.

[2] L. Sarmenta and S. Hirano, “Bayanihan: Building and studying web-based volunteer computing systems using java,” Future Generation Computer Systems, vol. 15, no. 5, pp. 675–686, 1999.

[3] G. Hardin, “The tragedy of the commons,” Science, vol. 162, no. 3859, pp. 1243–1248, 1968.

[4] M. Sirivianos, J. Park, R. Chen, and X. Yang, “Free-riding in bittorrent with the large view exploit,” in Proc. of IPTPS’07, 2007.

[5] B. Fan, D. Chiu, and J. Lui, “The delicate tradeoffs in bittorrent-like file sharing protocol design,” in Proc. IEEE ICNP’06, 2006, pp. 239–248.

[6] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee, “Bittorrent is an auction: analyzing and improving bittorrent’s incentives,” ACM SIGCOMM Computer Comm. Rev., vol. 38, no. 4, pp. 243–254, 2008.

[7] A. Legout, N. Liogkas, E. Kohler, and L. Zhang, “Clustering and sharing incentives in bittorrent systems,” ACM SIGMETRICS Performance Evaluation Review, vol. 35, no. 1, pp. 301–312, 2007.

[8] R. Rahman, M. Meulpolder, D. Hales, J. Pouwelse, D. Epema, and H. Sips, “Improving efficiency and fairness in p2p systems with effort-based incentives,” in Proc. IEEE ICC’10, 2010, pp. 1–5.

[9] W. Poundstone, “Prisoner’s dilemma,” 1992.

[10] D. Vega, E. Medina, R. Messeguer, D. Royo, F. Freitag, S.F. Ochoa, J. Pino.“Characterizing the effects of sharing hardware resources in mobile collaboration scenarios”, in Proc. of CSCWD’11, 2011, pp.465-472.

[11] A. Mislove, M. Marcon, K. Gummadi, P. Druschel, and B. Bhattacharjee, “Measurement and analysis of online social networks,” in Proc. ACM SIGCOMM’07. 2007, pp. 29-42.

[12] A. Barabási and R. Albert, “Emergence of scaling in random networks,” Science, vol. 286, no. 5439, pp. 509–512, 1999.