Journal of Universal Computer Science, vol. 20, no. 13 (2014), 1738-1765
submitted: 29/1/14, accepted: 6/9/14, appeared: 28/11/14 © J.UCS

Understanding Collaboration in Volunteer Computing Systems

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

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

Abstract: Volunteer computing is a paradigm in which devices participating in a distributed environment share part of their resources to help others perform their activities. The effectiveness of this computing paradigm depends on the collaboration attitude adopted by the participating devices. Unfortunately for software designers it is not clear how to contribute with local resources to the shared environment without compromising resources that could then be required by the contributors. Therefore, many designers adopt a conservative position when defining the collaboration strategy to be embedded in volunteer computing applications. This position produces an underutilization of the devices' local resources and reduces the effectiveness of these solutions. This article presents a study that helps designers understand the impact of adopting a particular collaboration attitude to contribute with local resources to the distributed shared environment. The study considers five collaboration strategies, which are analyzed in computing environments with both, abundance and scarcity of resources. The obtained results indicate that collaboration strategies based on effort-based incentives work better than those using contribution-based incentives. These results also show that the use of effort-based incentives does not jeopardize the availability of local resources for the local needs.

Keywords: Volunteer Computing, Collaboration Strategy, Software Design, Effort-Based Incentives, Resource Sharing.
Categories: C.2.1, C.2.3, C.2.4, C.2.m

1 Introduction

Volunteer computing typically involves a networked and distributed environment, through which heterogeneous devices share part of their resources (e.g., CPU or memory) to help other participants perform certain tasks [Sarmenta and Hirano, 99; Anderson, 04; Chandra and Weissman, 09; Kannan et al., 11]. Although these applications are most often centrally managed, there are proposals that follow a distributed approach [Lazaro et al., 09; Chmaj and Walkowiak, 12] and others that are based on social behaviors [Chard et al., 12].

The distributed environment supporting this computing paradigm can be seen as a heterogeneous mesh, where each network node represents a participant. When a node requires more resources than those that it has locally available, it must borrow the extra resources from other nodes. As nodes request resources from other nodes and also provide their own resources to others, a sharing ecosystem is created in which the participants share time slots of their resources. In order to have a sustainable ecosystem, the participants must return the borrowed resources once they are no longer needed.

This computing paradigm has been used to support several types of applications, such as crowdsensing [Campbell et al., 06; Ganti et al., 11], mobile collaboration [Ochoa et al., 11; Monares et al., 11] and grid computing [Cankar et al., 13; Bittencourt et al., 12]. Over recent years volunteer computing has started to be used to support not only collaborative solutions, but also some ubiquitous applications and ambient intelligence systems.

In volunteer computing, counting on a free and unrestricted access to the shared resources, without the need to contribute to the environment would probably lead to the collapse of the environment. Therefore, that solution becomes useless. This problem is known as the “tragedy of the commons” [Hardin, 68], and it is typically caused by nodes engaging in free-riding [Sirivianos et al., 07].

Designers of volunteer computing applications must avoid the environment collapse, therefore the devices running these applications must contribute with resources that can be used by others. However, in order to do that, these designers have to deal with two dilemmas: (1) how many resources they should share with others, and (2) when to share them.

Many previous proposals use incentives mechanisms to promote collaboration among nodes, and thus dealing with the stated problem. These incentives regulate the global and individual benefits, by encouraging nodes to collaborate, granting them a fair return for their contributions. Traditional methods typically use absolute metrics of contribution (e.g.,, each node calculates the expected return of its decisions) to determine when a resource can be shared with other peers. This approach typically discriminates nodes with few resources and does not provide a fair scenario for sharing resources [Vega et al., 13a].

Trying to help designers address the stated dilemmas, this paper extends on a previous work [Vega et al., 13a] and presents a study that explores the use of different collaboration strategies to motivate network nodes to share resources with others. The impact of using these strategies is analyzed from the contributor and environment point of view.

In order to determine the impact of using each strategy we performed several simulations that considered collaboration environments with abundance and scarcity of resources. The obtained results show that the effort-based incentives allow participants to accomplish at least the same number of tasks as when contribution-based incentives are used. Moreover, in the first case the nodes having few resources are not discriminated by the rest of participants, resulting thus in a more robust and fair collaboration process. Several features of the collaboration process are analyzed and discussed in this paper in order to demonstrate that this finding is not by chance. These study results help designers make more informed decisions during the design of volunteer computing applications.

The next section presents the related work. Section 3 describes the framework used in this study. Section 4 shows and discusses the obtained results. Section 5 presents the conclusions and the future work.

2 Related Work

Incentive mechanisms for promoting cooperation have been studied on Peer-to-Peer (P2P) resource sharing and volunteer computing scenarios in order to convey selfishness behaviors from both: a theoretical and practical point of view. Theoretical approaches address the cooperation problem through game theory, typically using some variation of the prisoner's dilemma game [Poundstone, 92]; e.g., the iterated prisoner's dilemma. In the case of repeated encounters among nodes (i.e., potential collaborators), positive interaction is very important for obtaining sustainable cooperation among participants [Rand et al., 09].

The theoretical models have helped designers understand what properties must hold such incentive mechanisms to fairly allocate heterogeneous resources in a cooperative way. Two properties are especially desirable in the design of the node cooperation strategy; *incentive-compatibility* and *envy-freeness,* to impose a notion of fairness on the outcomes of every action made by the nodes. The first property means that players cannot improve their utility by lying about its resources. The second one indicates that players cannot prefer to use the resources of a particular participant. In [Feldman and Lai, 11] the authors established that at least one mechanism exists that holds both properties for agents with heterogeneous capacities, if there are only two kinds of shared goods or if the individual goods' values are binary.

Another approach for providing fairness of effort-based strategies was proposed by Santos et al. [Santos et al., 13] who proposed the Quid Pro Quo (QPQ) mechanism to assign tasks on a self-organized and distributed scenario. QPQ also holds the envy-freeness property and the incentive-compatibility. However, as the previous theoretical models, it implicitly assumes that each participant scrupulously follows the specification of these mechanisms.

The practical approaches to address fair cooperation are mainly based on the proposal of BitTorrent [Cohen, 08]; a popular P2P protocol for file distribution. A key for the BitTorrent success lies in its Tit-For-Tat strategy, a reciprocity based mechanism, which works reasonably well to foster cooperation among downloading peers. In this scenario, some studies argue that nodes contributing more resources should receive in return better service than those that contribute less [Fan et al., 06; Levin et al., 08]. These contribution-based mechanisms are simple and efficient in most cases; however they are unfair [Bharambe et al., 06; Piatek et al., 07], because some nodes end up contributing much more than they download.

In [Fan et al., 06] the authors present a general heterogeneous model to evaluate the tradeoff between performance and fairness in BitTorrent systems. This work shows that the current protocol used in BitTorrent is only one of many possible solutions; in this case, the fairness metric is used to indicate incentive compatibility. The work also proposes three rate assignment strategies to optimize respectively one of the following variables: average downloading time, perfect fairness or max-min allocation. They also present a simple design knob that helps designers analyze the possible tradeoffs (in terms of performance) that the use of a certain rate assignment strategy has on the collaboration space. The performance is quantified in terms of both, average downloading time and fairness. A performance evaluation is finally conducted to show the merits and properties of BitTorrent-like protocols.

Similarly, the proposal in [Jia et al., 11a] focuses on the performance analysis of different policies using Tit-For-Tat. These authors introduce a model that characterizes the relationship between a peer's performance and the design parameters of the BitTorrent protocol. These parameters determine the effectiveness of the incentive mechanism for the nodes. This model is then used to determine how the incentive mechanism can be adjusted to enhance reciprocity or reduce inequity in the collaboration scenario. 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 [Legout et al., 07]. Hence, nodes with scarce resources suffer discrimination because they do not have enough resources to contribute. While these works focus on a particular strategy, we analyze the performance of different collaboration incentives from a higher level.

Recently, a new practical scenario, named private BitTorrent communities, has appeared [Zhang et al., 10]. Such a proposal aims to motivate the resource uploading in P2P networks. Community administrators specify a minimum uploading threshold that must be addressed by the community members. In this way, it is guaranteed that each peer provides a certain level of contribution to the community. This mechanism, known as Sharing Ratio Enforcement (SRE), is very effective in increasing supply [Chen et al., 10]. However, [Jia et al., 11b] show that SRE has two undesired negative effects: (1) peers are forced to seed for long times and (2) SRE discriminates against peers with low bandwidth capacity.

Although most of the BitTorrent incentives are contribution based, Rahman et al. [Rahman et al., 10] show that these systems can also be used with effort-based incentives. The use of such a strategy showed a greater utilization of the available resources and a reduction of the download times in slow peers.

The problem of finding appropriate incentives to make nodes cooperate in a distributed collaborative system is also present in cloud computing scenarios. In order to address the problem, Cloud@Home [Cunsolo et al., 09] combined the cloud and volunteer computing paradigms, generalizing volunteer computing environments using cloud principles and techniques. Their vision is that user-contributed resources can form a cloud ready to inter-operate with the commercial ones, but they maintain a centralized structure within the clouds.

A different approach is proposed in the project Cloud4Home [Kannan et al., 11], which is focused on edge storage. Cloud4Home uses virtualization technologies to create data services, so that both, the locations of data objects used by these services and their processing are changeable. Therefore, the data manipulation can be adapted to needs and resource available for the collaborative application that uses the data.

Chandra and Weissman [Chandra and Weissman, 09] introduce the concept of nebulas, which is an alternative approach for creating cloud infrastructures using distributed voluntary resources. These nebulas enable both storage and computation, critical aspects to achieve locality for data-intensive computing. The proposal also presents the requirements and challenges for hosting such services on volunteer platforms, and some solutions to deal with these challenges.

Lazzaro et al. [Lazaro et al., 09] address this problem in cloud computing, desktop grids and peer-to-peer networks using non-dedicated and distributed resources. These researchers propose a model known as contributory computing, where users contribute their resources to be used collectively. They also propose a layered architecture to support the model and discuss how the existing technologies can be used in a possible implementation of the system. Although this proposal is interesting, it does not take into account incentives for participation.

[Chmaj and Walkowiak, 12] propose a P2P approach to support volunteer computing. The proposal is focused on decision strategies developed for the system participants. Using simulation the authors evaluate its proposal in various scenarios. The obtained results indicate that the strategy known as peer-to-peer flow provides lower operational cost of the computing system, compared to other decision strategies such as unicast and anycast flows. According to the study results, the appropriate selection of the decision strategy impacts significantly on operational cost of the system.

Chard et al. [Chard et al., 12] propose the use of Social Cloud to share resources. A resource could encompass people, information, computing capacity, or software licenses—hence, a resource provides a particular capability that can be used by other members of a group or community. Resources shared in a Social Cloud are by definition heterogeneous and potentially complementary. In order to participate in a Social Cloud, each user must contribute with a certain amount of their resources. The sharing is controlled (or regulated) by a socially oriented market place, which adapts common allocation protocols to a social context.

Although these proposals are interesting and address the stated problem for particular computing scenarios, none of them provide particular answers to the design questions that appear at the application layer; for instance, what is the role played by the network topology on the effectiveness of the collaboration process? How does the resource availability impact the effectiveness of the collaboration strategy? Is there a collaboration strategy that is appropriate for most volunteer computing scenarios?

Using simulations to answer these questions, we analyze the effectiveness of various collaboration strategies considering abundance and scarcity of resources, and also two typical network topologies. We also explore the impact of using various collaboration strategies in a same distributed environment. The next section describes the experimentation framework involved in this study.

3 Experimentation Framework

The simulations conducted in this study were performed with the cooperative game simulator described in [Vega et al., 11]. This tool allows us to configure a large number of scenarios, reproduce the experiments and isolate the variables to be observed. The time discrete simulator is also able to extract statistical results from the node’s behavior when the nodes play an iterated prisoner dilemma game [Poundstone, 92].

We have considered a stationary network scenario for the simulations as a way to isolate the effect produced by the network topology and nodes features. Every network used in this study considered 125 nodes with different amounts of CPU slots (i.e., resources) for sharing. This number of network nodes was determined based on a previous study [Vega et al., 13b], where the authors discovered that although the distribution of the collaboration NSP (Node Success Percentage) follows the same pattern, the global cooperation willingness decreases with the network size. Hence, a small network was selected in order to assure a large margin for cooperation improvement. The links between nodes were considered homogeneous, bidirectional and stable; i.e., they did not change during the experiment.

Every experiment involved simulations performed over a discrete scenario with 250 rounds. The results of a previous study show that this number of rounds is enough to extract significant statistical conclusions, after discarding the first fifty transitional rounds [Vega et al., 11].

Next we briefly describe how each round is solved. At the beginning of each round, all nodes are pooled into two categories: *active nodes* – meaning that they are running their own task on the topology – and *waiting nodes* – that have no own tasks running at the moment. Then, waiting nodes decide if they want to perform a new task using a uniformly distributed probability of 50%. If they are interested in doing that, they then start the request-response algorithm as follows:

a) Firstly, each requester calculates the CPU-slots required for the new task and the running time. Then, it does a self-request for the needed number of CPU-slots. However, if the own free resources are not enough to complete the task, the requester will send a unitary request to each adjacent node for the remaining CPU-slots. After all requester nodes have made their request, the response phase starts.

b) In the response phase, all nodes that received slot requests use their resource sharing strategy to decide if they wanted to cooperate and share its free CPU slots. A response message is then sent for each request.

c) Finally, requester nodes determine if they have enough CPU-slots to run the desired task, and the superfluous CPU-slots assigned are randomly discarded. If a node gets the minimum amount of slots, its state changes to active and the borrowed slots will remain blocked until the end of the task.

3.1 Resource Sharing Strategy

Every node plays an Iterated Prisoner's Dilemma game with its neighbors whenever the node needs more CPU slots to complete a task. In this game, two players involved in a potential collaboration process choosing between cooperation and defection. The decision to choose one action or another can be based on the trust in others and their reciprocity. The consequences of the participants' choices have a different impact on the global community and also on their own probability to obtain resources from other nodes. This study evaluates these impacts when the decision is based on two approaches: *contribution-based* and *effort-based* collaboration strategies. The effort-based mechanisms are based on the proposal of [Rahman et al., 10] that considers the nodes' effort, instead of taking into account only the absolute value of the nodes' contributions. Next, we briefly describe the five collaboration strategies used in these experiments.

- *Contribution-based Incentive*: This approach considers the absolute contribution of the requesting node, when the requested nodes have to decide whether to cooperate or defeat. A well-known strategy to maximize the payoff when using this approach is the tit-for-tat strategy. Our contribution-based incentive implementation follows the tit-for-tat strategy proposed by Rapoport, A. [Axelrod, 80]. The success of BitTorrent protocol is an example of a real application using Tit-For-Tat as a core strategy. Nodes contributing more resources should receive, in turn, a better service than those that contribute less [Fan et al., 06; Levin et al., 08]. However, while in the original proposal all participants always start their interaction with another participant cooperating, in our implementation each participant makes a new random choice every time is meets a new participant. Hence, at first glance a participant assumes a cooperative attitude in favor of some neighbors, while at the same time it assumes a non-cooperative attitude with others nodes. This design decision helps us populate the network (in each simulation scenario) with several random configurations and different initial behaviors. Using this 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 and we consider that node B is a collaborator with 50% probability.

- *Contribution-based Incentive with Forgiveness*: This strategy implements a naive peacemaker incentive, which is a variant of tit-for-tat. Here, the participants sometimes make peace by co-operating in lieu of defecting. The objective is to avoid being highly harmed by single and/or random defection of another co-player. The study and analysis in [Axelrod, 84] shows that forgiveness is a necessary condition to achieve cooperation. Several complex strategies have been evaluated trying to reach forgiveness [Axelrod, 80]; however none of them have improved the performance of the original tit-for-tat strategy, when it is applied in a competitive environment. In cooperative real applications forgiveness is implemented in order to speed-up cooperation reconstruction after single defections. Most BitTorrent clients use opportunistic unchoking to compensate nodes' mutual defection caused when one peer's chocking policy replaces one link [Seung et al., 05]). In our work we decided to implement a simple mechanism where participants forgive a neighbor after 15 consecutive rounds of defections. Particularly, when node B has defeated node A during a consecutive number of rounds, node A considers it as an unknown node; therefore node A will cooperate with B with 50% probability.
- *Slot Effort-based Incentive*: This strategy considers the node's relative contribution or effort to determine if it cooperates or defeats. The node contribution is defined as the percentage of available resources that a node shares with others. 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 CPU 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 nodes A and B have not met before, and B has free CPU slots, then A evaluates each slot with a 50% cooperation probability. The metric is inspired by the work of Rahman et al. [Rahman et al., 10] in which authors rank and prioritize BitTorrent chunk requests from another peer, by using the percentage of bandwidth devoted to the other participant during a slicing windows of time. We adapted such a proposal because in our scenario each CPU slot is not transferable - in contrast of a BitTorrent schema where chunks (equivalent to our CPU slots), could be obtained from multiple sources -. Hence, we have used tasks' slots instead of bandwidth to calculate the ratio between shared resources and total

amount of resources. In our strategy the ratio has been used as a probability to cooperate instead of a ranking metric. The slicing window is just one interaction round. The idea behind the effort-based incentive is to create a mechanism that does not harm nodes having few CPU slots. At the same time, this strategy aims to maintain a reciprocity policy similar to tit-for-tat.

- *Task Effort-based Incentive (one-by-one)*: Similar to effort-based incentive for sharing CPU slots, this strategy is based on a node's relative contribution. However, in this case contribution is measured at task level, instead of CPU slot. This strategy is a new variation of the Slot Effort-based inspired by the work of Santos et al. [Santos et al., 13], where the authors discuss – evaluating complete tasks instead of slots – how the granularity of this metric affects the performance of the sharing strategy. As in the previous strategy 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. This probability is now calculated as the ration between the *mutual accepted tasks* and the *maximum of expected tasks* that B could run. The *mutual accepted tasks* represents the sum of tasks that B accepted during last round from both node A and internal requests. The *maximum of expected tasks* is the number of B slots divided by the simulation average cost of tasks (in CPU slots). Similar to the previous strategies, the nodes consider a 50% cooperation probability if they have not met before. Note that in this strategy we included the equivalent effort devoted to nodes themselves in the calculation of the mutual accepted tasks. It increases the diversity of probabilities P, avoids binary evaluations, and aims to maintain a reciprocity policy.
- *Task Effort-based Incentive (local)*: It is based on a variant of the task effort-based incentive presented above (called one-by-one), but instead of evaluating just the effort perceived by a particular node, the mechanism evaluates the effort noticed by a sub-network cluster. In order to evaluate the cooperation process using this incentive, we have simulated the strategy as follows. For a given round, a certain node A answers 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 tasks that B accepted from its neighbors and the maximum of expected tasks. As in previous strategies, if nodes A and B have not met before, then they have a 50% cooperation probability. The intention of the mechanism is to evaluate positively, nodes that devote resources to others, no matter if it implies a direct benefit to the requester. As a consequence, the direct reciprocity between requester node and the requested cannot be assured, only the local one. The implemented strategy is a variation from the Task Effort-based (one-by-one) strategy, inspired by some ideas from network exchange theory [Willer, 99]. This theory discusses, among other things, how the participants' outcome can be partly rooted in the structure of a (social) network, because they hold pivotal positions on the network, or in other case they are in a dependent position for responding to a request.

3.2 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 only with the neighboring nodes according to the network topology. The node placement is randomly determined during each simulation initialization, while respecting the network topology.

The type of resource to be shared was just CPU slots; however any other type of resource can also be shared in a real situation, e.g., memory and the network cards. The CPU slots were randomly distributed among the nodes during each simulation initialization, using a discretized normal distribution with a lower bound of one CPU slot per node.

In each request-answer round, if a node completed its own tasks, such a node will generate a new task with a given probability (50% in baseline cases). When a node decides to perform a task, it calculates the task cost in terms of needed CPU slots and time; i.e., the amount resources and rounds required to accomplish the task. The CPU requirements are randomly determined using a discrete normal distribution, originally with a mean of six slots and a variance of two slots. The minimum number of CPU slots considered in a request is one. The number of rounds needed to perform the task is also randomly determined using a discrete uniform probability distribution, with a minimum value of one and a maximum of three rounds.

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 its neighboring nodes for the CPU slots needed to accomplish this task. Once the 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 enough slots are not obtained for the tasks, all obtained slots are released, therefore allowing them to be used to support other requests.

3.3 Network Topologies

This study considered networks with a small-world and torus topology (Fig. 1). Social networked scenarios exhibit properties that are usually present in small-world topologies [Mislove et al., 07; Barabási and Albert, 99]. The small-world effect has also been considered for collaborative and peer-to-peer networking applications. Two characteristics distinguish this network 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.

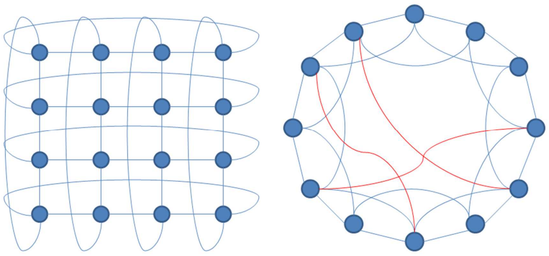

Figure 1: Torus and small-world network topologies

We have also used a torus topology to assess if there is any dependence of the obtained simulation results on the network topology. The torus topology involves an N-dimensional array, 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 the topological properties; therefore the properties of one node are representative for the rest of the network nodes.

3.4 Collaboration Scenarios and Metrics

In this study we used collaboration scenarios with resource scarcity and abundance. In scarcity scenarios, the amount of resources available is (in average) twice the amount requested per node and round. However, in resource abundance scenarios, the available resources are four times the requested amount of resources. We have assumed that these resources will never be 100% used, as when they are required to support local tasks. Each node can perform at most, one task during a maximum of three rounds. The simulation results were assessed using the following metrics:

- *Node Cooperation Coefficient:* This metric is calculated as the ratio between the amount of requested CPU slots and the number of positive answers – in terms of CPU slots – obtained during the simulation, regardless if the lent slots were used or not. It is therefore a direct measure of the nodes' willingness to collaborate with each other.

- *Node Success Percentage (NSP):* This value is 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.

- *Envy-fairness:* For a given node n with i neighbors, the envy-fairness metric is calculated as the difference between success percentage of node n (NSP_N) and the average node success percentage of its i neighbors. Hence, it is a comparative measure of how envious a node is, considering its neighbors' tasks success. . It is commonly referred to as node success correlation.

- *Tasks Fully Satisfied:* This value represents 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.

- *Utilization of CPU slots:* We have here several sub-metrics related to the utilization of 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.

The first three were measured directly on each participating node, and the last two considered the whole network.

4 Effort-Based Versus Contribution-Based Strategies

The simulation results allow us to understand the impact of each incentive strategy (described in section 3.1) in every collaborative scenario (i.e., in resource scarcity and abundance), considering static network topologies and heterogeneous resources distribution. In order to determine the impact of effort-based and contribution-based approaches, we have performed simulations on a peer-to-peer network with a small-world topology in a resource scarcity scenario, and 125 nodes playing the Iterated Prisoner's Dilemma game. The network topology and the scenario with resources scarcity were selected for this study as they represent a challenge for the collaboration process. If the features of the scenario improve, then we can expect that the number of collaboration instances also improves.

Figure 2 shows the cooperation coefficient – round by round – for contribution-based and effort-based strategies. The results indicate that strategies following an effort-based approach obtain a more important number of collaboration instances than those implementing contribution-based incentives. The results also show that in the contribution-based strategy without forgiveness, i.e., replying strictly and reciprocally to the previous action of the other node, all nodes start to reject requests of other nodes after the first 50 transitional rounds.

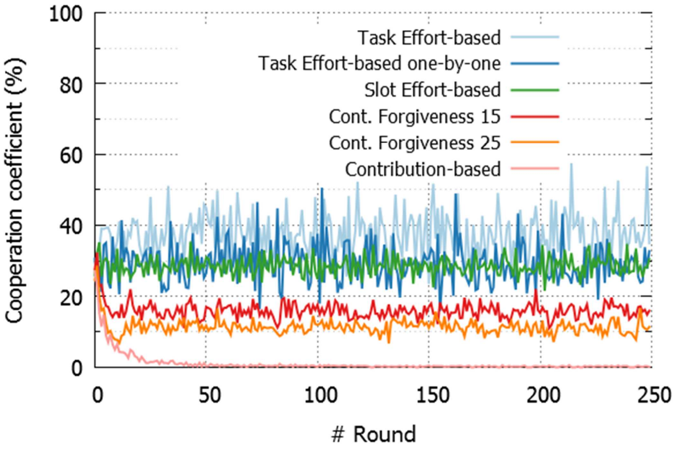

Figure 2: Nodes cooperation results, round by round

When a forgiveness variant is applied, the nodes are able to recover from this situation and cooperate between 17% and 11% of the time, depending on how many rounds they wait to forgive. However, in all effort-based strategies, the nodes receive more contributions from their neighbors and thus achieve a higher satisfaction. In order to determine if these initial findings can also be found in other collaboration scenarios, we next analyze the main indicators involved in the collaboration process.

4.1 Analysis of the Cooperation Coefficient

Figure 3 presents the results of the cooperation coefficient in both, resource scarcity and abundance scenarios. The cooperation coefficient shows high variations in the minimum and average values, considering the use of contribution-based and effort-based incentives. This indicates that the system swings between periods with high and low cooperation.

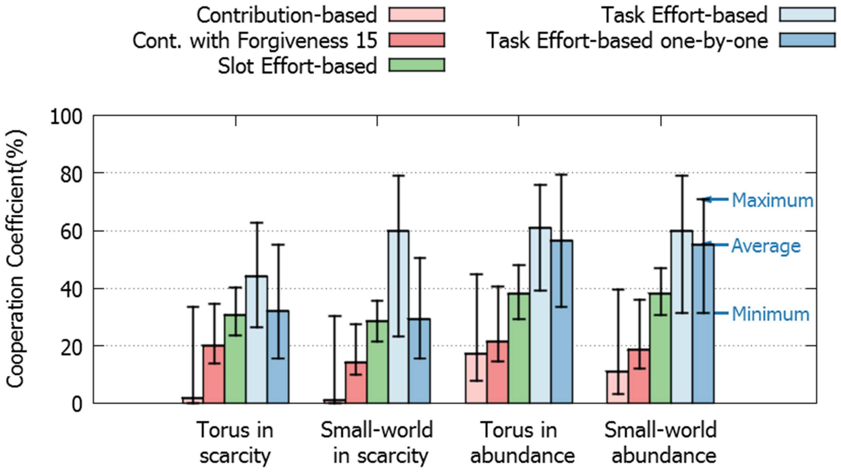

Figure 3: Cooperation Coefficient for Torus and Small-World Topologies

The results indicate that in all scenarios and topologies, the effort-based incentives obtain a higher collaboration level than those implementing contribution-based incentives. In our simulation, the reduction of 50% of available resources –from abundance to scarcity scenarios – represented a significant 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. These results support the previous findings. It is also 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. Moreover, these results show that the network topology does not play a significant role in the cooperation process, which is good news for the systems designers, due they can forget the presence of such a variable.

4.2 Analysis of the Node Success

Trying to isolate the impact of these incentive strategies, we have computed the Node Success Percentage (NSP), which is a direct measure of the node request satisfaction during the collaboration process. Similar to that shown in Figure 3, Figure 4 shows the average, minimum and maximum values of the NSP.

These results reflect the fact that when nodes use a slot effort-based strategy, the average satisfaction – measured as the average NSP – is a 6% over the satisfaction when they use a contribution-based strategy. This tendency is kept in resource scarcity and abundance scenarios. The results also show that the topology of the environment does not affect this result. Once again, the results support the previous findings.

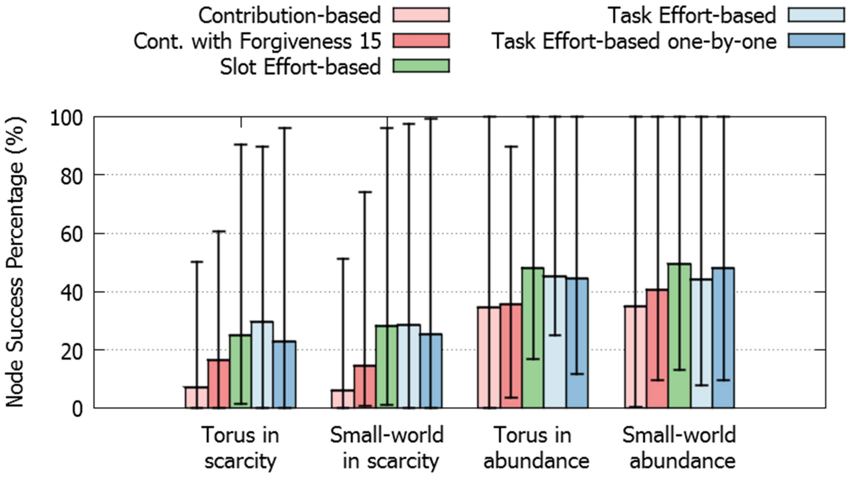

Figure 4: Node Success Percentage for Torus and Small-World Topologies

Table 1 shows the relative percentage of tasks satisfied by the nodes, depending on the topology of the environment and the level of resources availability. This relative percentage is calculated respect to the maximum number of tasks accomplished in any of the studied scenarios; i.e., that percentage does not represent the absolute value of the satisfied tasks in every scenario.

The use of satisfied tasks as metric allows us to see how useful for the end-user application was the collaboration process performed by the nodes. According to the results shown in Table 1, the collaboration process was useful because there is a small dispersion among the collaboration strategies that were analyzed (in all cases the percentage of satisfied tasks is close to the 100%).

Strategy	**Scarcity**		**Abundance**	
	Torus	**Small-World**	**Torus**	**Small-World**
Contribution-based	98%	97%	95%	97%
Contrib.-based Forgiveness 15	95%	100%	97%	95%
Slot Effort-based	100%	99%	100%	99%
Task Effort-based (local)	99%	98%	94%	96%
Task Effort-based (one-by-one)	97%	96%	98%	100%

Table 1: Percentage of satisfied tasks

Although the percentage of satisfied tasks is similar when using contribution-based and effort-based strategies, there is an important difference in the way in which such a percentage is obtained. In contribution-based collaboration the nodes use their own resources and some from their neighbors to complete the task. However, when effort-based strategies are used, the task accomplishment is mainly done through slots sharing among nodes. This shows that the nodes can adopt a more generous position when participates in volunteers computing scenarios. Such a generous position does not put at risk the availability of resources to address the local requests. This is also an important finding for designers of these systems, since if every participating node uses a single collaboration strategy, the designers do not need to be worried about determining a limit of resources to be shared. Clearly this finding does not apply if the nodes use different strategies for sharing resources.

Analyzing the results of the three alternatives of effort-based incentives we can see that the *slot effort-based* strategy reaches the best scores in almost any scenario. This means that this strategy not only promotes the collaboration, but also it allows nodes to accomplish (on average) a higher number of tasks. The key element that allows us to understand this result is the granularity of the shared resources. In the slot effort-based incentive every slot is requested and shared individually. However, in task effort-based incentives the requested or shared resources are those needed to accomplish the task; i.e., blocks with a variable number of CPU slots are requested and shared among nodes as a single unit. This imposes additional restrictions to the collaboration process. Therefore, the collaboration process supported by effort-based strategies will be more effective and useful, when smaller is the resource being requested and shared.

Figure 5 shows an Empirical Cumulative Distribution Function (CDF) plot for the NSP obtained by the participating nodes. The CDF is presented for scarcity and abundance scenarios, using a Torus topology. The results for both collaboration strategies keep the tendency that was observed in the previous analysis; i.e., the effort-based strategies allow accomplishing at least the same number of tasks as the contribution-based strategies. However, in the first case the nodes perform a fairer collaboration process; i.e., the small nodes are not discriminated when asked for help.

Figure 5 also shows that there is a high number of nodes with low NSP values (below the average) in scarcity scenarios. For instance, approximately 60% of nodes have an NSP below 25%. Moreover, 60% of the nodes have a NSP below 5% when contribution-based strategies are used. Furthermore, the NSP curve rises quickly in contribution-based strategies and reaches a maximum value barely over 50%. However, when effort-based strategies are used, the curve rises more slowly, but it reaches maximum values close to 100%. These results confirm once again that effort-based strategies are more effective than the contribution-based to promote collaboration among nodes.

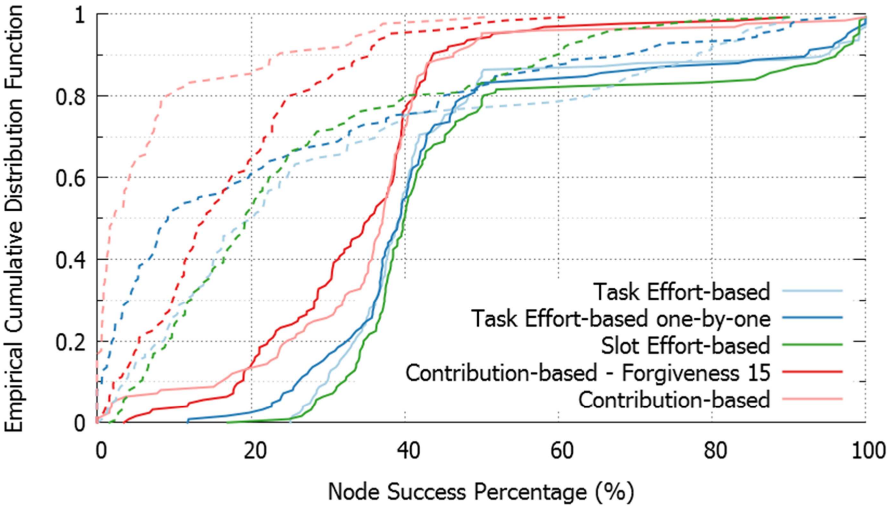

Figure 5: CDF in scenarios with resource scarcity (dashed line) and abundance (continuous line)

In abundance scenarios there are almost no low NSP values, and the curve shows a significant rise when the NSP is close to 40%. In fact, the curve looks like steps for all collaboration strategies. This behavior of the collaboration process, particularly the stationary values, can be explained by the phenomenon of networks undergoing a phase transition. A Barabási study [Barabási, 02] describes this phenomenon, and shows that it is a common behavior pattern that appears when the network nodes are “socially related”. This phenomenon means that when a given threshold (called the tipping point) is reached, all the network nodes undergo a transition phase and they start acting as a single entity. In a previous study the authors showed that this effect appears when the nodes use a contribution-based strategy [Vega et al., 13b]; however, Fig. 5 also shows this phenomenon when the nodes use effort-based strategies. In terms of cooperation among nodes, this figure confirms the previous results.

4.3 Analysis of the Envy-fairness

Similar to the previous section, Figure 6 shows the CDF plot according to the envy-fairness value recorded by the network nodes. The collaboration process was conducted in a scarcity scenario and involved a network with a Torus topology. The results show different behaviors for the analyzed collaboration strategies.

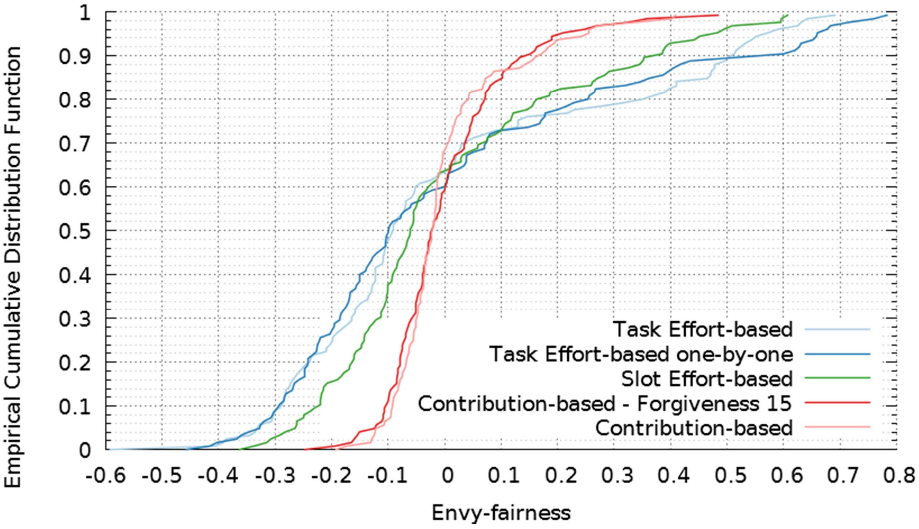

Figure 6: Envy-fairness in a resource scarcity scenario with Torus topology

Analyzing the contribution-based strategies we can see small variations in terms of envy-fairness. This means that the nodes have a satisfaction level similar to their neighbors. This result is not surprising, because the Tit-For-Tat strategy generates strong cooperation dependence among nodes. However, the effort-based strategies show higher variations of the envy-fairness metric, due the satisfaction level of a node does not depend on its neighbors' satisfaction. It is important to notice that slot effort-based strategy is, according to this metric, fairer than tasks effort-based strategies. In resource abundance scenarios the results follow the same behavioral pattern.

All collaboration strategies show approximately 60% of nodes with negative (but small) values of envy-fairness. In this case, once again the effort-based collaboration strategies have better performance than the other strategies.

Figure 7 shows the envy-fairness value achieved by each node, ordered according to their total amount of CPU slots in a Torus topology. The results show different behavior for nodes having less than 8 CPU slots than nodes with 8 or more slots. In the first case, the performance of effort-based strategies is higher than the contribution-based policies, due to the former prioritize the satisfaction of small nodes (i.e., those with few resources). Contrarily, in the second case the envy-fairness value rises linearly and it is also higher than in effort-based strategies. This means that the contribution-based strategy prioritizes the satisfaction of large nodes, which tends to be unfair.

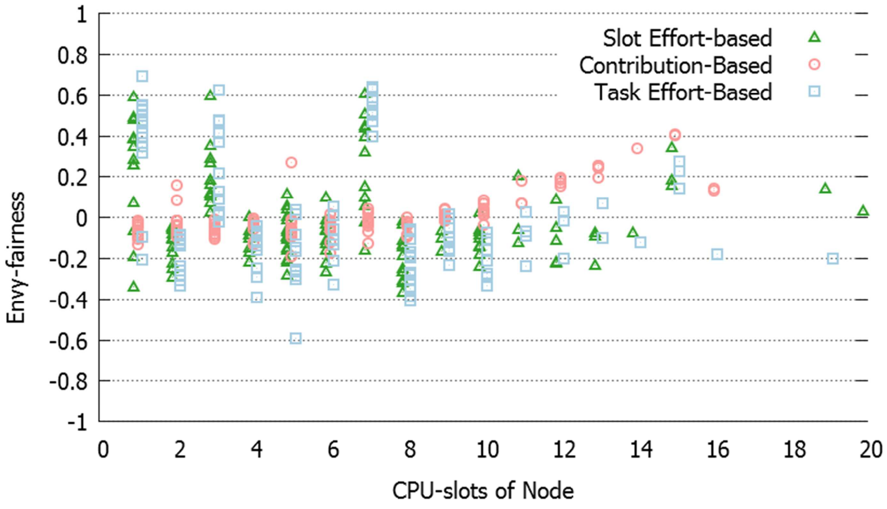

Figure 7: Envy-fairness versus CPU-slots of the nodes in a resource scarcity scenario

4.4 Analysis of the Resources Utilization

Another feature of the system that must be studied is the resource utilization. This metric provides an additional perspective of the fairness and effectiveness of the collaboration process. Figure 8 shows a histogram with the average distribution of CPU slots in the six scenarios, using a torus topology. Four metrics are utilized to understand the resource utilization: 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 results show that nodes using a slot effort-based strategy dedicate at least the same CPU slots to tasks belonging to other nodes as the slots assigned to local tasks. When the node efforts are measured using tasks, nodes act in a neutral way to maintain the equilibrium between the amounts of resources loaned and devoted to their own tasks. This almost doubles the number of free resources in comparison with slot effort-based results. When node efforts are measured using slots, instead, nodes tend to increase their solidarity towards other nodes. This solidarity helps other nodes get all the required resources for their tasks, turning the competition on cooperation, making the resource sharing process easier, and thus increasing the NSP and the system utilization. These situations do not occur when a contribution-based strategy is used by the nodes.

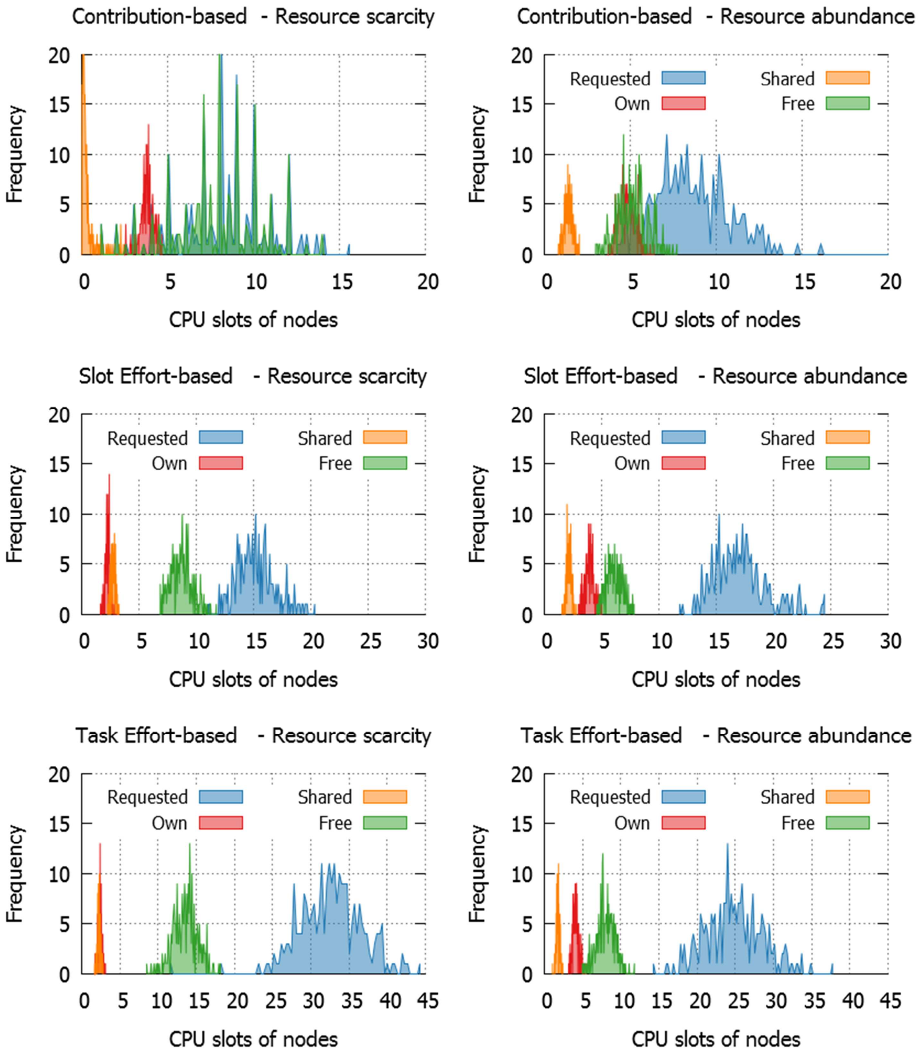

Figure 8: Histograms of average CPU slots requested, own, shared and free on different scenarios involving a torus topology

In resource abundance scenarios, this cooperation improvement is smaller, and it also leads to a reduction of resources that nodes use for themselves. However, it never turns the system to a point of solidarity, nor even neutrality. As previously stated, it is noticeable that when the percentage of requests per node and round is the same or less than in effort-based strategies, the amount of unused CPU slots also decreases significantly.

We have also to note that in all cases there is a high number of slots underused by nodes, which could contribute to increasing the value of the satisfied tasks metric. This side effect occurs because all strategies consider in their decisions only the previous relationship between the nodes. Any other factors, such as resource usage, energy consumption, continuous defection from same neighbors or equitability, are not considered at this time.

4.5 Analysis of the Impact on the Node Properties

Considering the previous results, we have to determine if the differences of performance between the collaboration strategies used in this study change with the node properties. The previous results show important differences between the maximum and minimum values of the NSP (Section 4.2) and a high number of CPU dedication for the participating nodes (in Section 4.4). This suggests that there exist node conditions or topological properties that help some nodes to achieve a better satisfaction than others.

Discarding the second reason – since the Torus topology shows almost the same behavior as Small-world – we conducted an additional simulation to measure the individual satisfaction of the nodes. Figure 9 and 10 show the NSP achieved by each node according to their total amount of CPU slots, in a scarcity and abundance scenario. The experiment involved a small-world topology and 125 nodes.

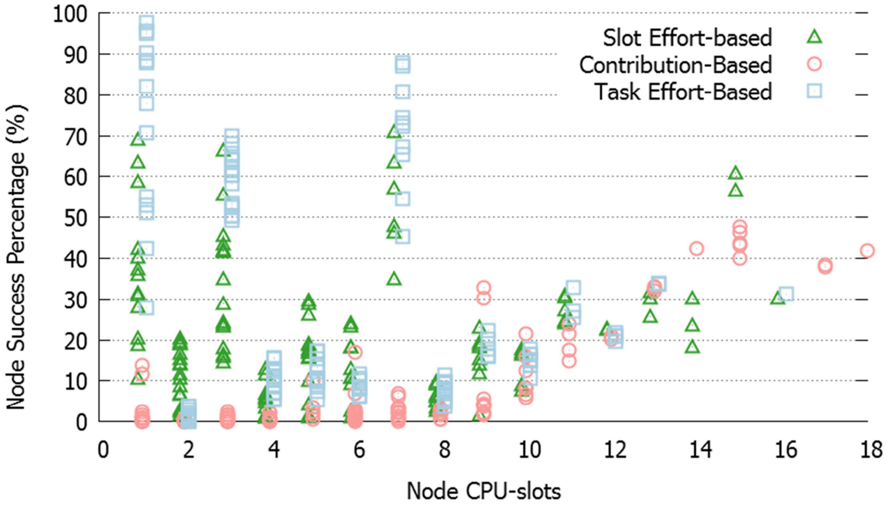

Figure 9: NSP versus CPU- Nodes slots in a resource scarcity scenario

The results show 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 similar NSP values. However, when nodes have few slots, the contribution-based strategy is unfavorable. Contrarily, when using an effort-based strategy, the success percentage is between 3% and 70%. This shows that the effort-based strategies are fairer than contribution-based strategies, when they are used in volunteer computing environments with heterogeneous participants (in terms of resources).

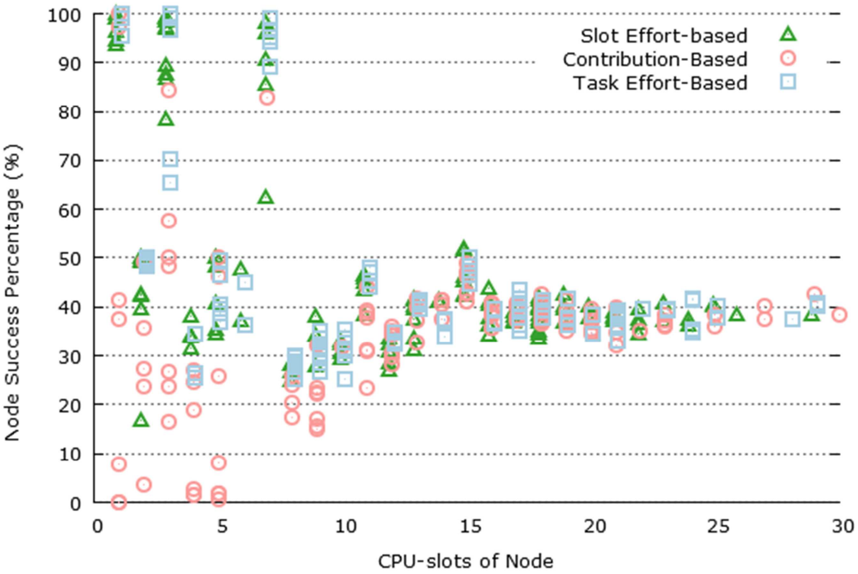

Figure 10: NSP versus CPU-slots of the nodes in a resource abundance scenario

Although in resource abundance scenarios the results follow the same behavioral pattern as in scarcity scenarios, the nodes with more resources (i.e., large nodes) behave differently. In abundance scenarios the NSP of these nodes show a low dispersion, which means that they are almost self-sufficient; therefore, these nodes provide and request few slots to the environment jeopardizing the collaboration process and the task accomplishment in small nodes.

4.6 Mixing Contribution-Based Strategies and Slot Effort-Based Incentives

The previous results allow us to understand the impact produced by the use of a particular collaboration strategy in a volunteer computing scenario. However, in several volunteer computing scenarios the application designer cannot ensure that all nodes will use the same collaboration strategy. Therefore, it is also important to understand the behavior of the cooperation coefficient when we have heterogeneity of collaboration strategies in the environment. Although that study is part of the future work, this section presents some preliminary results.

Figure 11 shows the cooperation coefficient and NSP when the collaboration environment involves different percentages of nodes playing contribution-based and slot effort-based strategies. For example, 40% / 60% means that 40% of nodes are playing a contribution-based strategy and 60% are playing the slot effort-based strategy. The simulation considered a resource abundance scenario with a Torus topology. These results indicate that the minimal and average value of NSP tends to improve with the percentage of nodes playing an effort-based strategy. Particularly, the collaboration environment shows two different behaviors depending on whether the percentage of the nodes playing effort-based strategies is below or over 60%. If that percentage is at least 60% (indicated in Fig. 11 as *40% / 60%* or higher), the cooperation coefficient and the NSP tend to be similar. In addition, these values are always over the same metrics but for the other observed segment.

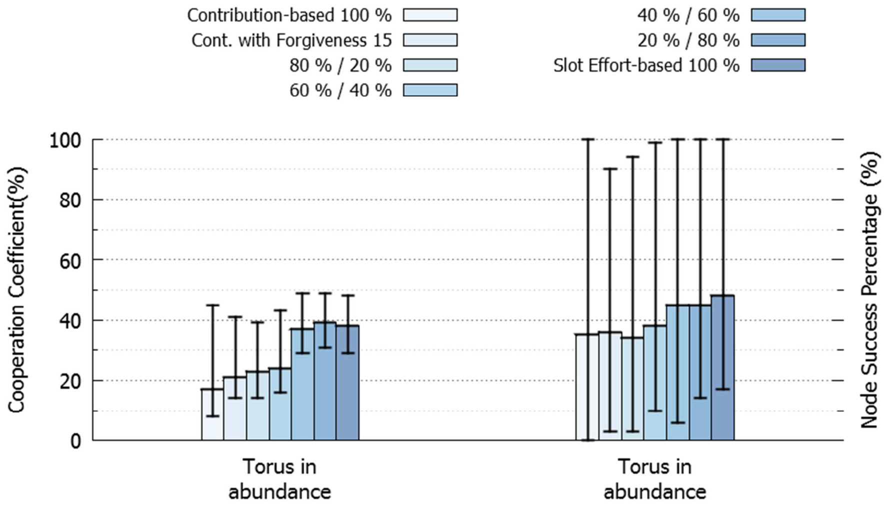

Figure 11: Cooperation coefficient and NSP mixing collaboration strategies

Although these results are still preliminary, they indicate that if the volunteer computing environment allows the use of heterogeneous collaboration strategies, the designer of these applications should try to ensure that the environment keeps at least 60% of nodes playing a slot effort-based strategy. That percentage seems to be the minimal threshold to ensure a good level of cooperation among nodes and also fairness of the volunteer computing environment.

4.7 Analysis of Uncertainty on Effort-based Incentives

In practical terms, one limitation of effort-based strategies is that participants must somehow know the maximum number of slots or tasks that other participants can assume in order to evaluate their effort ratio. Two basic approaches can be used to address this issue:

- *Direct or indirect measuring:* This approach requires to install a specific program on peer computers to measure and report their maximum capabilities. However, this is costly in terms of processing time; therefore most systems use indirect measures to infer the capabilities of other nodes. The main problem with using inference is that such a process could be inaccurate, and therefore the system could promote the use of incentive mechanisms that are not suitable for such a scenario.

- *Node self-reporting:* In this technique, the nodes announce their maximum capabilities each time they make a new request for slots or tasks. Honesty is a key aspect here. Nodes with many resources could try to trick the incentive mechanism, announcing less CPU slots than what they actually have, making the effort evaluation higher.

Figure 12 shows the cooperation coefficient of each effort-based strategy in a Torus network with abundance of resources. In each experiment, we forced a percentage of uncertainty – from 5% to 20% – on the nodes' maximum CPU slots for sharing.

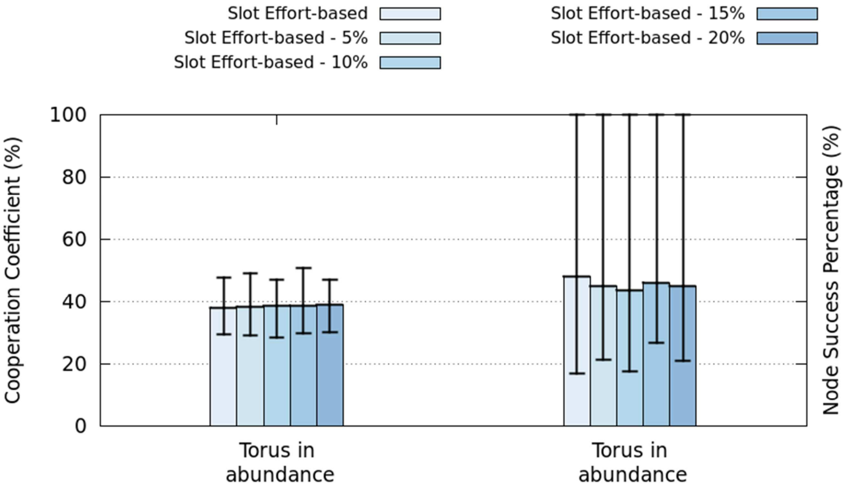

Figure 12: Results of study of uncertainty in the estimation of number of slots

By analyzing these results we can see few variations in terms of node cooperation coefficient and node success percentage. The errors in the node effort estimation seem to make no difference in the node's willingness to collaborate nor on the success of the collaboration process.

The malicious behavior of some nodes, when they self-report their own limits, can be analyzed using the results shown in Figure 13 and Table 2. Figure 13 indicates the NSP and table 2 shows envy-fairness of *liar* and *trustworthy* nodes, when all of them play the same collaboration strategy in a Torus network with resource abundance. During the experiment, we selected a percentage of nodes (from 5% to 20%), named *liars*, which report less than their maximum CPU slots – from 1 to 3 in scarcity scenarios and from 1 to 6 in abundance – in order to be better evaluated by their partners.

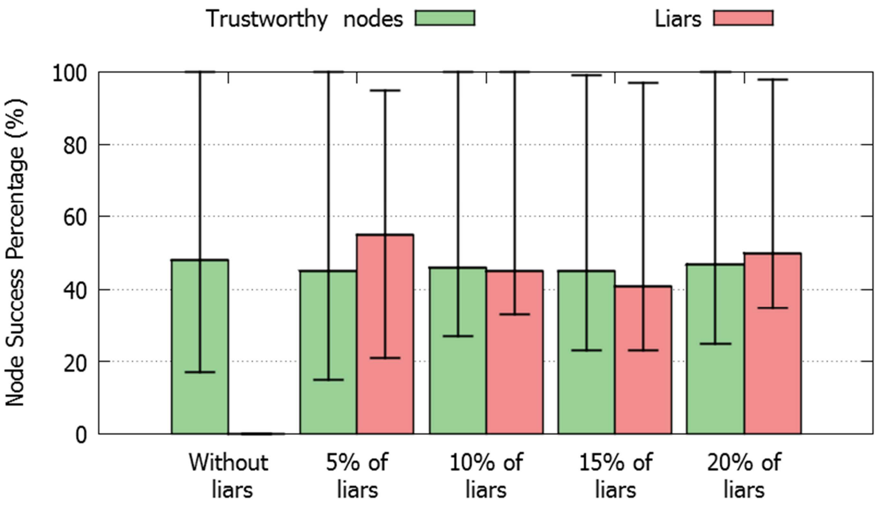

Figure 13: Node success percentage for trustworthy and liar nodes

Even if the liar nodes gain a better reputation during the first rounds, the results in Figure 13 show that there is no advantage for lying instead of truly reporting the node's maximum CPU slots. Although there is a small advantage to liar nodes when the percentage of these nodes is small, such a tendency decreases (until disappearing) when the percentage of liar nodes increases. Table 2 shows these results. There we can also see that if the environment has a 20% of liars, the envy-fairness metric becomes similar for trustworthy and liar nodes.

Strategy	**Trustworthy Nodes**			**Liar Nodes**		
	Min	**Max**	**Avg**	**Min**	**Max**	**Avg**
Slot Effort-based	-0.43	0.63	0.00	--	--	--
Slot Effort-based – 5%	-0.50	0.65	0.00	0.00	0.56	0.18
Slot Effort-based – 10%	-0.31	0.62	0.00	-0.14	0.61	0.03
Slot Effort-based – 15%	-0.36	0.66	0.00	-0.25	0.48	0.01
Slot Effort-based – 20%	-0.32	0.64	0.00	-0.34	0.63	0.00

Table 2: Envy-fairness of trustworthy and liar nodes

5 Conclusions and Future Work

The volunteer computing paradigm tends to be used more and more to support fully distributed collaborative systems; for instance those supporting loosely-coupled mobile work and crowdsensing. Various other studies have analyzed several aspects of this paradigm and have provided interesting proposals to make the collaboration process more efficient and effective. However, there is no clear understanding on the impact the collaboration strategy used by the participants has on the performance and fairness of the whole system.

The lack of knowledge on this aspect makes designers of these applications usually adopt a conservative attitude in their designs in order to avoid jeopardizing the availability of local resources to address the local needs of each node. Although the nodes having this attitude usually do not intend to take advantage of resources shared by other nodes, their lack of solidarity promotes the underuse of the devices' local resources and reduces the effectiveness of these solutions.

The study presented in this paper provides evidence that indicates that it is possible to generously share resources with others without putting at risk the local resources required for the local tasks. This study analyzed, based on simulations, different strategies for incentivizing collaboration in a resource sharing ecosystem. The impact of using contribution-based and effort-based incentives was compared in scenarios with abundance and scarcity of resources, and also involving two well-known network topologies. The result trends were similar in both network topologies, which means that this variable has a minor influence the collaboration process.

When an effort-based strategy was used, the nodes showed a higher willingness to collaborate, no matter how many resources they have. Moreover, the collaboration process was fair for most nodes, regardless of the amount of resources that they have to share with others. However, when a contribution-based strategy was used, the nodes tended to cooperate only with their equals in terms of available resources. This makes the collaboration process unfair, mainly for small nodes that cannot make important contributions.

The results also allow us to identify that the effectiveness and usefulness of the collaboration process improves by reducing the size of the shared resources. These are important findings that allow software designers to deal with the dilemmas presented in the introduction section.

In global terms, the number of satisfied tasks after a collaboration process is higher in contribution-based strategies, because the nodes are self-providing the resources they need. This leads the system to have many underused resources, since a significant number of nodes do not have enough CPU slots to share, which would allow other nodes to address their tasks. Thus, the resource sharing ecosystem becomes an inequitable 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 when effort-based strategies are used.

The differences between contribution-based and effort-based 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 resource needs.

In this study we also checked if it was necessary to precisely know the total amount of resources that other nodes have in order to use the proposed effort-based sharing incentives. Even if there are selfish nodes that lie about their own resources, the rest of the participants collectively behave as if there are no liar nodes. It does not mean however, that liar nodes cannot punctually obtain a small advantage.

The lack of global envy-freeness has also been checked during the envy-fairness study. Furthermore, we have given an upper bound in the percentage of envious nodes in each experiment – notice that all nodes with negative envy-fairness are envious of (at least) one neighbor, while all nodes with positive envy-fairness do not have to be envy-free. The later results on the influence of node properties show that fairness does not depend on network topology, but on the amount of resources shared by the nodes and the nodes willingness to collaborate.

Summarizing, effort-based incentives are good strategies to be used in volunteer computing, since they help nodes with scarce resources to improve their satisfaction without harming powerful devices. However, it generates a feeling of envy among large contributors (i.e., powerful resources). Evaluating such an effort in terms of number of tasks will further improve the number of satisfied tasks per node; however it generates even more inequity.

Perhaps the main drawback to the model is the fact that the network structure is considered static. In most P2P and ad hoc networks, the network topology evolves continuously. The next step in this research considers improving the topological model to consider the dynamism of these networks. Similarly, we have considered only one type of resources and a simple task model. It helped us to infer our conclusions in a very fair and clear way, but real-scenarios may not resemble the simulated model. The future work also considers the use of real messages and task traces from real resource sharing applications.

Finally, as a way to close the validation cycle we plan to verify these findings in several real-world deployments. Various research testbeds, such as CONFINE and Clommunity projects, give us the opportunity to test such collaboration strategies with real users and applications.

Acknowledgements

This work was partially supported by Fondecyt (Chile) No. 1120207, and by the European Community through the projects Community Networks Testbed for the Future Internet (CONFINE): FP7-288535, A Community Networking Cloud in a Box (Clommunity): FP7-317879, and also by Spanish government under contract TIN2013-47245-C2-1-R, and also by the Generalitat de Catalunya as a Consolidated Research Group 2014-SGR-881.

References

[Axelrod, 80] Axelrod, R.: Effective Choice in the Prisoner’s Dilemma. In: The Journal of Conflict Resolution. 24, 1 (1980) 3-25

[Axelrod, 84] Axelrod, R.; The Evolution of Cooperation. New York, Basic Books (1984)

[Anderson, 04] Anderson, D.P.: BOINC: A system for public-resource computing and storage. In: Proc. IEEE/ACM Workshop on Grid Computing (2004) 4-10

[Barabási and Albert, 99] Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286, 5439 (1999) 509-512

[Barabási, 02] Barabási, A.L.: Linked: The new science of networks. Perseus Publishing (2002)

[Bharambe et al., 06] Bharambe, A.R., Herley, C., Padmanabhan, V.N.: Analyzing and improving a BitTorrent networks performance mechanisms. In: Proc. of IEEE INFOCOM'06 (2006) 1-12

[Bittencourt et al., 12] Bittencourt, L.F., Miyazawa, F.K., Vignatti, A.L.: Distributed load balancing algorithms for heterogeneous players in asynchronous networks. J. UCS 18, 20 (2012) 2771-2797

[Campbell et al., 06] Campbell, A.T., Eisenman, S.B., Lane, N.D., Miluzzo, E., Peterson, R.A.: People-centric urban sensing. In: Proc. ACM Workshop on Wireless Internet (WICON'06) (2006)

[Cankar et al., 13] Cankar, M., Artac, M., Sterk, M., Lotric, U., Slivnik, B.: Co-allocation with collective requests in grid systems. J. UCS 19, 3 (2013) 282-300

[Chandra and Weissman, 09] Chandra, A.,Weissman, J.: Nebulas: Using distributed voluntary resources to build clouds. In: Proc. of Conf. on Hot Topics in Cloud Computing. (HotCloud'09), USENIX (2009)

[Chard et al., 12] Chard, K., Bubendorfer, K., Caton, S., Rana, O.: Social cloud computing: A vision for socially motivated resource sharing. IEEE Transactions on Services Computing 5, 4 (2012) 551-563

[Chen et al., 10] Chen, X., Jiang, Y., Chu, X.: Measurements, analysis and modeling of private trackers. In: Proc. IEEE Int. Conf. on Peer-to-Peer Computing (P2P'10) (2010) 1-10

[Chmaj and Walkowiak, 12] Chmaj, G., Walkowiak, K.: Decision strategies for a P2P computing system. J. UCS 18, 5 (2012) 599-622

[Cohen, 08] Cohen, B.: The BitTorrent Protocol Specification, Jan. 2008. Available at: http://www.BitTorrent.org/beps/bep_0003.html. Last visit: Jan 2014.

[Cunsolo et al., 09] Cunsolo, V.D., Distefano, S., Puliato, A., Scarpa, M.: Volunteer computing and desktop cloud: The cloud@home paradigm. In: Proc. IEEE Int. Symposium on Network Computing and Applications (NCA'09) (2009) 134-139

[Fan et al., 06] Fan, B., Chiu, D.m., Lui, J.: The delicate tradeoffs in BitTorrent-like file sharing protocol design. In: Proc. IEEE Int. Conf. on Network Protocols (ICNP'06) (2006) 239-248

[Feldman and Lai, 11] Feldman, M., Lai, J.: Mechanisms and impossibilities for truthful, envy-free allocations. In: Proc. of Int. Conf. on Algorithmic Game Theory (SAGT'12) (2012) 120-131

[Ganti et al., 11] Ganti, R., Ye, F., Lei, H.: Mobile crowdsensing: current state and future challenges. IEEE Communications Magazine 49, 11 (2011) 32-39

[Hardin, 68] Hardin, G.: The tragedy of the commons. Science 162, 3859 (1968) 1243-1248

[Jia et al., 11a] Jia, A., D'Acunto, L., Meulpolder, M., Pouwelse, J., Epema, D.H.J.: BitTorrent's dilemma: Enhancing reciprocity or reducing inequity. In: IEEE Consumer Communications and Networking Conference (CNCC'11) (2011) 705-709

[Jia et al., 11b] Jia, A.L., Rahman, R., Vinkó, T., Pouwelse, J.A., Epema, D.H.J.: Fast download but eternal seeding: The reward and punishment of sharing ratio enforcement. In: Peer-to-Peer Computing (P2P'11) (2011) 280-289

[Kannan et al., 11] Kannan, S., Gavrilovska, A., Schwan, K.: Cloud4home - Enhancing data services with @home clouds. In: Proc. Int. Conf. on Distributed Computing Systems (ICDCS'11) (2011) 539-548

[Lazaro et al., 09] Lazaro, D., Marques, J.M., Jorba, J.: Towards an architecture for service deployment in contributory communities. International Journal of Grid and Utility Computing 1, 3 (2009) 227-238

[Legout et al., 07] Legout, A., Liogkas, N., Kohler, E., Zhang, L.: Clustering and sharing incentives in BitTorrent systems. ACM SIGMETRICS Performance Evaluation Review 35, 1 (2007) 301-312

[Levin et al., 08] Levin, D., LaCurts, K., Spring, N., Bhattacharjee, B.: BitTorrent is an auction: Analyzing and improving BitTorrent's incentives. ACM SIGCOMM Computer Communication Review 38, 4 (2008) 243-254

[Mislove et al., 07] Mislove, A., Marcon, M., Gummadi, K.P., Druschel, P., Bhattacharjee, B.: Measurement and analysis of online social networks. In: Proc. ACM SIGCOMM Conf. on Internet Measurement. (IMC'07) (2007) 29-42

[Monares et al., 11] Monares, A., Ochoa, S.F., Pino, J.A., Herskovic, V., Rodriguez-Covili, J., Neyem, A.: Mobile computing in urban emergency situations: Improving the support to firefighters in the field. Expert Systems with Applications 38, 2 (2011) 1255-1267

[Ochoa et al., 11] Ochoa, S., Bravo, G., Pino, J., Rodrguez-Covili, J.: Coordinating loosely-coupled work in construction inspection activities. Group Decision and Negotiation 20, 1 (2011) 39-56

[Piatek et al., 07] Piatek, M., Isdal, T., Anderson, T.E., Krishnamurthy, A., Venkataramani, A.: Do incentives build robustness in BitTorrent? In: Proc. Symposium on Networked Systems Design and Implementation (NSDI'07) (2007)

[Poundstone, 92] Poundstone, W.: Prisoner's dilemma. Anchor Books (1992)

[Rahman et al., 10] Rahman, R., Meulpolder, M., Hales, D., Pouwelse, J., Epema, D., Sips, H.: Improving efficiency and fairness in P2P systems with effort-based incentives. In: Proc. IEEE Int. Conf. on Communications (ICC'10) (2010) 1-5

[Rand et al., 09] Rand, D.G., Dreber, A., Ellingsen, T., Fudenberg, D., Nowak, M.A.: Positive interactions promote public cooperation. Science 325, 5945 (2009) 1272-1275

[Santos et al., 13] Santos, A., Anta, A.F., Fernández, L.L.: Quid Pro Quo: A mechanism for fair collaboration in networked systems. PloS one 8, 9 (2013) e66575

[Sarmenta and Hirano, 99] Sarmenta, L.F.G., Hirano, S.: Bayanihan: Building and studying web-based volunteer computing systems using java. Future Generation Computer Systems 15, 5-6 (1999) 675-686

[Seung et al., 05] Seung, J. and Mustaque, A.: Incentives in BitTorrent induce free riding. In: Proc. of ACM SIGCOMM Workshop on Economics of peer-to-peer systems (P2PECON'05) (2005) 116-121

[Sirivianos et al., 07] Sirivianos, M., Park, J.H., Chen, R., Yang, X.: Free-riding in BitTorrent networks with the large view exploit. In: Proc. Int. Workshop on Peer-To-Peer Systems (IPTPS'07) (2007)

[Vega et al., 11] Vega, D., Medina, E., Messeguer, R., Royo, D., Freitag, F., Ochoa, S.F., Pino, J.A.: Characterizing the effects of sharing hardware resources in mobile collaboration scenarios. In: Proc. of Int. Conf. on Computer Supported Cooperative Work in Design. (CSCWD'11) (2011) 465-472

[Vega et al., 13a] Vega, D., Messeguer, R., Freitag, F., Ochoa, S.F.: Effort-based incentives for resource sharing in collaborative volunteer applications. In: Proc. of Int. Conf. on Computer Supported Cooperative Work in Design (CSCWD'13) (2013) 37-42

[Vega et al., 13b] Vega, D., Messeguer, R., Ochoa, S.F., Pino, J.A., Freitag, F., Medina, E., Royo, D.: Sharing hardware resources in heterogeneous computer-supported collaboration scenarios. Integrated Computer-Aided Engineering 20, 1 (2013) 59-77

[Willer, 99] Willer, David. Network exchange theory. Westport, CT: Praeger (1999)

[Zhang et al., 10] Zhang, C., Dhungel, P., Wu, D., Liu, Z., Ross, K.W.: BitTorrent darknets. In: Proc. of IEEE INFOCOM'10 (2010) 1460-1468