Int. J. Communications, Network and System Sciences, 2014, 7, 303-315
Published Online August 2014 in SciRes. http://www.scirp.org/journal/ijcns
http://dx.doi.org/10.4236/ijcns.2014.78033

Scientific Research

Self Organizing Communities: Enhancing Cooperation through Competition

Aristotelis Glentis, Damianos Kypriadis, Nancy Alonistioti

National and Kapodistrian University of Athens, Athens, Greece
Email: arisg@di.uoa.gr, damianosk@di.uoa.gr, nancy@di.uoa.gr

Received 23 June 2014; revised 23 July 2014; accepted 10 August 2014

Abstract

One of the open issues in today's networking world is efficient content retrieval, storage, and provision. Numerous studies have been carried out in order to optimize the distribution and provision of content to end users. This is emphasized in today's mobile networks that the ubiquity of network resources is considered an expected fact. In this paper we propose an architecture that is based on node co-operation that enhances the content distribution and retrieval. The architecture consists of the formation of loosely coupled and autonomous communities among nodes that share common interests where the already existing content is shared in a "common pool". The co-operation is enforced by the acquisition of extra bandwidth that is distributed to the most helpful members. The autonomic aspect of the communities is based on the embedded mechanisms for cluster behavior, namely, voting, where the nodes decide themselves which members should be awarded, and reporting, where misbehaving nodes are removed from the community. In this paper the different elements of the architecture will be described, together with a theoretic study of its viability, and finally a set of simulations that evaluate its performance.

Keywords

Mobile Resource Management, Community Building, Node Co-Operation, Game Theory

1. Introduction

Traditionally, node co-operation has been one of the most useful tools for problem solving in a number of different cases, ranging from scientific purposes, where distributed algorithms have been created to solve domain specific problems (e.g., searching [1]), to more mundane reasons, or even malware purposes, such as denial of service attacks. Whenever the requirements of a problem surpass the abilities of a single node, a horizontal scaling (using more nodes) approach has been used together with a vertical scaling one (using more powerful ma-

How to cite this paper: Glentis, A., Kypriadis, D. and Alonistioti, N. (2014) Self Organizing Communities: Enhancing Cooperation through Competition. *Int. J. Communications*, *Network and System Sciences*, **7**, 303-315.
http://dx.doi.org/10.4236/ijcns.2014.78033

chines, or larger storage). Notwithstanding the above, in the specification of future networks and mobile communications, *i.e*., Long Term Evolution System Architecture (LTE SAE) [2], one of the key features is the Self-organization of network nodes, terminals and network clusters. This concept has been originated from the Autonomic Communications features [3]. Thus, the dynamic formation of clusters of nodes pursuing a common optimization target is adhering to a new approach that can have applicability in a series of problems.

One of the open issues in today's networking world is efficient content retrieval, storage, and provision. Judging from the popularity of a number of relevant sites, such as YouTube and torrent trackers, user-acquired or even user-generated content has increased dramatically, and its quality has been improved. This causes higher demand for storage, bandwidth, and speed of downloading.

The phenomenon appears even more intensely in wireless mobile networks where traffic generated from mobile devices grows rapidly [4]. The advent of terminals such as laptops, smartphones and tablets has changed the way users communicate and share content. A shift to the usage of wireless networks reflects the reality and according to predictions the traffic being generated from wireless mobile terminals will exceed that of wired ones [5]. Furthermore, mobile networks that permit their users to be free of physical connections are frailer to the flash crowd effect, where the sudden increase of users in the area and high network demand can cause the deterioration of network services, creating high load to the network gateway.

Both of the two predominant Internet application architectures (*i.e*., client-server and peer-to-peer (p2p)) essentially assume that nodes operate in isolation, even when close proximity exists. There are many cases where one user wants to retrieve or share information or content (even in real time) with other nearby users, but the communication has to take place over centralized services through the Internet. Example of such cases includes the sharing of a video through YouTube, and location based sites such as foursquare. This type of communication results in a lot of duplication of effort for content retrieval, leading to network congestion, higher latency and link saturation in network gateways, which increase with the number of users. Therefore, our focus is on decentralized architectures, while also exploiting the proximity that exists within the network nodes.

Previous work on node co-operation can be found on number of publications [6]-[11] that prove, by using a game theoretic approach, node co-operation is feasible. They also introduce the use of strategies based on reward-penalty systems, or reputation systems. Furthermore, content delivery networks have been studied extensively [12]-[15] in order to provide maximization and efficiency, using either centralized or p2p architectures, for content retrieval and distribution. Most of the previous work is based on nodes that behave altruistically, or they follow a well-defined protocol that enforces co-operation, making co-operation the best strategy. In contrast, our work enhances co-operation using competition, making it the best strategy. Increasing the competition among nodes for the shared resources, the deviation from the co-operation strategy is made unprofitable due to the penalties which will occur from the other nodes. In this regard, increasing competition magnifies the co-operation of nodes; while each node is trying to maximize his personal gain, also the total gain from the network point of view in incremented.

2. Methods

The central idea of our approach is the formation and management of autonomous self-organizing loosely coupled communities between network users in close proximity that share resources and content. Besides the content that the members exchange, each member offers a percentage of his memory to the community, so that a global cache is created. In this cache, members can store and retrieve temporary data, according to their needs. This resource sharing schema can also be extended to include a number of different resources, but for content sharing and distribution the focus is given to memory. The communities are self-governed and two different mechanisms are used to help the formation and regulate the workings within them, namely the voting, and the reporting mechanism. These mechanisms ensure the fairness and equality of the community operation; where equality means the members are treated as equals with regards to the community decisions, and fairness is based on the notion that each member can receive benefits proportionally to what it offers backs to the common pool. Finally the communities created are loosely coupled because the nodes still operate in isolation regarding content requests, co-operating only for already existing content within the community. This is in contrast to highly coupled communities where the content retrieval can be orchestrated by a central entity.

In this section a general overview of the community entities, operation and mechanisms will be provided, giving the key points on the main issues without delving in much detail. Throughout this paper the following assumptions hold true:

- The nodes are selfish but rational. They try to maximize their personal gain by consuming the community resources, but they will not behave in a way that is against their interests.
- The nodes can be uniquely identified in the communication both with the network gateway and among them.

2.1. Community Entities

In the proposed architecture which is depicted in **Figure 1** three different entities affect and participate the formation of the communities: the network gateway, the community leaders and the participating nodes. Although our architecture is not specific or limited to mobile nodes, in its evaluation a model of mobile network was used. The reason is that mobile networks pose bigger challenges to tackle due to the hardware limitations, and node mobility. However, in the following presentation and discussion the architecture can be used interchangeably for wired or wireless nodes.

The network gateway's role is to distribute the bandwidth among the clients with regard to the policies that collects from the community leaders. It does not participate actively in the inner procedures of the communities, but it is the element that enforces the policy described by the community leaders, and also creates the ecosystem that small and large communities can grow and sustain. The network gateway reserves a percentage of the total external bandwidth that is allocated to the community leaders for further distribution to their members. The community leaders on predefined time windows sent to the gateway a list of members and the extra bandwidth that they can consume, and the network gateway ensures the proper QoS for these members. The amount of bandwidth reserved for each community leader is proportional to the relative size (*i.e.* number of nodes) of each community. Small communities that are not able to reserve an adequate amount of bandwidth on certain time frames enter a "grace period" where a percentage of the bandwidth is reserved for their use. Furthermore, when a request for a new community is made, the new community is also granted a "grace period" so that it is able to recruit new members.

The community leader is responsible for advertising the community, control the community mechanisms and distribute the community allocated bandwidth to the prominent members. By transmitting lists to the network gateway with members of the community that are eligible to use part of the community bandwidth, is able to control the allocated bandwidth and reward members that have offered to the community, while also punish members that have been misbehaving. Since the community leader is responsible to control the bandwidth provided to it, it can also reserve a small percentage, according to its own needs, to be used for its own purposes. The community leader also advertises the existence of the community together with the admission requirements for joining the community to the nodes that are within its range. The advertisement includes the following information:

- Community ID is a freeform string that is used to uniquely identify the community.
- tag_i is a freeform string that is used to describe the semantics of a specific content. It is the same as the tags users use to describe content in different content based sites. Since this is a freeform string, the leader advertises the most popular tags that exist within the community.

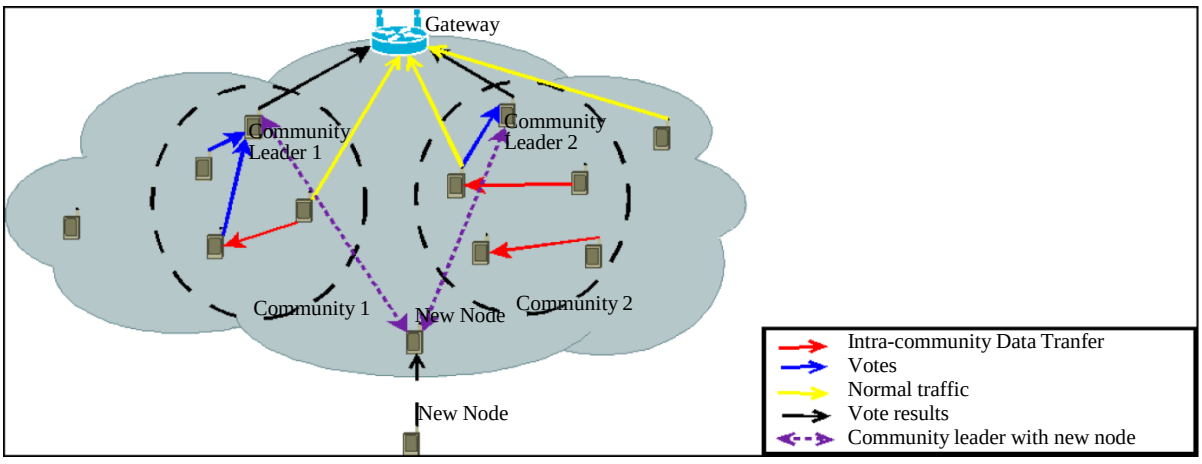

Figure 1. Architecture overview.

- C_{tag_i} is the size for the specific that the leader asks for new nodes to offer to the community.
- M_{adv} is the memory that the user is asked to offer to the community global cache.
- m is the number of the community members.

This information is broadcasted to the nodes within the range of the community leader. In case the communication occurs on a medium that does not provide broadcasting capabilities, or even the broadcast range of the community leader is inadequate, the network gateway can also assist, by transmitting the values of the different communities that exist within the region to the new joining nodes.

Finally, the community members try to maximize their personal gain by utilizing the community resources, and also acquire extra bandwidth if it is possible. Using the advertised values of the community leader for each community, the node can decide if it wants to participate. More precise observing the new node can find out what is the focus of the community and compare it with its own content or interests, while also with C_{tag_i} it can have an estimate of the size of content that exists for each tag. M_{adv} can be used as an estimate for the global cache that exists within the community, while with m it can learn the current size of the community. In general, the nodes can be classified in two major categories: The "content hungry", and the "bandwidth hungry". "Content hungry" can be considered the nodes that join a community aiming to acquire the content that exists within. There is a high correlation of interests between the tag_i of the community, and the node's interests, however, the node lacks such content "Bandwidth hungry", on the other hand, can be considered the nodes that have a lot of content on some (or all) of the tag_i values that are requested by the community, which they can offer to the rest of the members, permitting them to request more external bandwidth. This classification of the nodes is not something static. It depends on the different communities that exist in the region and also it can vary with time as user acquires content and provides content to other members and it can change its position and target within the community. As tedious as a process this might appear on first glance, using machine learning on the client side, a cluster of different words can be created by monitoring user requests. This classification can be used to create a user profile, which in turn can be used for filtering of communities, or even automatic joining based on user preferences.

2.2. Community Construction and Operation

A community can be constructed by a node that sends a request to the network gateway informing the formation of the new community and the community ID that will be used in order to identify the community. The node takes the role of the community leader for this community, and the new formed community enters a "grace period" where a percentage of the community allocated bandwidth is reserved for its need. The community leader then tries to recruit new members in order to increase the community's size, and request additional bandwidth from the network gateway. Periodically the community leader informs the network gateway about the size of the community, and also for the members that are eligible together with the percentage of the community bandwidth they can use.

Within the community, a node in search for content that is not in its local cache, broadcasts a message to the other members requesting for the specific content. If the content exists in the community, the nodes that have the content answer this request, and the nodes negotiate the parameters of the content transfer. The transfer takes place by transferring the data using the agreed method (either peer-to-peer like BitTorrent, or client-server like HTTP or FTP) and in chunks, which both are defined in the previous step. When a chunk is completed an acknowledge message is send from the receiver to the sender. This message is also sent to the community leader, which constitutes the vote for the sender. This message must be sent by at least the receiving node, but can also be sent by the sending node, so that the community leader can identify misbehaving nodes that take advantage of the community resources without voting for the helpful nodes. In case there are some issues in the transfer, or the sending node drops the transfer, the incident is reported to the community leader to take further actions if necessary. All the messages that are exchanged between the members are digitally signed and encrypted.

In case that the requested content does not exist in the community, the node gets the content from its original source. Then it stores it to the local cache and can share it with the rest of the community, or it can keep it for private use. Finally, especially "bandwidth hungry" members of the community can advertise the content they have to their neighbors in order to initiate more transfers and collect more votes from the described procedure.

2.3. Community Mechanisms

The two mechanisms that govern and ensure fairness in the community is the voting mechanism and the reporting

mechanism. The reason for two different mechanisms is that a well behaved and helpful node will be rewarded after a time of offering the services, while nodes that try to circumvent the stability of the community should be punished as soon as possible.

Voting takes place in rounds, where in each round the members vote for other members that have shared content or resources with them. Each vote consists of the size of the data transferred between the nodes, and the node identification. The votes are collected by the community leader. The community leader sorts the members according to the number of votes that they have collected, and distributes the community bandwidth to these members. Furthermore, the community leader might want to utilise a history schema for the votes, so that the votes a node has collected in the past are not lost on each round, but a percentage is kept.

The reporting mechanism is used to punish nodes that are faking information, or they try to take advantage of the community. This mechanism is operated by nodes that they report misbehaving peers. The misbehavior can be classified in two broad categories: either a node distributes fake or inappropriate content, or a destination node does not conform to the voting procedure and does not provide the votes to the helpful node. The community leader collects the reports from the nodes and decides how to act upon them. If evidence can be provided for the misbehaving node, the punishment can be carried out immediately, however if the community leader cannot verify the reporting reason, or the evidence provided, it can base the decision on the number of reports sent for a specific node. If the number of reports for a certain node reaches a certain threshold, the offending node should be punished. The punishment can range from the expelling from the community, to the request to the network gateway to provide a worse than best effort QoS to the offending node.

3. Results and Discussion

Following the previous presentation of the community architecture several questions rise about the benefits and viability of this approach.

- Which are the benefits of community formation judged on a network level? Is there a reduction on the external traffic that is achieved by the intracommunity content sharing?
- Are the nodes willing to join the community and share their resources with the other members? Does the presence of different communities in a region affect the size and internal cooperation of the nodes?
- What can happen if the community leaders are misbehaving themselves? Can the nodes be protected in this case?

All these questions are studied in this section, where a theoretic approach is combined with simulation results. The results are presented, together with an inline discussion analyzing them.

3.1. Traffic Reduction

In order to estimate the expected traffic reduction that incurs because of the community formation, two networks of nodes (one with ten and one with twenty) were simulated. In each network, two different cases where taken into account, if the nodes where operating in isolation, or whether they have formed a community. In the simulations without the community, each node randomly requests a chunk of a randomly chosen file. If this chunk does not already exist in the node's local cache, then the request is forwarded to the network gateway and it is serviced there. In the simulations with the community cache, the node initially checks its local cache for the file chunk, if it is not found, it checks the community cache. If it exists there it downloads it from the community, else it sends the request to the network gateway. In the last case, after the chunk is downloaded from the network it is added to the community cache, so that future requests are serviced within the community.

The file pool for the simulations consisted of 1000 files, each one segmented into 100 chunks. The file selection for each simulation was done using two alternatives: 1) random choice (*i.e.*, choose a file with equal probability), and 2) a zipf distribution for the file requests [16]. The reason for running the simulations for two different selection algorithms was to better evaluate the community mechanism. The random file selection was carried in order to "stress-test" the community under a "worst-case" scenario (having random file requests makes it harder for the cache to build and service, since the spatial locality is lost), while the zipf distribution simulated a more real-life scenario (some files have a higher request ratio than others). The following assumptions hold for the simulation:

- The number of nodes is constant during the simulation.
- At the beginning, the community cache is empty. This is a deliberate choice in order to put the network ga-

teway under strain for requests, since everything in the beginning must pass through the network gateway. In normal situations a number of these files would exist in the nodes, and a number of requests could be serviced within the community. The choice was also made in order to study a "worst-case" scenario.

- The nodes make constant requests to download file chunks. Under normal conditions, a node would make a number of requests, and then pause for a while. However since we do a "worst-case" scenario modeling, and the number of nodes is relatively small, we can assume that having more nodes with idle periods, can be modeled with fewer nodes but higher demand.

Since we start with an empty community content, and the requests are randomly made, we can assume that the accumulated content is distributed equally among the nodes. This avoids the case where large delays are introduced because a small set of nodes have all the content and attract all the requests for the content delivery Even in this case, because of the community caching from the different nodes, the content will be diffused to more members and the bottleneck will be lifted.

The simulation was done using Matlab Simulink for the event requests and servicing. The results are shown in **Figure 2** and **Figure 3**. In each figure the number of requests serviced by the network gateway is shown, for 20 nodes with and without the community, using a random distribution and a zipf distribution, respectively for the file requests.

Observe from the figures that the community approach attains significant traffic reduction through the network gateway. Without the community all the traffic has to pass through the network gateway causing higher bandwidth consumption and higher number of queued requests. Observe from **Figure 2** that with 20 nodes and using the random distribution we obtain a 20% reduction in the outgoing requests. Running the simulation for 10 nodes, almost the same amount of outgoing traffic reduction was achieved. Under the zipf distribution (**Figure 3**), the difference is much more impressive reaching a reduction of almost 400%. The reason is that the community cache was having many hits, and most of the requests were serviced within the community. This can be viewed as an increase to the active bandwidth that the community members can utilize. In **Figure 2**, there is a higher fluctuation of the number of requests, due to the random selection of the chunks and the files. The selection is space is quite large, and the selection is following a uniform random distribution creating a lot of jitter. On the other hand in **Figure 3**, the file selection that follows the zipf distribution creates a quite more predictable selection space, therefore reducing the jitter and the fluctuations.

3.2. Community Viability

In order to handle the viability questions posed in the beginning of this section, a theoretic model of the community building and operation is studied using game theory, and a set of simulation is provided based on this study. In the theoretic model a set of utility functions is proposed, which are used in the simulation to further evaluate their suitability.

3.2.1. Node Recruiting

The first game described is the joining of a new node to a set of m existing communities. The node learns by the community advertisement the admission values of each community, and communicates back to the community leaders it's corresponding offer values.

More formally, the game can be defined as:

$$G = \langle P, S, u \rangle$$

where $P = \{P_1, P_2, \cdots, P_k, P_N\}$ is the set of the players; P_1 to P_k are the community leaders and P_N the node that wishes to join one of the communities. We assume that the node is able to join k different communities, and is able to communicate with all the k community leaders.

Likewise, $S = \{S_1, S_2, \cdots, S_k, S_N\}$ is the set of the strategies that each player can follow:

$S_i = \{0,1\}$, $i = \{1, \cdots, k\}$ the strategy for the i-th community leader, where 0 is to refuse the joining of the new member and 1 is to accept it.

$S_N = \{0,1,\cdots,k\}$ the strategy for the node, where 0 is choosing not to join any community and 1 to k is choosing to join the k-th one.

Finally, $u = \{u_1, u_2, \cdots, u_k, u_N\}$ is the set for the utility functions for the community leaders and the node respectively.

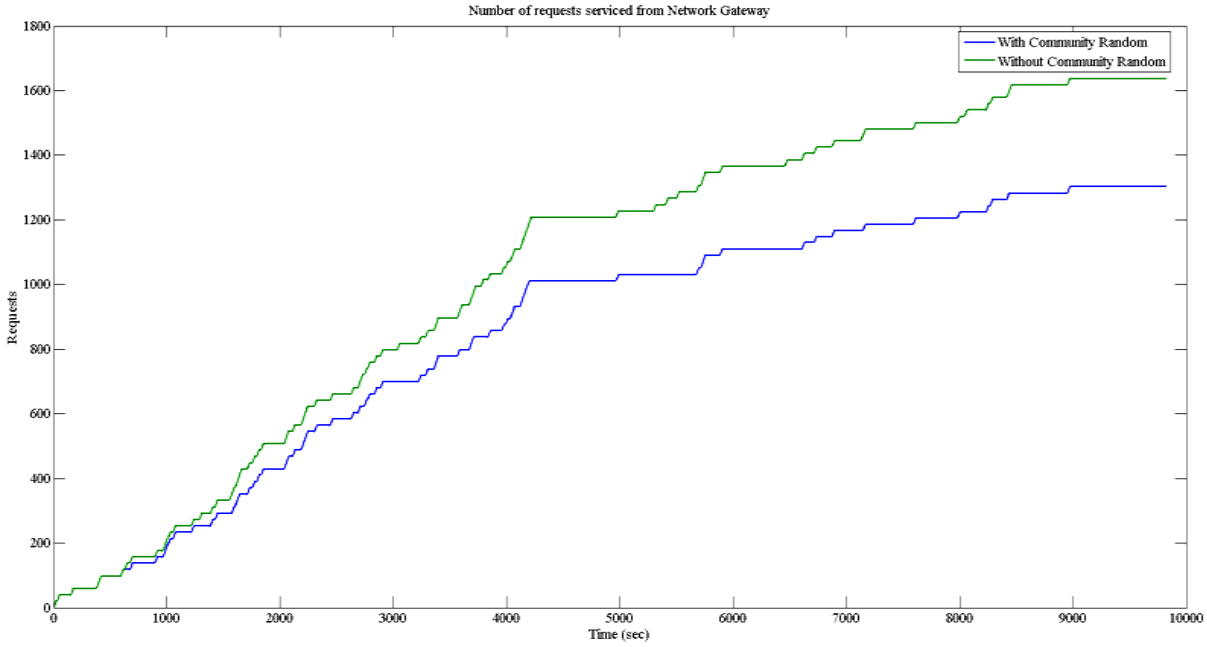

Figure 2. Number of requests through gateway with and without the community (20 nodes, random distribution).

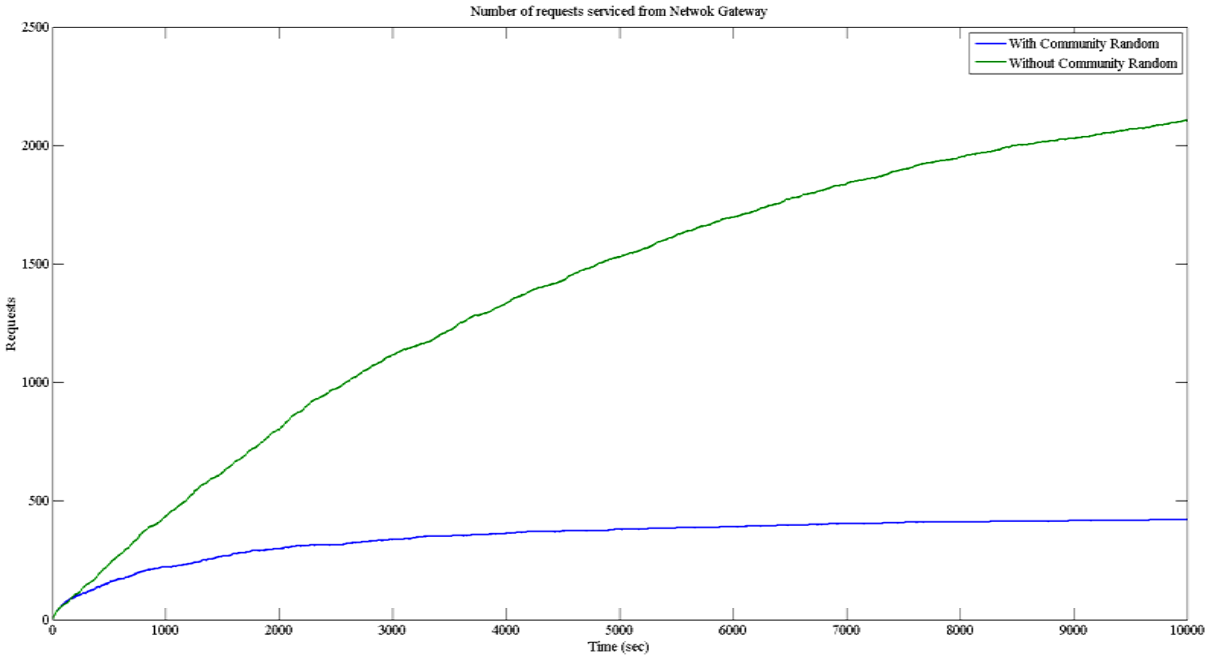

Figure 3. Number of requests through gateway with and without the community (20 nodes, zipf distribution).

In this section the utility functions that are used in our model are defined. The choice of this utility functions where made by trying to keep a balance among the estimation and ease of calculation.

Utility functions The utility function for each community leader can be defined as:

$$u_i = G_{\text{com}} - L_{\text{com}}$$

where G_{com} is the gain the community leader anticipates from the new node and L_{com} is the expected loss.

G_{com} is a metric of how much the node offers to the community plus the gain the community can have from the increased size. Therefore G_{com} can be further analyzed to:

$$G_{\text{com}} = G_{\text{content}} + G_{\text{memory}} + G_{\text{bw}}$$

where G_{memory} is the community gain from the memory offered by the node to the community and can be estimated as

$$G_{\text{memory}} = M_{\text{off}} / M_{\text{adv}}$$

where M_{off} is the size of the node's offered memory that can be used for global community cache, and M_{adv} the corresponding advertised memory.

G_{content} is the gain for the community from the extra content that the node offers. It can be estimated by summing for each tag the content offered by the node divided by the advertised content for this tag.

$$G_{\text{content}} = \sum_{tag_i} C_{\text{off}_i} / C_{\text{adv}_i}$$

Finally, G_{bw} can be estimated as:

$$G_{\text{bw}} = \left(m+1/n+1\right)/\left(m/n\right)$$

where m in the size of the community, and n is the size of all the devices in the region. This estimate is based on the increase of the relative size of the community by the inclusion of the new node.

On the other hand, L_{com} is a metric of the penalty that the community suffers in case a node is admitted which is trying to take advantage of the community resources and not offering back. L_{com} can be analyzed as:

$$L_{\text{com}} = L_{\text{content}} + L_{\text{memory}}$$

where

$$L_{\text{content}} = 1/G_{\text{content}}$$

and

$$L_{\text{memory}} = 1/G_{\text{memory}}$$

The community model is based on the intuition that each community leader is aiming to recruit members that either are willing to offer back to the community, or with the increase of size of the community, the community can acquire a better bandwidth reservation from the network gateway. The nodes that are not offering back much to the community are considered as "leech nodes" and their lack of offering is the loss the community will suffer in case they are permitted to join.

The corresponding utility functions can be derived for the node. In this case a distinction is made between the "bandwidth hungry" and the "content hungry" cases.

"Content hungry"

In this case the node is interested in requesting content from one or more tag_i types. Therefore, the node would choose a community that has more of the content it is seeking, aiming to use the intracommunity communication to download content from the other members. If the node has more content that the advertised values, then probably he will be a sharing node, instead of a sink node as its original target. Following the previous analysis and taking into account the "content hungry" node aims, the utility function is modeled as:

$$u_N = G_N - L_N$$

where

$$G_N = G_{\text{content}} + G_{\text{memory}}$$

$$G_{\text{content}} = \sum_{tag_i} C_{\text{adv}_i} / C_{\text{off}_i}$$

and

$$G_{\text{memory}} = M_{\text{adv}} / M_{\text{off}}$$

The corresponding loss function is defined:

$$L_N = L_{\text{content}} + L_{\text{memory}}$$

where,

$$L_{\text{content}} = 1/G_{\text{content}}$$

$$L_{\text{memory}} = 1/G_{\text{memory}}$$

It should be noted for a node that is searching for specific content that itself has little to offer back, then the gain of joining a community that provides this content is quite high, and the loss will be quite small.

"Bandwidth hungry"

In this case the node is interested in acquiring more bandwidth from the network gateway, having therefore a better QoS. It's primary target are communities that the node has a relative high amount of content, which he has share with the other nodes, collecting votes and therefore requesting better QoS from the network gateway. According to the previous analysis in this case the utility function is defined as:

$$u_N = G_N - L_N$$

where

$$G_N = G_{\text{bandwidth}}$$

$G_{\text{bandwidth}}$ can be approximated by the surplus the node estimates to have comparing to the other nodes.

The node estimates as loss L_N the amount of bandwidth that the node will use in order to service the requests from the other members. The more bandwidth the node uses for servicing the other members, the less it will be available for its own needs. Therefore an estimation for the L_N is:

$$L_N = 1/G_{\text{bandwidth}}$$

If the node finds a community that the members are searching for content that he can provide, by joining the community he can expect to become a prominent member quite fast, and acquire the most out of the community bandwidth.

Game Matrix In order to better present the resulting game matrix, the community leaders $(P_1, P_2, \cdots, P_m)$ have been concatenated to one entity P_{CL}whereas the node keeps the P_N representation. The utility function u_{CL} is defined as: $u_{CL} = \max\{u_1, u_2, \cdots, u_m\}$ which easily represents the case where the node is accepted in any of the available communities, even if not the optimal one for him. The game matrix is in **Table 1**.

This game matrix represents the family of the following four cases:

1) u_{CL}, $u_N < 0$. No community regards the node as an eligible member to join, and the node itself believes that he is better off isolated. In this case the Nash equilibrium is that neither of them is willing to cooperate.

2) $u_{CL} > 0$, $u_N < 0$. At least one of the communities wishes the node to join, but the node decides that it is not beneficial. The Nash equilibrium in this case is for the community to accept the request but the node will not join.

3) $u_{CL} < 0$, $u_N > 0$. It is the same as in case ii but with the roles reversed.

4) u_{CL}, $u_N > 0$ In this case, both sides think it is beneficial to cooperate and the node enters the community. The node chooses the community that anticipates will provide him the highest utility function.

It is apparent from the previous analysis that cooperation is based in the mutual estimation that no party is at loss. Community leaders perform a screening process to new users, if they estimate that this can lead to the deterioration of the community. This ensures the uniformity of the community that further enhances fairness and avoids nodes who try to leech. One the other hand, the new node will avoid wasting resources by joining a community that does not offer a clear benefit for it.

3.2.2. Node Cooperation

The question that is left to be answered is whether the formation of communities is going to be stable; *i.e.*, whether the nodes participating will favor to cooperate instead of isolate, and whether the competition that is

Table 1. The game matrix.

	P_{CL}	P_N
P_{CL}	0, 0	0, u_N
P_N	u_{CL}, 0	u_{CL}, u_N

created among the community leaders can enhance the cooperative behavior. In order to model the community operation we introduce the satisfaction functions that give a measure of the relative belief of each participant on the community performance. We define for each community leader:

$$Sat_{cl} = m/n\,,$$

where m is the size of the community and n is the number of available nodes. If this number drops, the community is shrinking, and the community leader is unable to acquire the needed bandwidth from the network gateway. Respectively, for a "content hungry" node we define:

$$Sat_N = C_{\text{com}}/C_{\text{nocom}}$$

where G_{com} is the size of the content acquired from the community and G_{nocom} the size of the content required through the network gateway. Finally for a "bandwidth hungry" node the satisfaction function is

$$Sat_N = BW_{\text{extra}}/BW_{\text{com}}$$

where BW_{extra} is the extra bandwidth that was obtained through the voting procedure, and BW_{com} is the bandwidth used for completing the other member requests. In case any of the satisfaction functions drops below a node defined threshold, the node is leaving the community, either to join another community, or operating in isolated mode.

For each node in a community with m members and community leader CL, we define the game describing its behavior as the following:

$$G = \langle P, S, u \rangle$$

where $P = \{CL, N_1, \cdots, N_m\}$ is the player in each case, $S = \{S_{SL}, S_1, \cdots, S_M\}$ the available strategies for each node, where $S_i = \{0,1\}$ for node P_i assuming that 0 is the strategy that P_i chooses not to serve, and 1 the opposite, $u = \{u_{SL}, u_1, \cdots, u_M\}$ the set of the utility functions for each node that participates. The utility functions in this case are the aforedescribed satisfaction functions.

We assume that on first round each player that wishes to maximize his utility function will try to cooperate. Each player is operating in a competitive way and tries to maximize his utility function in order to maximize the bandwidth gain. The Nash Equilibrium for each node exists if he tries to cooperate and does not change his strategy. We assume that in round k player P_i chooses to deviate from his original strategy 1 and change it to strategy 0. u'_k is the utility function in round k with strategy 1 and u_k the utility function with strategy 0. We separate each of the three cases:

- If P_i is a "bandwidth hungry" node, then by deviating from the cooperative strategy, it does not serve the peer nodes, and it does not collect votes. Therefore, it is not able to acquire more bandwidth.
- If P_i is a "content hungry" node, then the non cooperative strategy is to restrain from voting the peer nodes that provided the content. In this case the node will be reported, and removed from the community by the community leader, losing the ability to obtain the content from the nearby nodes.
- Finally if P_i is the community leader, the non cooperative strategy is to be unfair to the members and do not distribute the bandwidth accordingly to their usefulness. In this case, the "bandwidth hungry" members will leave the community since their satisfaction function will drop dramatically, and the community will loose a lot of the content providing points. This will also result in the reduction of the "content hungry" members who will try to follow the content trail. Therefore, the community will shrink to a great extent. The existence of other communities in the area, will increase the turnover percentage, since the nodes have a higher chance to choose a different community that achieve their target.

It is easy to see that in each of these cases $u'_k > u_k$, so P_i has no intent from deviating from strategy 1 of cooperation, therefore enforcing a fair behavior, making the communities stable.

3.2.3. Simulations

In order to better evaluate the presented model, a set of simulations was carried. In the simulations we measured the number of members of two different communities. Each community had a predefined set of tag contents that was advertising and there was some overlap between the two communities. Some tags that were used by the nodes didn't exist at any of the communities. The simulation was carried out in rounds. In each round a number of nodes where created that were informed about the presence of the two communities and

the described game took place with the community leaders. The outcome for each node was that either becomes a member of one of the two communities, the one that presented the higher conceived gain for the node, or the node declined both offers. Furthermore, on the end of each round, a number of nodes was departing from the community because their satisfaction number was low. Finally, in order to take into account the node's churn, a Weibull distribution was used [17] at the end of each round, and a number of nodes were removed because of node mobility.

The following assumptions hold true for this simulation:

- The random values used for the simulation, except when mentioned otherwise, were created using a standard uniform distribution.
- Each node when created has a random value ranged from 100 to 1000 MB of content, distributed to different tags.
- Each node offers a random percentage with a mean of 0.2 of its content and memory to the community in order to be admitted.
- The creation and arrival of the new nodes was done using a Poisson distribution of with an λ value of 20 nodes per round.
- In each round, both "content hungry" and "bandwidth hungry" nodes where created, and their class was defined randomly.
- The advertised values by the community leaders are the average of the value of the content and memory offered by the community members.
- Each node can be part on one only community.

Two different sets of simulations where performed. In the first set whose results are depicted in **Figure 4** the community leaders are fair, and the number of dissatisfied nodes is low, whereas in the second set depicted in **Figure 5** the one community leader is fair and the other in unfair, resulting having a higher number of dissatisfied nodes.

It can be observed from **Figure 4** that the number of nodes is increasing on each of the two communities with a rate that is reducing. This is consistent with the previous analysis, since as the size of the community is getting larger, and the sizes of the two communities are comparable, it is harder for new nodes to be admitted. Furthermore, since the nodes in the simulation are mobile nodes, as the size increases the disconnection of the members also increases because of mobility issues. However, the size of both communities is larger than the number of nodes that stay isolated. On the other hand in **Figure 5** the existence of the unfair community leader, and the higher dissatisfied number of nodes, leads the second community in decomposition. At each round more nodes are leaving from the unfair community and finally the community size nullifies. The unfair community content decreases resulting to recruiting of the incoming nodes to the fair community. Although we didn't provide the option of the dissatisfied nodes to migrate on the other community, it is anticipated that if we permitted that, the size of the first community will further increase, because of the recruiting of the leaving nodes.

4. Conclusions

The architecture presented in this paper enables node cooperation on mobile networks for content sharing through the formation of loosely coupled communities. Using game theory the viability of this architecture was studied, and the foundations for the model development were built. A number of simulations where run in order to shed some light on the key concepts of the community formation, and evaluate the gains from the use of this architecture. The results show that by using communities, as a form of node organization, is beneficial both to the network infrastructure (the network gateway) and to the participating nodes. The network benefits from reduced bandwidth demand and better utilization of its resources, whereas the participating nodes enjoy faster downloads, and better QoS. Therefore the presented architecture can be used as means to optimize content retrieval by exploiting the close proximity among nodes which have similar interests. The cooperation is ensured and driven by the ability to acquire more external bandwidth or QoS by the network gateway, on return of sharing the stored content.

Future work can be carried in studying the formation of tightly coupled communities. In the architecture presented here, the members are loosely participating in the community. Each node is providing the content that already has or just acquired. However, each node works in isolation when trying to retrieve content. A

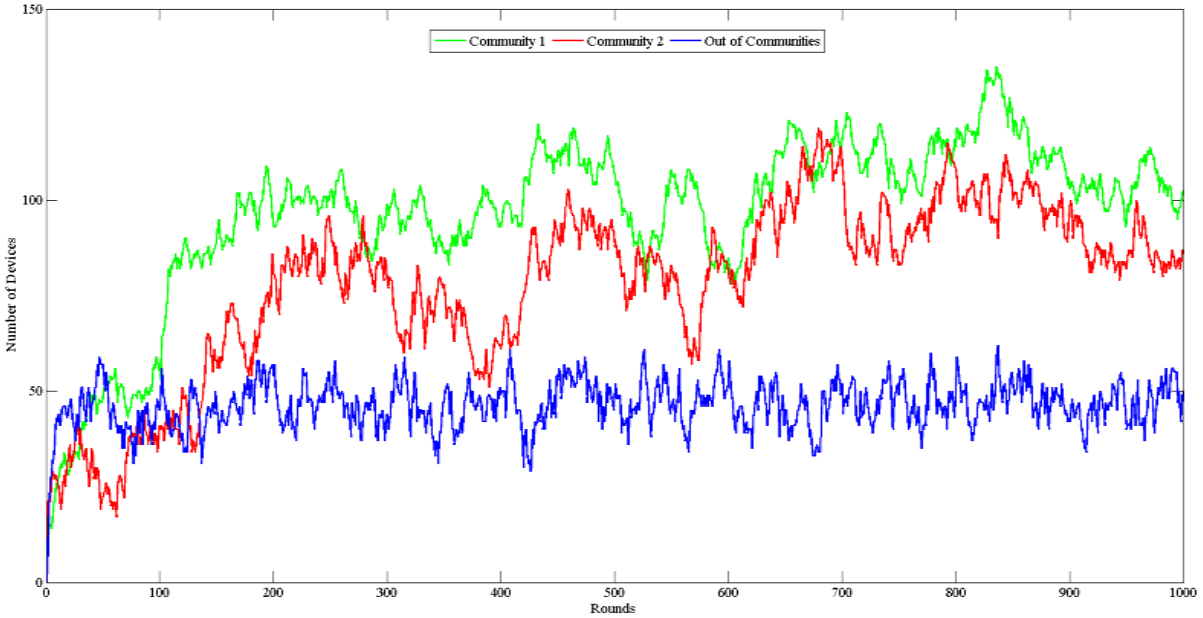

Figure 4. Number of nodes per round with fair community leaders.

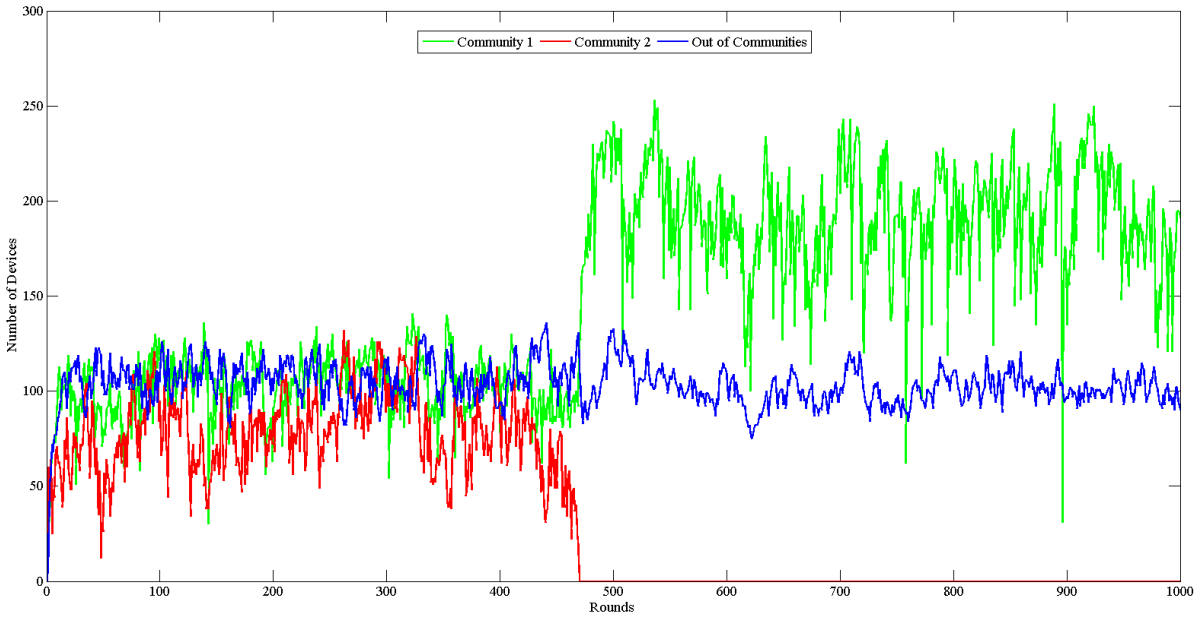

Figure 5. Number of nodes per round with fair and unfair community leaders.

more tightly coupled architecture can be explored, in which that nodes form a tight cooperation schema in order to acquire content from different sources and exchange downloaded chunks within the community, using more efficient communication channels. Content distribution can also take into consideration geolocation proximity and mobility among community members, and algorithms that make better utilization of the global cache can be used. Furthermore, new ways of distributing content retrieval can be investigated so that "smarter" ways of downloading (e.g., using swarm intelligence) can be utilized.

References

[1] Ahmed, R. and Boutaba, R. (2009) Plexus: A Scalable Peer-to-Peer Protocol Enabling Efficient Subset Search. *IEEE/ACM Transactions on Networking*, **17**, 130-143. http://dx.doi.org/10.1109/TNET.2008.2001466

[2] Kottkamp, M., Rössler, A., Schlienz, J. and Schütz, J. (2011) LTE Release 9 Technology Introduction Whitepaper.

[3] Dobson, S., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N. and Zambonelli, F. (2006) A Survey of Autonomic Communications. *ACM Transactions on Autonomous and Adaptive Systems*, **1**, 223-259.

[4] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.pdf

[5] http://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx

[6] Chun, B.-G., Chaudhuri, K., Wee, H., Barreno, M., Papadimitriou, C.H. and Kubiatowicz, J. (2004) Selfish Caching in Distributed Systems: A Game-Theoretic Analysis. *Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing*, New York, 21-30.

[7] Inaltekin, H. and Wicker, S. (2008) The Analysis of Nash Equilibria of the One-Shot Random-Access Game for Wireless Networks and the Behavior of Selfish Nodes. *IEEE/ACM Transactions on Networking*, **16**, 1094-1107. http://dx.doi.org/10.1109/TNET.2007.909668

[8] Chao, S.-L., Lin, G.-Y. and Wei, H.-Y. (2008) Mixed Altruistic and Selfish Users in Wireless Mesh Networks: A Game Theoretic Model for Multihop Bandwidth Sharing. *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, New York, 461-462.

[9] Milan, F., Jaramillo, J.J. and Srikant, R. (2006) Achieving Cooperation in Multihop Wireless Networks of Selfish Nodes, *Proceeding from the* 2006 *Workshop on Game Theory for Communications and Networks*, New York, 3. http://dx.doi.org/10.1145/1190195.1190197

[10] Hu, J. and Burmester, M. (2006) Lars: A Locally Aware Reputation System for Mobile Ad Hoc Networks. *Proceedings of the 44th Annual Southeast Regional Conference*, New York, 119-123. http://dx.doi.org/10.1145/1185448.1185475

[11] Hwang, J., Shin, A. and Yoon, H. (2008) Dynamic Reputation-Based Incentive Mechanism Considering Heterogeneous Networks. *Proceedings of the 3rd ACM Workshop on Performance Monitoring and Measurement of Heterogeneous Wireless and Wired Networks*, New York, 137-144. http://dx.doi.org/10.1145/1454630.1454651

[12] Molina, B., Pileggi, S.F., Esteve, M. and Palau, C.E. (2009) A Negotiation Framework for Content Distribution in Mobile Transient Networks. *Journal of Network and Computer Applications*, **32**, 1000-1011. http://dx.doi.org/10.1016/j.jnca.2009.03.007

[13] Eger, K. and Killat, U. (2008) Bandwidth Trading in Bittorrent-Like p2p Networks for Content Distribution. *Computer Communications*, **31**, 201-211. http://dx.doi.org/10.1016/j.comcom.2007.08.005

[14] Han, S.C. and Xia, Y. (2009) Optimal Node-Selection Algorithm for Parallel Download in Overlay Content-Distribution Networks. *Computer Networks*, **53**, 1480-1496. http://dx.doi.org/10.1016/j.comnet.2009.01.011

[15] Lloret, J., Garcia, M., Bri, D. and Diaz, J.R. (2009) Study and Performance of a Group-Based Content Delivery Network. *Journal of Network and Computer Applications*, **32**, 991-999. http://dx.doi.org/10.1016/j.jnca.2009.03.008

[16] Adar, E. and Huberman, B.A. (2000) Free Riding on Gnutella.

[17] Stutzbach, D. and Rejaie, R. (2006) Understanding Churn in Peer-to-Peer Networks. *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, New York, 189-202.

Scientific Research Publishing (SCIRP) is one of the largest Open Access journal publishers. It is currently publishing more than 200 open access, online, peer-reviewed journals covering a wide range of academic disciplines. SCIRP serves the worldwide academic communities and contributes to the progress and application of science with its publication.

Other selected journals from SCIRP are listed as below. Submit your manuscript to us via either submit@scirp.org or Online Submission Portal.