Lu Yan

Performance Evaluation and Modeling of Peer-to-Peer Systems over Mobile Ad hoc Networks

Turku Centre for Computer Science

TUCS Technical Report
No 678, March 2005

Performance Evaluation and Modeling of Peer-to-Peer Systems over Mobile Ad hoc Networks

Lu Yan
Åbo Akademi University, Department of Computer Science

TUCS Technical Report
No 678, March 2005

Abstract

With the advance in mobile wireless communication technology and the increasing number of mobile users, peer-to-peer computing, in both academic research and industrial development, has recently begun to extend its scope to address problems relevant to mobile devices and wireless networks. This paper is a performance study of peer-to-peer systems over mobile ad hoc networks. We show that cross-layer approach performs better than separating the overlay from the access networks with the comparison of different settings for the peer-to-peer overlay and underlaying mobile ad hoc network. We then present a performance model which captures most facets of mobile peer-to-peer systems. We hope our results would potentially provide useful guidelines for mobile operators, value-added service providers and application developers to design and dimension mobile peer-to-peer systems.

Keywords: Mobile, Peer-to-Peer, Performance, Modeling, Evaluation

TUCS Laboratory
Distributed Systems Design Laboratory

1. Introduction

Peer-to-Peer (P2P) computing is a networking and distributed computing paradigm which allows the sharing of computing resources and services by direct, symmetric interaction between computers. With the advance in mobile wireless communication technology and the increasing number of mobile users, peer-to-peer computing, in both academic research and industrial development, has recently begun to extend its scope to address problems relevant to mobile devices and wireless networks.

Mobile Ad hoc Networks (MANET) and P2P systems share a lot of key characteristics: self-organization and decentralization, and both need to solve the same fundamental problem: connectivity. Although it seems natural and attractive to deploy P2P systems over MANET due to this common nature, the special characteristics of mobile environments and the diversity in wireless networks bring new challenges for research in P2P computing.

Currently, most P2P systems work on wired Internet, which depends on application layer connections among peers, forming an application layer overlay network. In MANET, overlay is also formed dynamically via connections among peers, but without requiring any wired infrastructure. So, the major differences between P2P and MANET in this paper are (a) P2P is generally referred to the *application layer*, but MANET is generally referred to the *network layer*, which is a lower layer concerning network access issues. Thus, the immediate result of this layer partition reflects the difference of the packet transmission methods between P2P and MANET: the P2P overlay is a unicast network with *virtual* broadcast consisting of numerous single unicast packets; while the MANET overlay always performs *physical* broadcasting. (b) Peers in P2P overlay are usually referred to *static* nodes though no priori knowledge of arriving and departing is assumed, but peers in MANET are usually referred to *mobile* nodes since connections are usually constrained by physical factors like limited battery energy, bandwidth, computing power, etc.

The above similarities and differences between P2P and MANET lead to an interesting but challenging research on P2P systems over MANET. In fact, this scenario seems feasible and promising, and possible applications include car-to-car communication in a field-range MANET, an e-campus system for mobile e-learning applications in a campus-range MANET on top of IEEE 802.11, and a small applet running on mobile phones or PDAs enabling mobile subscribers exchange music, ring tones and video clips via Bluetooth, etc.

This paper is a performance study of peer-to-peer systems over mobile ad hoc networks. In the following section we will review previous work on P2P and MANET. After comparing different settings for the peer-to-peer overlay and underlaying mobile ad hoc network, we show that cross-layer approach performs better than separating the overlay from the access networks in section 3. In section 4, we present a performance model which captures most facets of mobile peer-to-peer systems. In section 5, we apply our analytical model to practical network design problems and analyze some important QoS issues. Finally, section 6 concludes the paper.

2. Background and State-of-the-Art

Since both P2P and MANET are becoming popular only in recent years, the research on P2P systems over MANET is still in its early stage. The first documented system is Proem [1], which is a P2P platform for developing mobile P2P applications, but it seems to be a rough one and only IEEE 802.11b in ad hoc mode is supported. 7DS [2] is another primitive attempt to enable P2P resource sharing and information dissemination in mobile environments, but it is rather a P2P architecture proposal than a practical application. In a recent paper [3], Passive Distributed Indexing was proposed for such kind of systems to improve the search efficiency of P2P systems over MANET, and in ORION [4], a Broadcast over Broadcast routing protocol was proposed. The above works focus on either P2P architecture or routing schema design, but how efficient is the approach and what is the performance experienced by users are still in need of further investigation.

Previous work on performance study of P2P over MANET is mostly based on the simulative approach and no concrete analytical model is introduced. Performance issues of this kind of systems are first discussed [5] with experiment results. There is a survey of such kind of systems [6] but no further conclusions were derived. There are also some sophisticated experiments and discussions [7] on P2P communication in MANET. Recently, B. Bakos etc. with Nokia Research analyzed a Gnutella-style protocol query engine on mobile networks with different topologies [8], and T. Hossfeld etc. with Siemens Labs conducted a simulative performance evaluation of mobile P2P file-sharing [9]. However, all above works fall into practical experience report category and no performance models are proposed.

We believe that to understand the performance issues, rigorous analytical models are needed, which capture the relation between key system parameters and performance metrics. In the remaining sections we present our efforts on performance evaluation of mobile peer-to-peer systems, especially from users' point of view, e.g. what is the performance experience of a user in mobile P2P systems? We then present a performance model which captures most facets of mobile peer-to-peer systems. We hope our results would potentially provide useful guidelines to design and dimension mobile peer-to-peer systems.

3. Performance Evaluation of P2P over MANET

As stated before, we, in this paper, focus only on the performance of P2P systems over MANET from users' point of view since it makes greater impact on the design decisions of such kind of system for mobile operators, value-added service providers and application developers. Specifically, we want to answer the following questions: (1) How can we perform an efficient search in mobile P2P systems? (2) and what is the performance experience when many users try to retrieve data with parallel downloading scheme? (We leave the answer to the second question to section 4 and 5.) To answer the first question, the routing protocols and route discovery efficiency of different settings for the peer-to-peer overlay and underlaying mobile ad hoc network should be further investigated.

There are many routing protocols in P2P networks and MANET respectively. For instance, one can find a very substantial P2P routing scheme survey from HP Labs [10], and US Navy Research publish ongoing MANET routing schemes [11]; but all above schemes fall into two basic categories: broadcast-like and DHT-like. More specifically, most early P2P search algorithms, such as in Gnutella [12], Freenet [13] and Kazaa [14], are broadcast-like and some recent P2P searching, like in eMule [15] and BitTorrent [16], employs more or less some feathers of DHT. On the MANET side, most on-demand routing protocols, such as DSR [17] and AODV [18], are basically broadcast-like. Therefore, we here introduce different approaches to integrate these protocols in different ways according to categories.

3.1. Broadcast over Broadcast

A rudimental approach is to employ a broadcast-like P2P routing protocol at the application layer over a broadcast-like MANET routing protocol at the network layer. Intuitively, in these settings, every routing message broadcasting to the *virtual* neighbors at the application layer will result to a full broadcasting to the corresponding *physical* neighbors at the network layer.

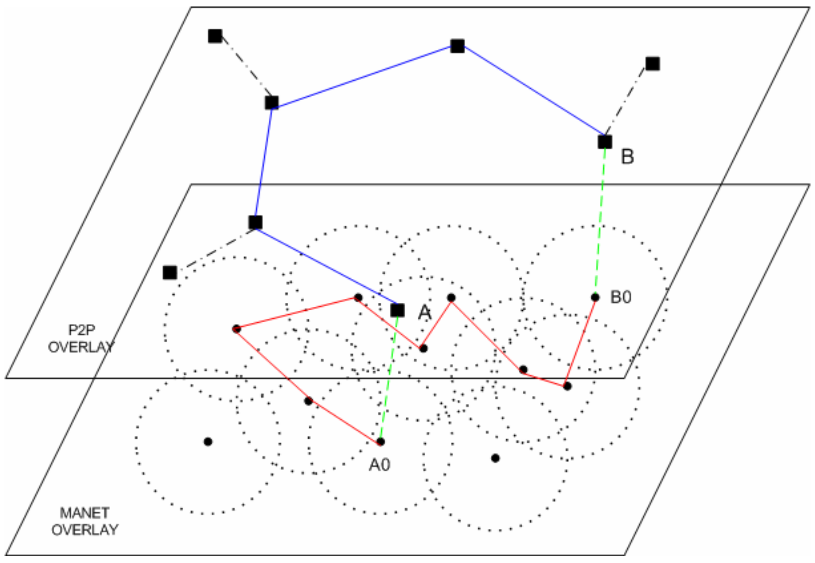

Figure 1. Broadcast over Broadcast

The scheme is illustrated in Figure 1 with a searching example: peer *A* in the P2P overlay is trying to search for a particular piece of information, which is actually available in peer *B*. Due to the broadcast mechanism, the search request is transmitted to *A*'s neighbors, and recursively to all the members in the network, until a match is found or timeout. There is a blue line representing the routing path at the application layer. Then we map this searching process into the MANET overlay, where node *A0* is the corresponding mobile node to the peer *A* in the P2P overlay, and *B0* is related to *B* in the same way. Since the MANET overlay also employs a broadcast-like routing protocol, the request from node *A0* is flooded (broadcast) to its directly connected neighbors, which themselves flood their neighbors etc., until the request is answered or a maximum number of flooding steps occur. The route establishing lines in that network layer is highlighted in red, where we can find that there are few overlapping routes between these two layers though each of them employs a broadcast-like protocol.

We have studied Guntella [19], a typical broadcast-like P2P protocol [20]. This is a pure P2P protocol, as shown in Figure 2, in which no advertisement of shared resources (e.g. directory or index server) occurs. Instead, each request from a peer is broadcasted to its directly connected peers, which themselves broadcast this request to their directly connected peers etc., until the request is answered or a maximum number of broadcast steps occur. It is easy to see that this protocol requires a lot of network bandwidth, and it does not prove to be very scalable. The complexity of this routing algorithm is $O(n)$ [21, 22].

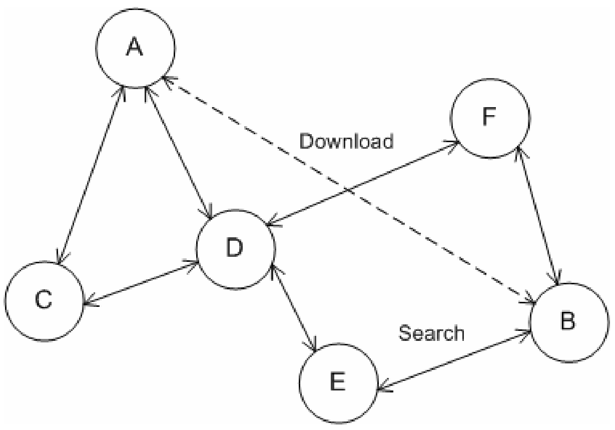

Figure 2. Broadcast-like P2P Protocol

Generally, most on-demand MANET protocols, like DSR [23] and AODV [24], are broadcast-like in nature [25]. Previously, we have studied AODV, one typical broadcast-like MANET protocol [26]. As shown in Figure 3, in that protocol, each node maintains a routing table only for active destinations: when a node needs a route to a destinations, a path discovery procedure is started, based on a RREQ (route request) packet; the packet will not collect a complete path (with all IDs of involved nodes) but only a hop count; when the packet reaches a node that has the destination in its routing table, or the destination itself, a RREP (route reply) packet is sent back to the source (through the path that has been set-up by the RREQ packet), which will insert the destination in its routing table and will associate the neighbour from which the RREP was received as preferred neighbour to that destination. Simply speaking, when a source node wants to send a packet to a destination, if it does not know a valid route, it initiates a route discovery process by flooding RREQ packet through the network. AODV is a pure on-demand protocol, as only nodes along a path maintain routing information and exchange routing tables. The complexity of this routing algorithm is $O(n)$ [27].

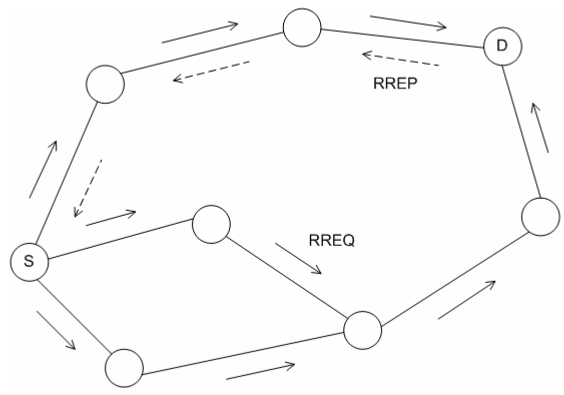

Figure 3. Broadcast-like MANET Protocol

This approach is probably the easiest one to implement, but the drawback is also obvious: the routing path of the requesting message is not the shortest path between the source and destination (e.g. the red line in Figure 1), because *virtual* neighbors in the P2P overlay are not necessarily *physical* neighbors in the MANET overlay, and actually these nodes might be physically far away from each other. Therefore, the resulting routing algorithm complexity of this broadcast over broadcast scheme is unfortunately $O(n^2)$ though each layer's routing algorithm complexity is $O(n)$ respectively.

It is not practical to deploy such kind of scheme for its serious scalability problem due to the double broadcast. Taking the energy consumption portion into consideration, which is critical to mobile devices, the double broadcast will also cost a lot of energy, and make it infeasible in cellular wireless networks.

3.2. DHT over Broadcast

The scalability problem of broadcast-like protocols has long been observed and many revisions and improvement schemas are proposed [28, 29, 30]. To overcome the scaling problems in broadcast-like protocols where data placement and overlay network construction are essentially random, there are a number of proposals on structured overlay designs. Distributed Hash Table (DHT) [31] and its varieties [32, 33, 34] advocated by Microsoft Research seem to be promising routing algorithms for overlay networks. Therefore it is interesting to see the second approach: to employ a DHT-like P2P routing protocol at the application layer over a broadcast-like MANET routing protocol at the network layer.

The scheme is illustrated in Figure 4 with the same searching example. Compared to the previous approach, the difference lies in the P2P overlay: in a DHT-like protocol, files are associated to keys (e.g. produced by hashing the file name); each node in the system handles a portion of the hash space and is responsible for storing a certain range of keys. After a lookup for a certain key, the system returns the identity (e.g. the IP address) of the node storing the object with that key. The DHT functionality allows nodes to put and get files based on their key, and each node handles a portion of the hash space and is responsible for a certain key range. Therefore, routing is location-deterministic distributed lookup (e.g. the blue line in Figure 4).

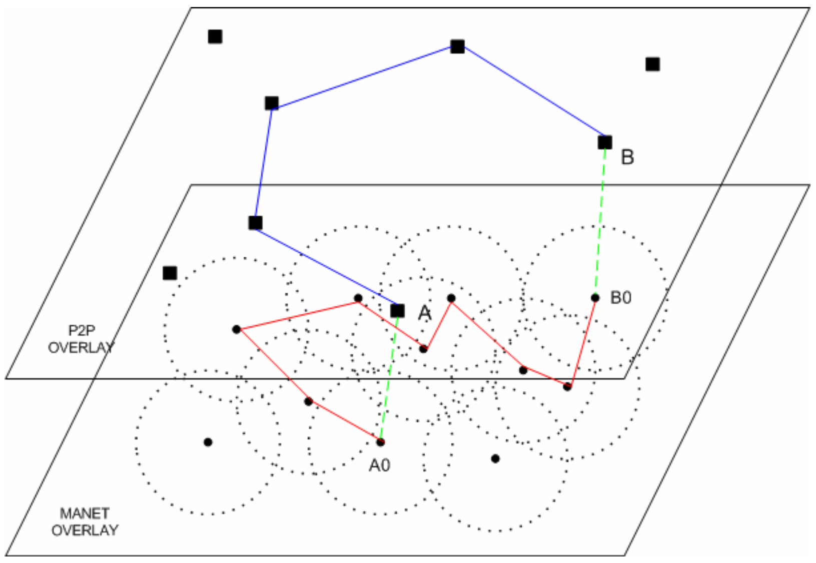

Figure 4. DHT over Broadcast

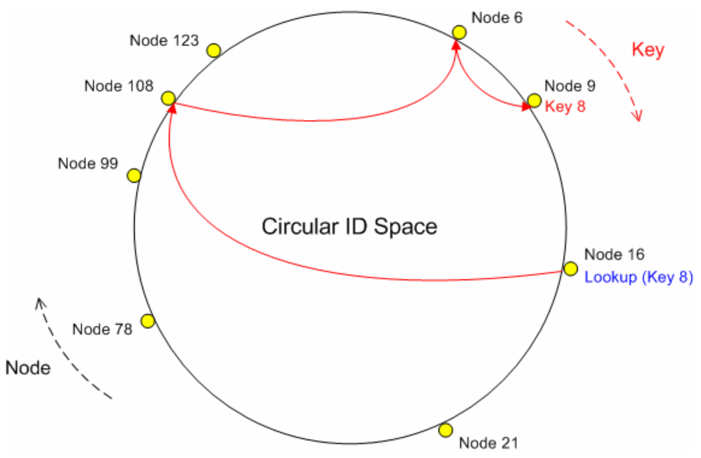

Figure 5. DHT-like P2P Protocol

DHT was first proposed by Plaxton [35], and soon proved to be a useful substrate for large distributed systems. A number of projects are proposed to build Internet-scale facilities layered above DHTs, among them are Chord [31], CAN [32], Pastry [33], Tapestry [34] etc. As illustrated in Figure 5, all of them take a key as input and route a message to the node responsible for that key. Nodes have identifiers, taken from the same space as the keys. Each node maintains a routing table consisting of a small subset of nodes in the system. When a node receives a query for a key for which it is not responsible, the node routes the query to the *hashed* neighbor node towards resolving the query. In such a design, for a system with n nodes, each node has $O(log\ n)$ neighbors, and the complexity of the DHT-like routing algorithm is $O(\log n)$ [36].

Additional work is required to implement this approach, partly because DHT requires periodical maintenance (i.e. it is just like an Internet-scale hash table, or a large distributed database): since each node maintains a routing table (i.e. hashed keys) to its neighbors according to the DHT algorithm, following a node join or leave, there is always a nearest key reassignment between nodes.

DHT over Broadcast approach is obviously better than the previous one, but it still does not solve the shortest path problem as in the Broadcast over Broadcast scheme. Though the P2P overlay algorithm complexity is optimized to $O(log\ n)$, the mapped message routing in the MANET overlay is still in the broadcast fashion with complexity $O(n)$; the resulting algorithm complexity of this approach is as high as $O(n\ log\ n)$.

This approach still requires a lot of network bandwidth, and hence does not prove to be very scalable, but could be efficient in limited communities, such as a company network.

3.3. Cross-Layer Routing

A further step of the Broadcast over Broadcast approach would be a Cross-Layer Broadcast. Due to the similarity of Broadcast-like P2P and MANET protocols, the second broadcast could be skipped if the peers in the P2P overlay would be mapped directly into the MANET overlay, and the result of this approach would be the merge of application layer and network layer (i.e. the *virtual* neighbors in P2P overlay overlaps the *physical* neighbors in MANET overlay).

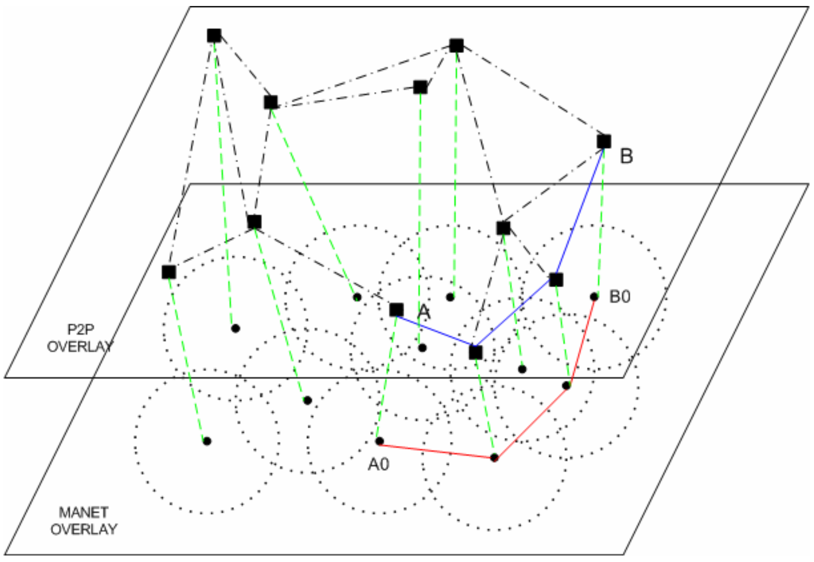

Figure 6. Cross-Layer Broadcast

The scheme is illustrated in Figure 6, where the advantage of this cross-layer approach is obvious: the routing path of the requesting message is the shortest path between source and destination (e.g. the blue and red lines in Figure 6), because the *virtual* neighbors in the P2P overlay are *de facto physical* neighbors in the MANET overlay due to the merge of two layers. Thanks to the nature of broadcast, the algorithm complexity of this approach is $O(n)$, making it suitable for deployment in relatively large scale networks, but still not feasible for Internet scale networks.

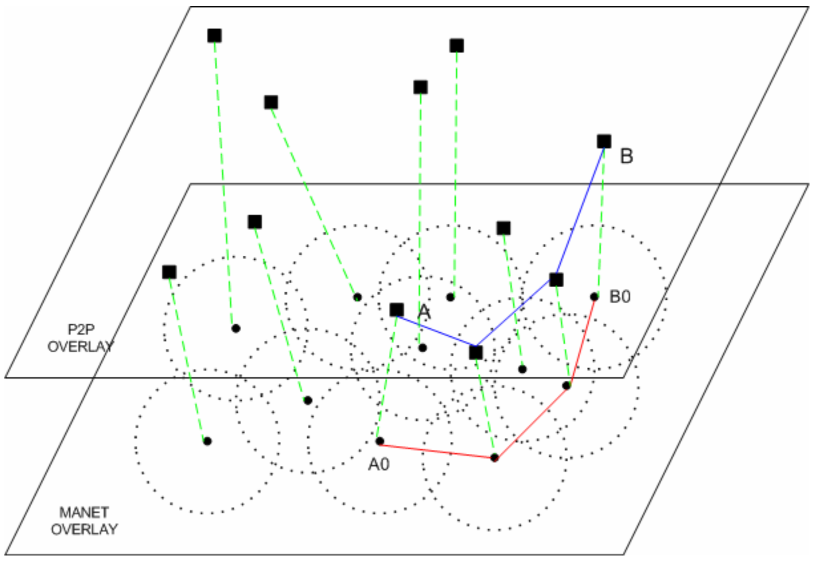

Figure 7. Cross-Layer DHT

It is also possible to design a Cross-Layer DHT in Figure 7 with the similar inspiration, and the algorithm complexity would be optimized to $O(log\ n)$ with the merit of DHT, which is advocated to be efficient even in Internet scale networks. The difficulty in that approach is implementation: there is no off-the-shelf DHT-like MANET protocol as far as we know, though recently, some research projects, like Ekta [37], towards a DHT substrate in MANET are proposed.

As an answer to Question 1, we show the cross-layer approach performs better than separating the overlay from the access networks, with the comparison of different settings for the peer-to-peer overlay and underlaying mobile ad hoc network in above four approaches in Table 1.

Table 1. How efficient does a user try to find a specific piece of data?

	Efficiency	Scalability	Implementation
Broadcast over Broadcast	$O(n^2)$	N.A.	Easy
DHT over Broadcast	$O(n\ log\ n)$	Bad	Medium
Cross-Layer Broadcast	$O(n)$	Medium	Difficult
Cross-Layer DHT	$O(log\ n)$	Good	N.A.

4. Modeling Download Performance

The download performance modeling is a relatively new issue compared to the search performance modeling, which was already extensively studied in some P2P and MANET research [38, 39, 40]. In this section, we present our efforts towards a performance model of downloading in such kind of systems, and thus answer Question

2: “what is the performance experience when many users try to retrieve data with parallel downloading scheme?”

4.1. Preliminary Assumptions

Though early research on modeling has mainly focused on routing performance and searching efficiency, recently, there were some works on modeling the download performance. The Markov chain approach has been brought forth [41] for a queue system model and some measurement studies were mentioned [42]; more recently, Stochastic fluid models are studied [43, 44, 45], which provide a more intuitive and deterministic approach. Our work uses the same approach as [45, 46]; but taking the idea into mobile environments, more realistic scenarios and physical constraints should be introduced, and old notions should have new interpretations.

Since the introduction of Tornado Code [47, 48] has been a popular technique on recently parallel downloading systems, here we assume: (1) the parallel download process in our model is Tornado-like, which reduces the requirement for coordination and signalling. Due to the limited bandwidth of existing wireless networks (probably accompanied with expensive data transmission charge, e.g. cellular network), (2) it is reasonable to allow the *pure downloader* (i.e. *leech*) exist in the system. Therefore, as illustrated in Figure 8, there are three types of peers in our model: (a) normal peer (i.e. *contributor*), which owns part of the file (i.e. ordinary downloader), but still allows others to download from itself. This type is the most common one and it actually constitutes the majority in our system. (b) pure downloader (i.e. *leech*), which just downloads but never uploads. The realistic implication of this type may be physically constrained mobile devices (e.g. cellular phones with limited bandwidth or associated with too expensive data transmission charge). (c) pure uploader (i.e. *seed*), which already have all pieces of the file but still stays in the system to allow others to download from itself. The realistic implication of that type may be content publishers (e.g. mobile operator’s service point).

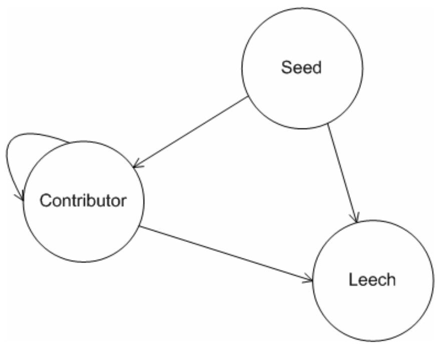

Figure 8. Three Types of Peers

Although there is heterogeneity in realistic infrastructure [49], such as bandwidth, latency, availability, etc., here we make a trade-off between the simplicity of the model and its ability to capture all facets, and assume (3) all peers in our model have equal capacity (i.e. all peers have the same upload and download bandwidth). With the above assumptions and the parameters in Table 2, we can derive that at time *t*, there are $\beta\, x(t)$ leeches and $(1-\beta)\, x(t)$ contributors in our system.

Table 2. Parameters Used in the Model

Parameter	Meaning
$x(t)$	Number of downloaders (i.e. contributors and leeches) at time *t*
β	Selfish rate (i.e. leech portion)
$y(t)$	Number of seeds at time *t*
λ	Arrival rate of new download request (Possion process)
μ	Upload bandwidth of each peer
τ	Download bandwidth of each peer
ρ	Abort rate of downloaders
κ	Leave rate of seeds

4.2. The Model

The queue-like model of one peer in our system is illustrated in Figure 9. As noted here, during the download and upload process, it is also possible that peers will get offline or abort the process, and in order to make the model simple, here we use abort rate ρ and leave rate κ to model these interrupted processes.

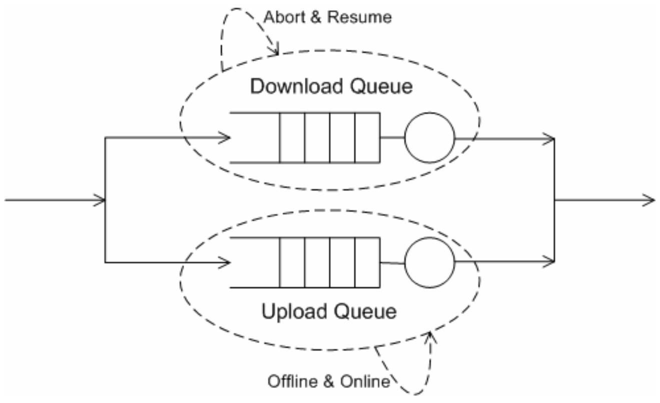

Figure 9. Queue-Like Model of One Peer

In a P2P download and upload scheme, it is natural to expect more on the download side (i.e. this implies $\tau \geq \mu$); so taken the download bandwidth constraint into account, the total upload bandwidth should be $min(\mu((1-\beta)\, x(t) + y(t)), \tau\, x(t))$, and the arrival and departure rate of download request will be λ and $min(\mu((1-\beta)\, x(t) + y(t)), \tau\, x(t)) + \rho\, x(t))$ respectively. The arrival and departure rate of upload request will be $min(\mu((1-\beta)\, x(t) + y(t)), \tau\, x(t))$ and $\kappa\, y(t)$. Thus the fluid model is derived as

$$\frac{d}{dt}x(t) = \lambda - \min[\mu[(1-\beta)x(t) + y(t), \tau x(t)]] - \rho x(t)$$

$$\frac{d}{dt}y(t) = \min[\mu[(1-\beta)x(t) + y(t), \tau x(t)]] - \kappa y(t)$$

In a steady state, the number of downloaders and seeds should be independent of time (i.e. $d(x(t))/dt = d(y(t))/dt = 0$); and then if we define

$$\frac{1}{\iota} = \frac{1}{1-\beta}\cdot\left(\frac{1}{\mu} - \frac{1}{\kappa}\right)$$

where ι can be interpreted as *effective* upload bandwidth compared to *nominal* upload bandwidth μ (i.e. after considering the impact of leeches), those equations can be solved as

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \dfrac{\lambda}{\tau\left(1+\dfrac{\rho}{\tau}\right)} \\ \dfrac{\lambda}{\kappa\left(1+\dfrac{\rho}{\tau}\right)} \end{pmatrix} \quad \text{when} \quad \frac{1}{\tau} \geq \frac{1}{\iota}$$

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \dfrac{\lambda}{\iota\left(1+\dfrac{\rho}{\iota}\right)} \\ \dfrac{\lambda}{\kappa\left(1+\dfrac{\rho}{\iota}\right)} \end{pmatrix} \quad \text{when} \quad \frac{1}{\tau} < \frac{1}{\iota}$$

where the limited download bandwidth and limited upload bandwidth is the constraint respectively. Furthermore, if we define

$$\frac{1}{\phi} = \max\left(\frac{1}{\tau}, \frac{1}{\iota}\right)$$

where φ can be interpreted as *bottleneck* bandwidth intuitively, we obtain the solution as

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \dfrac{\lambda}{\phi\left(1+\dfrac{\rho}{\phi}\right)} \\ \dfrac{\lambda}{\kappa\left(1+\dfrac{\rho}{\phi}\right)} \end{pmatrix}$$

Finally, we derive the average download time for a peer with Little's Law [50]

$$\text{Time} = \frac{1}{\phi + \rho} \quad \text{where} \quad \frac{1}{\phi} = \max\left(\frac{1}{\tau}, \frac{1}{\iota}\right)$$

5. Performance Analysis with the Model

In the model presented in the previous section, it is clear that different settings of β, μ, τ, ρ and κ will lead to different performance; so in this section we will use our analysis model to provide some insights in the network.

5.1. Selfish Peers

For a fixed set of network parameters, we first study the impact of β on the network performance. The realistic interpretation of β is interesting, which is somehow related to peer strategy and incentive mechanism (i.e. *selfish* peers or *leeches*).

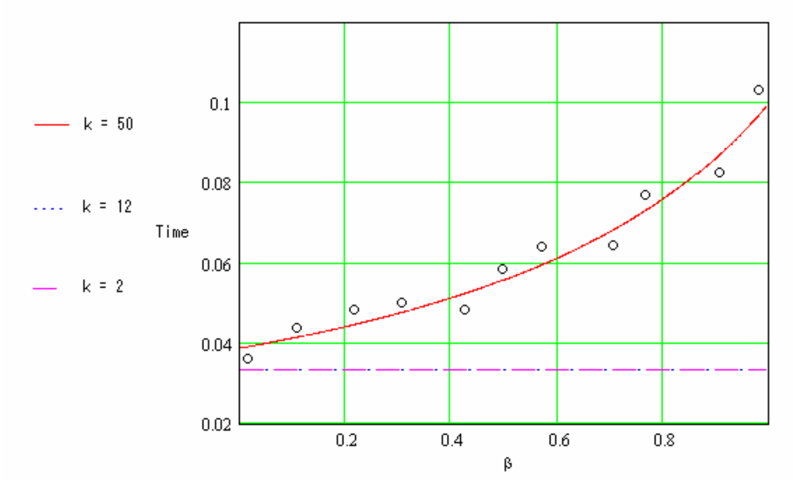

Figure 10. Impact of β on Network Performance

The network parameters we have chosen are: μ = 12kbps, τ = 20kbps, ρ = 10kbps, κ_0 = 50kbps, κ_1 = 12kbps, κ_2 = 2kbps. In this scenario, we consider the effect of *selfish* peers. Intuitively, the existing *leeches* will degrade the system performance because they just download from others and never upload. The red curve in Figure 10 for κ_0 = 50kbps justifies our intuition.

From the observation, it is obvious that *Time* is a non-decreasing function of β. We can also find the upper bound and lower bound of *Time* if we consider two extreme cases: $\beta = 1$ (i.e. all downloaders are *selfish* and no one uploads to others) and $\beta = 0$ (i.e. there is no *leeches* in the system).

At this point, we are all happy with our intuition; but if we change the value of κ into κ_1 = 12kbps and κ_2 = 2kbps, something strange happens. As shown in Figure 10 as two overlapped horizontal lines, the network performance is constant, independent of β. We briefly comment on this situation: recall the *bottleneck* bandwidth definition in the previous section, it actually means the downloading bandwidth is the bottleneck since $\mu \geq \kappa$; in such a situation, the *leeches* make no harm to the system since the whole system performance is constrained by the limited download speed (i.e. selfishness is *not* always harmful).

From this phenomenon, we argue that it is reasonable to introduce *leeches* into our model as in our preliminary assumptions, and actually there are lots of *leeches* existed in realistic systems. In other words, *what is real is rational and what is rational is real.*[1]

5.2. Download Bandwidth’s Role

In the previous subsection, we have seen the download bandwidth’s impact on the system performance. Intuitively, increasing the download bandwidth will lead to a shorter downloading time, as often observed in our daily experiences; but is this common sense always true? Now we study the impact of τ on the system performance (i.e. download bandwidth’s role).

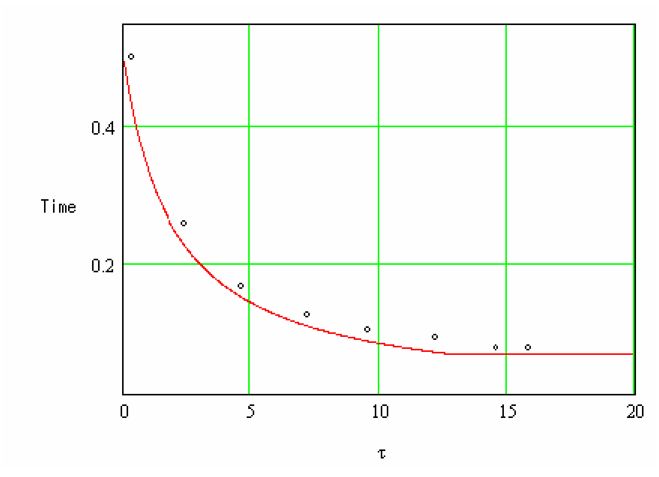

Figure 11. Impact of τ on Network Performance

The network parameters we have chosen are: $\beta = 0.2$, $\mu = 12$kbps, $\rho = 2$kbps, $\kappa = 50$kbps. Shown as the red curve in Figure 11, *Time* is a non-increasing function of τ. Besides, we can also derive the upper bound and lower bound of *Time* if we set $\tau = 0$ (i.e. the download channel is actually blocked) and $\tau = \infty$ (i.e. the download bandwidth is *much* higher than upload bandwidth) respectively.

The left half part of the curve justifies our intuition perfectly, but the right half seems to yaw from the common sense. The key to the phenomenon is still *bottleneck* bandwidth: initially, when τ increases, *Time* decreases accordingly because download bandwidth is the bottleneck now; however, once τ becomes big enough, increasing τ will not decrease *Time* any more, because the download bandwidth is no longer the bottleneck of the system performance.

In fact, if we consider the impact of μ on network performance (i.e. upload bandwidth’s role), we will get a similar curve. From these phenomena, we argue that there are not always performance gains with increased download bandwidth, and the key to network performance gains is to keep a good balance of download bandwidth

[1] Taken from Hegel's famous dictum *Das Wirkliche sei vernuenftig und das Vernuenfitige wirklich.*

and upload bandwidth, and actually to increase bottleneck bandwidth. In other words, *every coin has two sides.*[2]

5.3. Importance of Seeds

The *seeds* are a special kind of peers, which upload but don't download. Compared to *leeches*, seeds can be deemed as *selfless* peers. Intuitively, it is very important to have seeds in the system; and in this subsection, we study the impact of κ on the system performance (i.e. seeds' contribution).

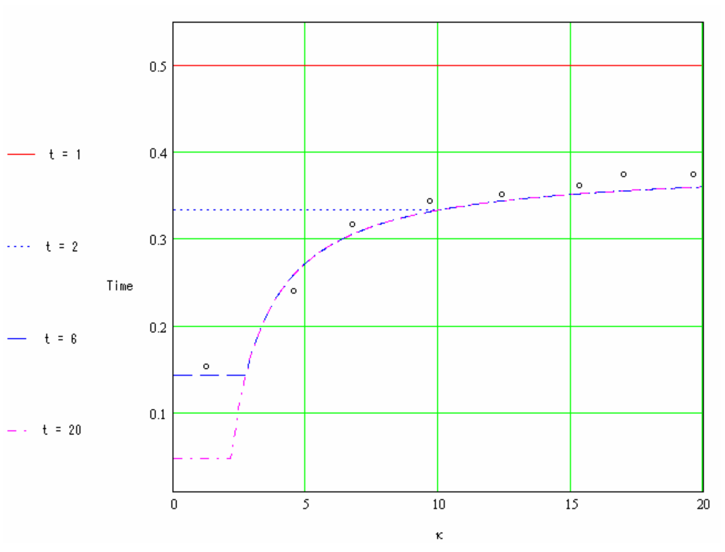

Figure 12. Impact of κ on Network Performance

The network parameters we have chosen are: β = 0.2, μ = 2kbps, ρ = 1kbps, τ_0 = 1kbps, τ_1 = 2kbps, τ_2 = 6kbps, τ_3 = 20kbps. With the curves shown in Figure 12, we are now not surprised to see the divisions of these curves and their singular points, because we already know their roots in the *bottleneck* bandwidth concept. Here we just briefly comment on the situation τ_2 = 6kbps because this speed seems to coincide with the practical speed of our daily cellular networks (e.g. GPRS): the ideal scenario is κ = 0 (i.e. all seeds are persistent in the network), where the lower bound of *Time* resides. As κ increases, initially, the slight loss of seeds doesn't degrade the system performance since the system is download bandwidth constrained; however, once κ is big enough, the system turns into upload bandwidth constrained, and the system performance degrades sharply with the loss of seeds; this also explains the singular point in the curve.

The realistic interpretation of *seeds* is service points or completed downloaders (but not all completed downloaders become seeds due to the existence of *leeches*), and the realistic meaning of the phenomenon is: it would be an effective way for mobile operators to improve QoS in such kind of systems via providing more service points.

[2] Ancient proverb.

6. Concluding Remarks

In this paper, we first studied the peer-to-peer systems over mobile ad hoc networks with a comparison of different settings for the peer-to-peer overlay and underlaying mobile ad hoc network. We show that cross-layer approach performs better than separating the overlay from the access networks. After characterizing the variability of the system by taking some preliminary assumptions, we then present a performance model which captures most facets of mobile peer-to-peer systems. We also briefly discussed three analytical examples on apply this model to capture the behavior of the system in steady states.

In order to make the paper concise, we didn't use the model to analyze the system in inequilibrious states, though it is not hard to simulate these cases with the given fluid model. We hope our results would potentially provide useful guidelines for mobile operators, value-added service providers and application developers to design and dimension mobile peer-to-peer systems, and as a foundation for our long term goals [51, 52].

References

[1] G. Kortuem, J. Schneider, D. Preuitt, T. G. C. Thompson, S. Fickas, Z. Segall. When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad hoc Networks. In *Proc. 1st International Conference on Peer-to-Peer Computing (P2P 2001)*, Linkoping, Sweden, August 2001.

[2] M. Papadopouli and H. Schulzrinne. A Performance Analysis of 7DS a Peer-to-Peer Data Dissemination and Prefetching Tool for Mobile Users. In *Advances in wired and wireless communications, IEEE Sarnoff Symposium Digest*, 2001, Ewing, NJ.

[3] C. Lindemann and O. Waldhorst. A Distributed Search Service for Peer-to-Peer File Sharing in Mobile Applications. In *Proc. 2nd IEEE Conf. on Peer-to-Peer Computing (P2P 2002)*, 2002.

[4] A. Klemm, Ch. Lindemann, and O. Waldhorst. A Special-Purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks. In *Proc. IEEE Vehicular Technology Conf.*, Orlando, FL, October 2003.

[5] S. K. Goel, M. Singh, D. Xu. Efficient Peer-to-Peer Data Dissemination in Mobile Ad-Hoc Networks. In *Proc. International Conference on Parallel Processing (ICPPW '02)*, IEEE Computer Society, 2002.

[6] G. Ding, B. Bhargava. Peer-to-peer File-sharing over Mobile Ad hoc Networks. In *Proc. 2nd IEEE Conf. on Pervasive Computing and Communications Workshops*. Orlando, Florida, 2004.

[7] H.Y. Hsieh and R. Sivakumar. On Using Peer-to-Peer Communication in Cellular Wireless Data Networks. In *IEEE Transaction on Mobile Computing*, vol. 3, no. 1, January-March 2004.

[8] B. Bakos, G. Csucs, L. Farkas, J. K. Nurminen. Peer-to-peer protocol evaluation in topologies resembling wireless networks. An Experiment with Gnutella Query Engine. In *Proc. International Conference on Networks*, Sydney, Oct., 2003.

[9] T. Hossfeld, K. Tutschku, F. U. Andersen, H. Meer, J. Oberender. Simulative Performance Evaluation of a Mobile Peer-to-Peer File-Sharing System. *Research Report 345*, University of Wurzburg, Nov. 2004.

[10] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu. Peer-to-Peer Computing. *Technical Report HPL-2002-57*, HP Labs.

[11] *MANET Implementation Survey*. Available at http://protean.itd.nrl.navy.mil/manet/survey/survey.html

[12] Gnutella: http://www.gnutella.com/

[13] Freenet: http://freenet.sourceforge.net/

[14] Kazaa: http://www.kazaa.com/

[15] eMule: http://www.emule-project.net/

[16] BitTorrent: http://bittorrent.com/

[17] *DSR IETF draft v1.0*. Available at http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt

[18] *AODV IETF draft v1.3*. Available at http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-13.txt

[19] Clip2. *The gnutella protocol specification v0.4 (document revision 1.2).* Available at http://www9.limewire.com/developer /gnutella protocol 0.4.pdf, Jun 2001.

[20] L. Yan and K. Sere. Stepwise Development of Peer-to-Peer Systems. In *Proc. 6th International Workshop in Formal Methods (IWFM'03).* Dublin, Ireland, July 2003.

[21] M. Ripeanu, I. Foster and A. Iamnitch. Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design. In *IEEE Internet Computing*, vol. 6(1) 2002.

[22] Y. Chawathe, S. Ratnasamy, L. Breslau, S. Shenker. Making Gnutella-like P2P Systems Scalable. In *Proceedings of ACM SIGCOMM,* 2003.

[23] D. B. Johnson, D. A. Maltz. Dynamic Source Routing in Ad-Hoc Wireless Networks. In *Mobile Computing*, Kluwer, 1996.

[24] C. E. Perkins and E. M. Royer. The Ad hoc On-Demand Distance Vector Protocol. In *Ad hoc Networking*. Addison-Wesley, 2000.

[25] F. Kojima, H. Harada and M. Fujise. A Study on Effective Packet Routing Scheme for Mobile Communication Network. In *Proc. 4th Intl. Symposium on Wireless Personal Multimedia Communications*, Denmark, Sept. 2001.

[26] L. Yan and J. Ni. Building a Formal Framework for Mobile Ad Hoc Computing. In *Proc. International Conf. on Computational Science (ICCS 2004).* Krakow, Poland, June 2004. LNCS 3036, Springer-Verlag.

[27] E. M. Royer and C. K. Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. In *IEEE Personal Communications*, April 1999.

[28] Q. Lv, S. Ratnasamy and S. Shenker. Can Heterogeneity Make Gnutella Scalable? In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*, Cambridge, MA, March 2002.

[29] B. Yang and H. Garcia-Molina. Improving Search in Peer-to-Peer Networks. In *Proc. Intl. Conf. on Distributed Systems (ICDCS)*, 2002.

[30] Y. Chawathe, S. Ratnasamy, L. Breslau, and S. Shenker. Making Gnutella-like P2P Systems Scalable. In *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, August 2003.

[31] I. Stoica, R. Morris, D. Karger, F. Kaashoek and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proc. ACM SIGCOMM*, 2001.

[32] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Schenker. A scalable content-addressable network. In *Proc. Conf. on applications, technologies, architectures, and protocols for computer communications*, ACM, 2001.

[33] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, pages 329-350, November, 2001.

[34] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz. Tapestry: A Resilient Global-scale Overlay for Service Deployment. In *IEEE Journal on Selected Areas in Communications*, January 2004, Vol. 22, No. 1.

[35] C. Plaxton, R. Rajaraman, A. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *Proc. ACM SPAA*, Rhode Island, June 1997.

[36] S. Ratnasamy, S. Shenker, I. Stoica. Routing Algorithms for DHTs: Some Open Questions. In *Proc. 1st International Workshop on Peer-to-Peer Systems*, March 2002.

[37] H. Pucha, S. M. Das and Y. C. Hu. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *Proc. 6th IEEE Workshop on Mobile Computing Systems and Applications*, December 2004, UK.

[38] R. Schollmeier and I. Gruber. Routing in Peer-to-Peer and Mobile Ad Hoc Networks. A Comparison. In *Proc. International Workshop on Peer-to-Peer Computing*, Pisa, Italy, 2002.

[39] P. Reynolds and A. Vahdat. Efficient Peer-to-Peer Keyword Searching. In *Proc. Middleware*, 2003.

[40] S. Corson and J. Macker. Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations. *RFC 2501*, Jan. 1999.

[41] Z. Ge, D. Figueiredo, S. Jaiswal, J. F. Kurose, D. Towsley. Modeling peer-to-peer file sharing systems. In *Proc. IEEE Infocom*, 2003.

[42] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proc. 19th ACM symposium on Operating systems principles*, 2003.

[43] F. Clevenot and P. Nain. A Simple Fluid Model for the Analysis of the Squirrel Peer-to-Peer Caching System. In *Proc. IEEE Infocom*, 2004.

[44] X. Yang and G. Veciana. Service Capacity of Peer to Peer Networks. In *Proc. IEEE Infocom*, 2004.

[45] D. Qiu and R. Srikant. Modeling and Performance Analysis of BitTorrent-Like Peer-to-Peer Networks. In *Proc. ACM SIGCOMM*, 2004.

[46] F. L. Piccolo, G. Neglia. The Effect of Heterogeneous Link Capacities in BitTorrent-Like File Sharing Systems. In *Proc. Intl. Workshop on Hot Topics in Peer-to-Peer Systems (HOT-P2P'04)*, Oct, 2004.

[47] J. Byers, M. Luby, M. Mitzenmacher. Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads. In *Proc. IEEE Infocom*, 1999.

[48] J. W. Byers, J. Considine, M. Mitzenmacher and S. Rost. Informed Content Delivery Across Adaptive Overlay Networks. In *Proc. ACM SIGCOMM*, 2002.

[49] S. Saroiu, P.K. Gummadi, S.D Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proc. Multimedia Computing and Networking (MMCN'02)*, 2002.

[50] P. J. Denning and J. P. Buzen. The Operational Analysis of Queueing Network Models. In *ACM Computer Survey*, 1978.

[51] L. Yan, K. Sere, X. Zhou, and J. Pang. Towards an Integrated Architecture for Peer-to-Peer and Ad Hoc Overlay Network Applications. In *Proc. 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS 2004)*, May 2004.

[52] L. Yan. MIN: Middleware for Network-Centric Ubiquitous Systems. In *IEEE Pervasive Computing*, Vol. 3, No. 3, 2004.

ISBN 952-12-1529-1
ISSN 1239-1891