An Incentive-based Resource Allocation Protocol For Wireless Peer-to-Peer(P2P) Network

Di Chen
Electronic Engineering, SJTU

Abstract

Share is good for peer-to-peer(P2P) system performance, while users of current P2P networks tend not to share since there is no appropriate incentive mechanism. This lead to the "free-riding" and the "tragedy of commons" problems. This paper proposes a incentive mechanism in wireless P2P network to encourage users to share resources. This mechanism satisfies three properties:(1)conservation of cumulative contribution and social utility in the P2P system, (2)maximization of social utility if all requesting nodes have the same contribution value, and (3)incentive-based resource allocation. We introduce two algorithms in this paper, Contribution-based Allocation algorithm and Contribution Update algorithm. We also prove that combined with the two algorithm, our protocol can achieve the three properties. Simulation results illustrate the efficiency and the fairness of our protocol.

I. Introduction

In recent years the decentralized and structured or unstructured peer-to-peer(P2P) network has been one of the most potential solutions for efficient information exchange in the Internet. The P2P network holds many advantages such as quick query transmitting rate[11], locating objects in $\log n$ time, where n is the number of nodes in the network[5]. Despite its great potential for developing, there are some remaining problems that limit its development. It is acknowledged that *free-riding* and the *tragedy of the commons* are the two major problems in P2P network. For example, nearly 70% of Gnutella users do not share any file with others in a P2P community and nearly 50% of all search responses come from the top 1% of content sharing nodes[10]. This can cause congestion in P2P network and lead to the tragedy of the commons. Another problem is that many users intentionally misrepresent their connection speeds so as to discourage others from going to their nodes for file download[5]. Worse yet, Gnutella-like systems give no service differentiation between users who do not share any information with or make any contribution to the P2P community.

To offer a better service in P2P networks, there exist some mechanisms nowadays. Some P2P system use the first-come-first-served(FCFS) policy to offer file transfer services. This strategy is not good enough because it can cause large response time. Another naive mechanism is to evenly distribute the transfer bandwidth among the users. It works, but is not suitable. First, it does not maximize the efficiency of bandwidth utilization because some nodes will receive more resource than it originally need while other nodes receive less, which causes waste of bandwidth. Secondly, it is not fair because the nodes that contribute more receive the same amount of resource with the nodes that contribute less. Because of the shortcomings of current resource allocation mechanism in P2P network, this paper aims at designing and analyzing a protocol that provides incentives for users to increase the service quality in P2P network and optimize the performance of the whole network. The proposed issues are shown below:

(1) How to allocate bandwidth resource efficiently?
(2) How to allocate bandwidth resource fairly?
(3) How to increase the satisfaction of most nodes?
(4) How to increase the security level so that the nodes cannot misrepresent their contributions and required bandwidth?

To achieve these goals, this paper intends to propose a protocol to provide service differentiation based on the contribution level and the utility of individual node. It also provide an algorithm for computing the contribution values for all participating nodes. At the meantime, by analyzing various of malicious behaviors by users, we offer some mechanisms to avoid these actions and enhance the network security.

II. Related work

In this section, we briefly talk about some related work. In [3], the authors address one possible mechanism for Napster-liked P2P network to optimize the performance of the network. On the other hand, this paper aims at another mechanism which is different from [3]. In [1], the concept of *reputation system* is discussed for the application of P2P system and ad-hoc networks. [9] discusses the economic behavior of P2P storage networks. A

Game Theory approach is discussed in [6] to avoid the phenomenon that some nodes report bidding values larger than its maximum receiving bandwidth. Our work has something to do with it and the problem of security level is among the topics we discussed in the paper.

III. Incentive based P2P network

First, we present a famous P2P model, Gnutella, which is non-incentive based, and talk about its shortcomings due to the characteristic in this section. Then we propose a mechanism considering incentives in peers for allocating bandwidth resources to solve some problems exist in some kinds of P2P networks, especially for Gnutella-like network.

1. Gnutella protocols and problems

In Gnutella, each node plays the role of both a server and a client. Gnutella protocol defines five kinds of data packets. They are:

(1) **Ping Message:** Used to find devices in the network by a host, means "Are you there".

(2) **Pong Message:** Used to reply Ping Message, means "I'm here". Usually the Pong Message contains the information of IP address, port, the number of files shared and the total size of those files of a peer.

(3) **Query Message:** Used to find files needed by a peer, means "I'm looking for x". It is uniquely identified and its source is unknown.

(4) **Query Hit:** Used to reply to a query message, including the information necessary to download the file. It also contains a a unique client ID associated with the replying peer. These messages are propagated backwards along the path that the query message originally took.

(5) **Get/Push Message:** Get Message is a simple request for a file returned by a query. If a host locates behind a firewall, then the Push Message can request the server to initiate the connection to the requesting peer and upload the file. The connection will not be established if both the server and the client are behind a firewall.

Each message in Gnutella network is flagged with a time-to-live(TTL) field. Each hop of a message causes its TTL decreased by 1. When the TTL decreased to 0, then the message will not be flooded any more, which can save the resources and increase the efficiency of the whole network.

To formally describe a Gnutella network, we have the following notations:

- N, the number of nodes in the system.
- $\lambda_{i,j}$, the average file transfer request rate from node i to node j.
- u_i, the maximal upload bandwidth (Mbps) of node i.
- d_i, the maximal download bandwidth (Mbps) of node i.

Typically, we have $u_i \leq d_i$. In Gnutella networks, we often face the situation that one node serve more than one node. Therefore, the clients connecting to the same server i have to share the upload bandwidth u_i and this is where the problems comes from. If u_i is less than the total download bandwidth required by the competing nodes, some nodes have to wait. The simplest way to solve the problem is to use the first-come-first-serve principles. But this can cause other nodes waiting for a long time. For example, if the active sessions are being occupied with large file transfers, requests in the waiting queue will experience a long waiting time, which decreases the efficiency of the network.

The other way to share the bandwidth resources is to equally allocate the bandwidth to all competing nodes. In this situation, every node can download file. But it also has some limitations as discussed before.

2. Properties of incentive networks

Incentive based networks have some common variables, such as connection type of nodes, utility functions, contributions and so on. In this paper, we give some necessary notations:

- $\Theta = (\theta_1, \theta_2, ..., \theta_N)$, the connection type of all nodes in P2P networks. In particularly, θ_i is the connection type of node i.
- $C_i(t)$, the cumulative contribution of node i at time t, where $C_i > 0$.
- $x_i(t)$, the bandwidth allocated to node i when i requests a file transfer.
- $U_i(\theta_i, x_i)$, the utility function of node i dependent of i's connection type and allocated bandwidth.
- RN, the set of all requesting nodes.

The incentive-based P2P network should satisfy the features below:

(1) The total contribution of all nodes in system at time t should equal to the aggregate utility of all nodes in the system. Which is formally presented as

$$\sum_{i=1}^{N} C_i(t) = \sum_{i=1}^{N} \int_0^t U_i(\theta_i, x_i(\tau))d\tau \quad \forall t > 0 \quad (1)$$

(2) For all request nodes, their total utility,

$$\sum_{i \in RN} U_i(\theta_i, x_i(t))$$

, should reach its maximum value, where RN means the set of requesting nodes. We define the function

$$f(\mathbf{x}, u_k) = \sum_{i \in RN} U_i(\theta_i, x_i) \quad (2)$$

where,

$$\mathbf{x} = \{x_1, x_2, ...x_{|RN|}\}$$

Our goal is to find the $\mathbf{x}$ that makes $f(\mathbf{x}, u_k)$ reach its maxima with limited upload bandwidth u_k.

(3) Higher contribution leads to higher utility. That means for any two nodes $i, j \in RN$ at time t

$$\frac{C_i(t)}{d_i} \geq \frac{C_j(t)}{d_j} \Rightarrow U_i(\theta_i, x_i(t)) \geq U_j(\theta_j, x_j(t)) \quad (3)$$

when $\sum_{i \in RN} d_i > u_k$, u_k is the upload bandwidth of the server, and

$$U_i(\theta_i, x_i(t)) = U_j(\theta_j, x_j(t)) \quad (4)$$

when $\sum_{i \in RN} d_i \leq u_k$.

3. Definition of utility function

First, the utility function should satisfy $\frac{\partial U_i(\theta_i, x_i)}{\partial x_i} \geq 0$, second, the function should be bounded. In fact, we use the *log* function to define our utility function.

$$U_i(\theta_i, x_i) = U_i(d_i, x_i) = \begin{cases} \log(\frac{x_i}{d_i} + 1) & x_i \leq d_i \\ \log 2 & x_i > d_i \end{cases} \quad (5)$$

where $i \in RN$.

IV. Incentive protocols

In this section, we introduce our protocol to allocate bandwidth resources to clients. Our protocol take the contribution of the competing nodes into account in order to achieve fairness. On the other hand, for better utilization of bandwidth resources, we also propose a protocol to maximize the total utility functions in the system. In this situation, we can allocate resources more efficiently.

1. Incentives

We start with the feature (3). We know that when the node k, the server, has enough upload bandwidth to satisfy all competing nodes, then the solution is $\mathbf{x} = (d_1, d_2, ..., d_{|RN|})$. Otherwise, the competition among the requesting nodes starts. By $\frac{C_i(t)}{d_i} \geq \frac{C_j(t)}{d_j} \Rightarrow U_i(\theta_i, x_i(t)) \geq U_j(\theta_j, x_j(t))$ and equation (5), we can have the equation

$$\frac{x_i(t) + d_i}{x_j(t) + d_j} = \left(\frac{C_i(t)}{C_j(t)}\right)^{\gamma} \quad \forall i, j \in RN \quad (6)$$

to satisfy feature (3), where γ is constant positive value. Note that $x_i(t) \leq d_i$ and $x_j(t) \leq d_j$. Here equation (6) is just a tendency of distributing resources, but usually not satisfied after a round of allocation.

2. Maximization of utilities

From the analysis before, we have the function

$$f(\mathbf{x}, u_k) = \sum_{i \in RN} U_i(x_i, \theta_i)$$

to satisfy feature (2), we should find a set $\mathbf{x} = (x_1, x_2, ..., x_{|RN|})$ that can maximize $f(\mathbf{x})$. The $f(\mathbf{x})$ can be translated to

$$g(\mathbf{x}, u_k) = \prod_{i \in RN} (x_i + d_i) \quad (7)$$

and $g(\mathbf{x}, u_k)$ should reach its maxima.

3. Protocol

Now we have two requirements of our protocols. One is the incentive requirement and the other is the utilities maximizing requirement. First we represent our contribution-based allocation (CA) algorithm below:

(1) Initiate the solution set $\mathbf{x} = (0, 0, ..., 0)$ and sort the set $\mathbf{R} = ((x_1 + d_1)/C_1^{\gamma}, (x_2 + d_2)/C_2^{\gamma}, ..., (x_{|RN|} + d_{|RN|})/C_{|RN|}^{\gamma})$ in order of ascending.
(2) Start allocating bandwidth to R[1], the node with the lowest $(x_i + d_i)/C_i^{\gamma}$.
(3) If the value $(x_i + d_i)/C_i^{\gamma}$ equals to the next node during the allocating process and $x_i < d_i$, then allocate bandwidth to the next node simultaneously.
(4) For any node being allocated, if $x_i == d_i$, stop allocating bandwidth to it.
(5) After upload bandwidth u_k being used up, terminate the whole process.

We can prove the algorithm achieves the two requirements. First we will show that it maximizes the utilities in the system and then we will show that it satisfy feature (3).

1. Proof: To maximize function (7) with limited u_k, we must distribute $(x_i + d_i)/C_i^{\gamma}$ as even as possible. Suppose $\mathbf{x} = (x_1, x_2, ...x_{|}RN|)$ is a solution of the CA algorithm. For $\forall i \in RN$, the node i has the value $R[i] = (x_i + d_i)/C_i^{\gamma}$. Let $x_i > 0$, (1)if $\frac{(x_i + d_i)}{C_i^{\gamma}} < \frac{(x_j + d_j)}{C_j^{\gamma}}$, it means that the allocation to node j has not been started because allocation is always started from the node with the least $\mathbf{R}$ value. So $x_j = 0$ and we cannot move any resource from node j to other node to even the $\mathbf{R}$ value. (2)If $\frac{(x_i + d_i)}{C_i^{\gamma}} > \frac{(x_j + d_j)}{C_j^{\gamma}}$, that means $x_j = d_j$ and the utility of node j reaches its maxima, $log2$. In summary, the CA algorithm guarantees the solution to reach its local maxima. Since $f(\mathbf{x}, u_k)$ is a convex function about $\mathbf{x}$, so the local maxima is the global maxima.

2. Proof: Assume that $\frac{C_i^{\gamma}}{d_i} \geq \frac{C_j^{\gamma}}{d_j}$. It means that

$$\frac{d_i}{C_i^{\gamma}} \leq \frac{d_j}{C_j^{\gamma}}$$

By the CA algorithm, after allocation, we have

$$\frac{(x_i + d_i)}{C_i^{\gamma}} \leq \frac{(x_j + d_j)}{C_j^{\gamma}} \quad (8)$$

if (8) strictly meets the less than condition, then $x_i = d_i$ and the utility of node i is maximum, $log2$. Clearly, we have

$$U_i(\theta_i, x_i) \geq U_j(\theta_j, x_j)$$

If (8) meets the equal condition, we have

$$\frac{x_i + d_i}{C_i^{\gamma}} = \frac{x_j + d_j}{C_j^{\gamma}} \Rightarrow \frac{x_i/d_i + 1}{C_i^{\gamma}/d_i} = \frac{x_j/d_j + 1}{C_j^{\gamma}/d_j}$$

Since

$$\frac{C_i^{\gamma}}{d_i} \geq \frac{C_j^{\gamma}}{d_j}$$

then

$$\frac{x_i}{d_i} + 1 \geq \frac{x_j}{d_j} + 1$$

which implies that $U_i(\theta_i, x_i) \geq U_j(\theta_j, x_j)$.

The pseudo code of CA algorithm is shown in Ap-

pendix.

V. Contribution update

In this section, we talk about how to evaluate the contribution of one certain node. The contribution should be updated after a round of file transfer and the basic updating principle is that the more you share, the more contribution you will have, and the more you download, the less contribution you will have.

1. Gain and pay

By intuition, it is clear for us that for the server k, it serves a set of nodes and should gain something, while for the requesting node i, it receives services from the server and should pay something. Here we give the definition of gain $G_k(u_k)$ and pay $P_i(x_i)$ mathematically.

$$G_k(u_k) = \max f(\mathbf{x}, u_k) = \max \sum_{i \in RN} U_i(\theta_i, x_i) \qquad (9)$$

$$P_i(x_i) = G_k(u_k) - U_i(\theta_i, x_i) - \max f(\mathbf{x} - \{x_i\}, u_k - x_i) \qquad (10)$$

So for node k, after the file transfer, its contribution C_k will increase by $G_k(u_k)$. For node i, its contribution C_i will decrease by $P_i(x_i)$.

2. Contribution update protocol

Now we introduce the contribution updating algorithm (CU).

(1) Let $\mathbf{x}$ be the solution of CA algorithm with the real contribution values and $\mathbf{y}$ be the solution of CA algorithm with all contribution values be 1. If $\mathbf{x} \neq \mathbf{y}$, go to step (2), else terminate the process.

(2) Find the maximum value $x_i - y_i$ of $\mathbf{x} - \mathbf{y}$. Calculate P_i and then decrease the contribution value of node i by P_i. Go to step (3).

(3) $u_k = u_k - x_i$, and let $RN = RN - \{x_i\}$, go to step (1).

The pseudo code of the algorithm is shown in Appendix. The contribution updating algorithm is based on a fluid model. We divide time into quanta denoted as Δt. At each beginning of Δt, we allocate bandwidth resources to requesting nodes and at the end of Δt, we update the contribution values. Next we will show that the CU algorithm satisfies the first property.

Proof: For simplicity, assume $C_i(0) = 0$, $\forall i \in RN$. Let $G_{k,i}(t)$ be the payment of node i for receiving bandwidth x_i from node k at time t and $G_k(t)$ be the gain of node k for providing its bandwidth at time t. Let $RN^\star \subseteq RN$ be the set consisting of all nodes requesting resources from node k and having positive payments at time t. Consider the time interval $[t, t + \Delta t)$. We have

$$\sum_{i \in RN \cup \{k\}} [C_i(t + \Delta t) - C_i(t)] = [G_k(t) - \sum_{i \in RN^*} G_{k,i}(t)] \Delta t$$

Let a_j be the j^{th} node that is required to pay. According to the payment rules, we have

$$\begin{aligned} G_{k,a_j}(t) &= G_k(u_k - \sum_{i=1}^{j-1} x_{a_i}(t)) \\ &- U_{a_j}(\theta_{a_j}, x_{a_j}(t)) \\ &- G_k(u_k - \sum_{i=1}^{j} x_{a_i}(t)) \end{aligned}$$

Summing for all the nodes in $RN^\star$, we have

$$\begin{aligned} \sum_{i \in RN^*} G_{k,i}(t) &= G_k(u_k) - \sum_{i \in RN^*} U_i(\theta_i, x_i(t)) \\ &- G_k(u_k - \sum_{i \in RN^*} x_i(t)) \end{aligned}$$

When the contribution update process finishes, a node, say i, who does not need to pay receives a transfer bandwidth of x_i under CA, which is equal to the bandwidth y_i. Therefore, we have

$$\begin{aligned} \int_0^t \sum_{i=1}^{N} \frac{dC_i(\tau)}{d\tau} d\tau &= \int_0^t \sum_{i=1}^{N} U_i(\theta_i, x_i(\tau)) d\tau \\ \sum_{i=1}^{N} C_i(t) &= \sum_{i=1}^{N} \int_0^t U_i(\theta_i, x_i(\tau)) d\tau \end{aligned}$$

VI. Simulation results

In this section, we present simulation results showing that our mechanism can fairly distribute transfer bandwidth among the requesting nodes.

Simulation 1:

Four nodes make requests to node 2 , which has a transfer resource of $u_k = 400$. The contribution values of these requesting nodes at time t_0 are $[C_1, C_2, C_3, C_4] = [1, 1.5, 2, 2.5]$. The connection types of all the requesting nodes are the same and their maximum download bandwidth are $d_1 = d_2 = d_3 = d_4 = 150$. Each simulation lasts 100 units of time in the interval of $[t_0 \leq t \leq t_0 + 100]$. Figure 1 illustrates the bandwidth assignments $x_i(t)$ and their respective contribution values $C_i(t)$ during the simulation period.

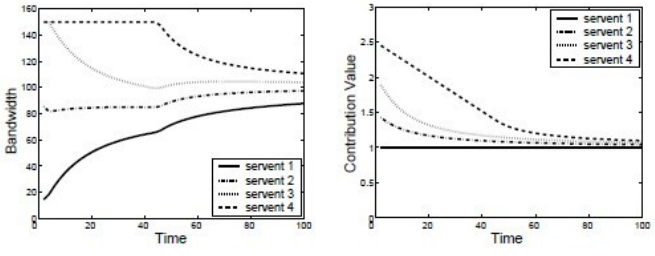

Figure 1: (a) Instantaneous bandwidth assignment and (b) instantaneous contribution values for competing nodes.

In Simulation 1, all nodes have the same requiring bandwidth, but by CA algorithm, the node with higher

contribution receives more resources. At the same time, our CU algorithm decrease their contributions as they are getting more resources. Eventually, all nodes tend to have the same contribution values and equal bandwidths, which maximize the aggregate utility in the system.

Simulation 2:

In this simulation, we just change the requesting bandwidth to $\mathbf{d} = [100, 150, 200, 250]$. All the other settings are the same with Simulation 1. The results are shown in Figure 2.

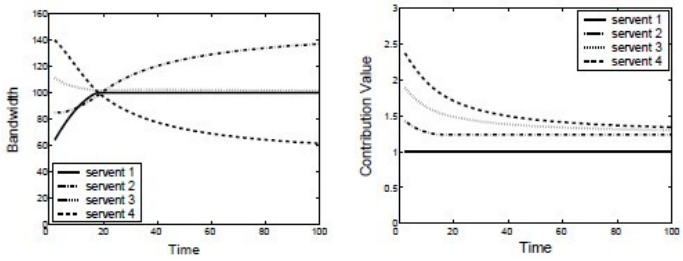

Figure 2: (a) Instantaneous bandwidth assignment and (b) instantaneous contribution values for competing nodes.

We can also find that all the contribution values tend to reach 1 and their instantaneous bandwidth converges to the solution of CA algorithm with all contributions be 1.

Limited by time and the resources we have, there is no way to compare the efficiency of our protocol with that of other resource distribution disciplines.

VII. Conclusion

We have proposed an incentive-based protocol for resource allocation in wireless P2P networks. This protocol is based on each node's utility function, connection type(or the requesting download bandwidth), and contribution. It can achieves both high aggregate utility and fairness. Since the larger the contribution a node has, the more resource it will receive and the only way to increase a node's contribution value is to serve for other nodes, every node in P2P system has incentive to share resources. The free-riding problem mentioned before can be resolved. Furthermore, our protocol may decrease the contribution values of nodes who access a congested resource. Therefore, it also provides incentive for nodes to access information from non-congested nodes and resolves the tragedy of the commons problem.

References

[1] Minaxi Gupta, Paul Judge, Mostafa Ammar, "A Reputation System for PeertoPeer Networks", *College of Computing, Georgia Institute of Technology.*

[2] Michal Feldman, Kelvin Lai, John Chuang, Ion Stoica, "Quantiying discentives in Peer-to-Peer netwokrs", *U. C. Berkly.*

[3] Philippe Golle, Kevin Leyton-Brown, Ilya Mironov, "Incentives for Sharing in Peer-to-Peer Networks", *Computer Science Department, Stanford University.*

[4] Torsten Ackemann, Cecilia Mascolo, Wolfgang Emmerich, "Lightweight Incentives for Peer-to-Peer Networks", *Proceedings of the 2001 ACM Conference on Electronic Commerce*, **2001**

[5] Richard T. B. Ma, Sam C. M. Lee, John C. S. Lui, David K. Y. Yau, "Incentive Resource Distribution in P2P Networks".

[6] Richard T. B. Ma, Sam C. M. Lee, John C. S. Lui, David K. Y. Yau, "Incentive and Service Differentiation in P2P Networks: A Game Theoretic Approach".

[7] Lian Jian, Jeffrey MacKie-Mason, "Why Share in Peer-to-Peer Networks?", **May 26, 2006**.

[8] Michal Feldman, Kevin Lai, Ion Stoica, John Chuang, "Robust Incentive Techniques for Peer-to-Peer Networks", *U. C. Berkly and HP Labs.*

[9] A. C. Fuqua, T. Ngan, and D. S.Wallach, "Economic Behavior of Peer-to- Peer Storage Networks", *Workshop on Economics of Peer-to-Peer Systems (Berkeley, California)*, **2003**.

[10] Eytan Adar, Bernardo A. Huberman, "Free Riding on Gnutella", *Technical report, Xerox PARC, 10 Aug. 2000. FirstMonday.*

[11] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications", *ACM SIGCOMM*, **2001**.

VIII. Appendix

Algorithm 1: CA

if $\sum_{i \in RN} d_i \leq u_k$ **then**
  // no congestion
  return $\mathbf{x} = \mathbf{d}$;
else
  // congestion
  $\mathbf{x}$=(0,0,...,0);
  $\mathbf{R}$=$((x_1 + d_1)/C_1^{\gamma}, (x_2 + d_2)/C_2^{\gamma}, ...,$
  $(x_{|RN|} + d_{|RN|})/C_{|RN|}^{\gamma})$;
  sort($\mathbf{R}$) **in ascending order**;
  i=1;
  level=$\mathbf{R}$[1];
  vol=$C_{\mathbf{R}[1]}^{\gamma}$;
  repeat
    i=i+1; // Initialize the index value
    nextLevel=R[i];
    if *(nextLevel-level)*vol*$\geq u_k$ **then**
      // u_k has been used up.
      level=level+u_k/vol;
      $u_k = 0$;
    else
      $u_k = u_k$−vol*(nextLevel-level);
      level=nextLevel;
      if $\mathbf{R}$**[i]**==*min(*$\mathbf{R}$*)* **then**
        vol=vol+$C_{\mathbf{R}[i]}^{\gamma}$;
      else
        vol=vol-$C_{\mathbf{R}[i]}^{\gamma}$;
  until $u_k \leq 0$;
  foreach $i \in RN$ **do**
    if *level*$> d_i/C_i^{\gamma}$ **then**
      x_i=[level-d_i/C_i^{γ}]*$(C_i)^{\gamma}$;
  return $\mathbf{x}$;

Algorithm 2: CU

$\mathbf{C} = (C_1, C_2, ...C_{|RN|})$;
$\mathbf{C}^* = (1, 1, ..., 1)$;
$\mathbf{x}$=$\mathbf{CA}(u_k$,RN,$\mathbf{C})$;
$\mathbf{y}$=$\mathbf{CA}(u_k$,RN,$\mathbf{C}^*)$;
// The node k increase its contribution by
// $G_k(u_k)$
$C_k = C_k + (\sum_{i \in RN} \log(\frac{y_i}{d_i} + 1)) * \Delta t$;
repeat
  q=argmax$\{x_i - y_i\}$;
  if $x_q - y_q > 0$ **then**
    P=$\sum_{i \in RN} \log(\frac{y_i}{d_i} + 1)$;
    RN=RN-$\{q\}$;
    $u_k = u_k - x_q$;
    $\mathbf{C}^* = \mathbf{C}^* - \{C_q^*\}$;
    $\mathbf{y}$=$\mathbf{CA}(u_k$,RN,$\mathbf{C}^*)$;
    // The node p decrease its
    // contribution by P_q
    P=P-log$(\frac{x_q}{d_q} + 1) + \sum_{i \in RN} \log(\frac{y_i}{d_i} + 1)$;
    $C_q = C_q - P * \Delta t$;
until $x_q - y_q \leq 0$;