PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 49, NO. 3–4, PP. 195–207 (2005)

PIC – A PEER-TO-PEER PROTOCOL FOR MOBILE DEVICES

Kálmán MAROSSY

Department of Automation and Applied Informatics
Budapest University of Technology and Economics
H–1111 Budapest, Goldmann Gy. tér 3.
H–1521 Budapest, Hungary
e-mail: coloman@avalon.aut.bme.hu

Received: June 29, 2005

Abstract

In wireless and especially mobile communications the bandwidth and the amount of transferred data become key aspects. Due to the bandwidth limitations wireless devices may join P2P (Peer-to-Peer) content sharing networks only for a limited time period. Enhancements and possibly new protocols are necessary for wireless peer-to-peer applications.

Introducing intelligent search/indexing techniques we can reduce the amount of traffic in the network and balance user load (with some additional cost of implementation complexity). The Parallel Index Cluster (PIC) approach is proposed as an efficient candidate, as a network traffic reduction is expected with at least one order of magnitude compared to basic and enhanced Gnutella networks [1].

In this article a new modeling of P2P systems, the SIL (Search Index Link) [2] method is described, and based on this a new P2P protocol is introduced, which is suitable for mobile devices. For this new protocol (PIC) different cluster topologies are analyzed. To produce minimal network traffic, simulation results and mathematical analysis are given to optimize the cluster sizes in the network.

Keywords: protocols, peer-to-peer, file sharing, wireless application.

1. Introduction — Mobile P2P Systems in Closed User Group

1.1. P2P Systems

A peer-to-peer (P2P) system is a distributed network architecture, where the participants (peers, nodes) share their resources, and these resources in the system can be accessed directly by other peers. [3] Resources are mostly files, processing power and user presence. P2P systems are most commonly used for file-sharing purposes, probably everybody may have heard about Kazaa or the infamous Napster.

1.2. Mobile Environment

These P2P systems are widely used on the Internet with resource-rich computers (big processing capacity and memory, large available network bandwidth, always connected), but there is a growing demand for a suitable P2P system for mobile

phones. The mobile environment has its own characteristics, which should be taken into account, while the demands of the users are also different on mobile devices.

In a mobile environment the nodes have low bandwidth connection which can be interrupted (e.g. out of coverage), the battery power can also be a reason for short participations, and the computational capacity and the memory are limited. Though, with mobile devices not only IP communication can be used, but other wireless communication methods (e.g. Bluetooth, WLAN), too. Currently only file-sharing and user presence sharing (chat, whiteboard) are in our main interest line.

1.3. User Group

An important dimension in content sharing is the user group. The most commonly used peer-to-peer protocols (e.g. like Gnutella in [1]) are assuming open user group, so everybody can be part of the network and no registration is necessary to access the resources offered by the content sharing network. This is not the case, when the user group is closed. In a closed user group environment the users have to indicate that they are going to participate in a network. This process is called registration, and its opposite is called deregistration. When a user is registered, it can join the network, search and download content, and can leave the network.

PIC networks are proposed to be used primarily in the closed group scenario, where registered users form communities around topics of interest. The structure of the network topology could be adjusted to match the size of the user group. In a closed user group the goal is to find all content matching search criteria. This allows searching for unique and more rare content that may be necessary e.g. for many business applications. This makes it necessary to adopt an approach different from the case of open user groups, when virtually everybody can become member of the network.

1.4. Goal

To summarize, our goul is to create a peer-to-peer protocol, which can be used in a mobile environment for content sharing (requires low bandwidth and processing capabilities), the benefits of a closed user group environment can be utilized with it, 100% hit ratio can be achieved (i.e. all files in the network can be found with a search request), and provides good load balancing.

Practical applications of this type of content sharing could be groups of small (up to 100) or moderately large (up to 10000) size that share various type of contents, like mp3 music, photos taken with mobile devices, text documents or video clips.

2. Existing P2P Systems

2.1. Hybrid

A P2P system is classified as hybrid or centralized, if a central entity is necessary for the system to provide the content/services offered by the network. [3]. This is the architecture of the first P2P systems, e.g. Napster. In this case, the "indexes" describing a file (file name, ID of the host peer, category, quality, author, etc.) are stored on a central server, while the content (the files) are stored in a distributed network of peers.

These systems are simple to implement, fast (as far as the central server(s) can serve the requests), and a search in the network requires relatively low network traffic. The main disadvantage of this architecture is that it has all the disadvantage a centralized system can have. If the central server is out of order, or it is forbidden to operate (like at the case of Napster), the whole network is inaccessible.

This architecture on its own is not suitable for a mobile environment, but the idea of index storing servers can be utilized in mixed networks (with not only mobile peers).

2.2. Pure

In the pure (or homogeneous) P2P architecture all peers are equal and do the same things, so every node is responsible for forwarding search request to other peers, and of course to response to the search request if the document is found (if there is a 'hit'). Instead of index maintenance (on a server, or on a distributed way, etc.) only a cache of query results is usual to maintain. The example for this architecture is the Gnutella protocol, which is widely used and there are many clients using this protocol (e.g. LimeWire, Shareaza, and Morpheus).

The advantage of this type of 'second generation' P2P networks is that they are totally distributed. A node can be easily removed from the network without endangering the network's proper operation. Though, a search request doesn't reach every node in a network, so maybe a document in the network will not be found. This could be not suitable in some smaller mobile usage cases.

The bigger disadvantage of this architecture is that it generates large network traffic for finding a file as every node forwards the search to its neighbors for a given depth. This is not suitable for a low-bandwidth environment. Though there are many attempts to decrease the generated network traffic with acceptable hit ratio, e.g. in [1] the adaptive TTL method.

2.3. Hierarchical

The commonly used P2P systems in our days are not centralized or fully distributed, rather have a hierarchical architecture. A peer with large amount of resources can be a super-node, and other peers with smaller capacity can be sub-nodes. So the network is separated into layers: the layer of the super nodes and the layer of the sub-nodes. [4]

Sub-nodes send their request only to the super node they belong to, so their network traffic is minimized. Also, they send their indexes to a super-node, so the bigger part of a search is done in the super-layer, where the bandwidth is sufficient. Though, to decrease network traffic a search doesn't reach all super-nodes typically.

Examples for this architecture can be the FastTrack system (Kazaa, Grokster, and Morpheus), WinMX, or the Gnutella 2 protocol (Shareaza).

The advantage of this architecture is, that it synthesizes the advantages of the previous two architectures: it is relatively fast, requires relatively low bandwidth, and the architecture is distributed enough to handle a removal of a super-node. Although the maintenance of the network requires larger effort than at the previous architectures.

This architecture can be good in a large, heterogeneous mobile environment, where the super-nodes are not mobile phones, because it is desirable for a super-node to be on-line almost all the time and to have large resources. Though, for small networks consisting of mobile phones with equally low resources this architecture is not suitable.

2.4. Distributed Indexing

Storing and handling indexes can reduce the search traffic radically, and the search time is also shorter. The first (centralized) architecture has also tried to utilize this. There are other techniques to use indexes than storing them on a server: they can be stored in a distributed way. There are several architectures with very fast index lookup, which store indexes in a ring (e.g. Chord [6]) or a tree architecture (e.g. P-Grid [7]).

To be able to do fast logarithmically scaling search, the indexes are usually stored in binary form (with a hash function), and therefore only exact match can be easily checked. Another consequence is that search can be made against one criteria (one index is stored for a document).

There are many attempts to overcome these limitations. One solution is to combine it with the previous architectures, like in Gridella [5]. Another approach is to use multi-layer distributed indexes. If we create a layer for every content category, we can get search for other criteria, by selecting the index layer in which the search is made [8]. The search also can be faster if we consider the physical location of the peers, like in Grapes [9].

The fast and low cost search sounds well when mobile devices are the targets,

but these algorithms are too complicated to implement and not simple to use in a dynamically changing mobile environment (the index structure must adapt all the time), so this approach doesn't fit well in a mobile view.

3. PIC — Parallel Index Clusters

3.1. The SIL model

There are other methods to classify a P2P system than viewing the type of the architecture. File sharing P2P systems can be usually described with the SIL model [2], where the network used for searching and indexing are separated. A peer has several "logical" connections to another peer, which can be used for starting to search or to send indexes (file describing metadata). Such a network link can be also forwarding or non-forwarding. A message got in a forwarding link is forwarded to other peers, while messages delivered through non-forwarding links are not forwarded.

Describing a network with the SIL model can reveal the redundancy introduced with indexing. Too much redundancy can lead to performance issues, while the avoidance of every redundant element usually leads to low fault tolerance.

For example, the hierarchical (super-node) network can also be described with the SIL model. The super-network is a connected directed graph with forwarding search links (FSL). A peer in the sub-network has one non-forwarding index link (NIL) and a FSL to a super-node. (*Fig. 1a*)

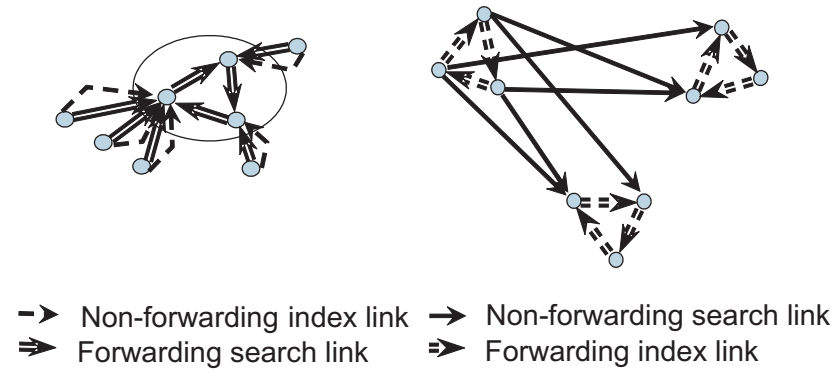

Fig. 1. a) the super-node network and b) the PIC network with the SIL model

With this model we can find other interesting topologies, like parallel search cluster or parallel index cluster. In the former, there are several groups of nodes (cluster) connected with FSLs. From each node, there are NILs to every cluster. In the latter, there are forwarding index links (FIL) inside a cluster, and non-forwarding search links (NSL) to other clusters (see *Fig. 1 b*).

3.2. Analyzing Methods

With SIL model a PIC system can be described, but this is too general to create a protocol based solely on it. A lot of free parameters (like topologies and cluster sizes) have to be chosen, which leads several PIC 'variations'. For evaluating a concrete PIC variation, we have used simulation methodology.

To compare the variations two measured values have been selected: $AvgLoad$ (average load) represents the generated network traffic in a network. $MaxLoad$ (maximal load of a node) was also measured, and it represents the load balancing facilities. When $MaxLoad$ is close to $AvgLoad$, it means a good load balancing.

A simulation was always made in the chosen network with regarding to different user behavior. User behavior is represented by the ratio of the different type of user requests: search requests (SR) and file uploads (index update requests, UR). The SR/UR ratio had always the same discrete values in the [0.1, 10] interval, like in [2]. The number of user request was always constant (10) times the network size.

The used simulator was the same as in [1] used for Adaptive TTL measures, integrated with a new general PIC protocol module.

3.3. Choosing Topologies

There are several directions, where the SIL model can be extended. In practice, there are not only search and index links, but the network is changing: peers join to the network, others leave.

These actions are made via 'registration' links. In a mobile environment temporarily leaving the network and later rejoining it is also usual, therefore a new link type could be also introduced. Another direction is to introduce not only two dimensions in the parallel index/search cluster model, but more.

We have analyzed only the former generalization (different kind of link types) and we have identified several topologies for a link type. For an example of the role of the different topologies, let us see the case of the index updating, as the index update topology in a cluster is important because it determines the speed and load balancing parameters.

The two extreme alternatives that we can propose are the following: Star topology, very fast index updating at all costs (even high traffic allowed) versus very low and highly balanced traffic at all costs (even slow index updating allowed) as in the Ring topology. Between these two, there are compromise solutions, like Random N, Planned N or Balanced N.

Star topology (*Fig. 2 a*) assures a very fast index updating. An updating node sends the update message to every co-cluster node. The links between the nodes in a cluster are bi-directional NILs. Index updates propagate in a very fast manner since every co-cluster node is informed simultaneously.

If the index links use TCP, the updating node has to maintain $n-1$ TCP links (n is the number of nodes in an index cluster) or to establish and terminate $n-1$ TCP

links, both being overwhelming for a mobile with limited processing capabilities. Star topology over UDP is feasible, requiring an application layer acknowledge.

Ring is the second alternative (*Fig. 2 b*). There are two main options: random ring and deterministic ring. In a random ring the next node is selected randomly, while in a deterministic ring the order of the nodes is fixed. The random ring needs a 'heard list' to keep account of nodes already visited. The deterministic ring does not need a 'heard list', therefore it is more efficient in terms of generated traffic. In both cases index update propagates slowly. Random rings use acknowledged UDP for the same reason as Star; deterministic rings use TCP. Index update balance is very good in both cases.

As a compromise between Star and Ring, other topologies are introduced (*Fig. 2c*), which provide faster index updates than Ring and better load balance than Star.

In Random N a node forwards an index update to N randomly chosen nodes (in the case when $N = 1$ that is random Ring). Because of this random choice the load balance is very good. However, we have found in simulations, that if the algorithm assures that each node gets the index update message, this can lead to large traffic.

In Planned N the source node (the origin of the index update) determines the index update tree, so a certain node will choose the same N destination nodes if the source node is the same, but another N nodes if the source node is another node. In *Fig. 3* the message flow can be seen, where the number of a node means the place of the node in the ordered list of the on-line nodes. This algorithm is easy to implement, and the index update is also very fast. But this kind of approach will result in a slightly unbalanced load in the topology if update generation rate is not uniform.

Balanced N solves this latter problem with an extra random variable. This random variable and the source node together determine the index update tree (which node sends the message to which nodes). With slightly larger messages (due to the random variable) better load balancing can be achieved.

Random N, Planned N, and Balanced N can use acknowledged UDP.

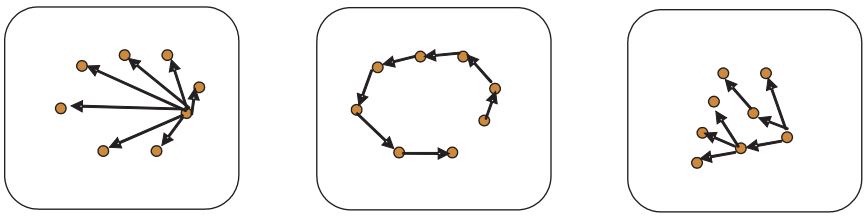

Fig. 2. Index update topologies a) Star b) Ring c) Random N / Planned N / Balanced N (on the figure $N = 3$)

4. Optimizing the Cluster Size in PIC for Network Traffic

As a starting-point, we assume that for a specific closed user group the SR/UR ratio is constant, but different user groups mean different users with different SR/UR ratio. So we wanted to find an appropriate cluster size for a network with a given SR/UR ratio in our simulations, where the least traffic is generated.

4.1. PIC Performance Optimization

If the SR/UR ratio is changing, we can analyze the effect of the change in the network traffic (if the node distribution remains unchanged). We have found that a certain number of clusters (equaling the square root of the size of the network) generate the same traffic even if the SR/UR ratio changes. (*Fig. 4*)

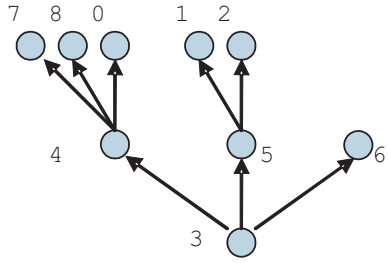

Fig. 3. A Planned 3 index updating scenario, node 3 is the source node

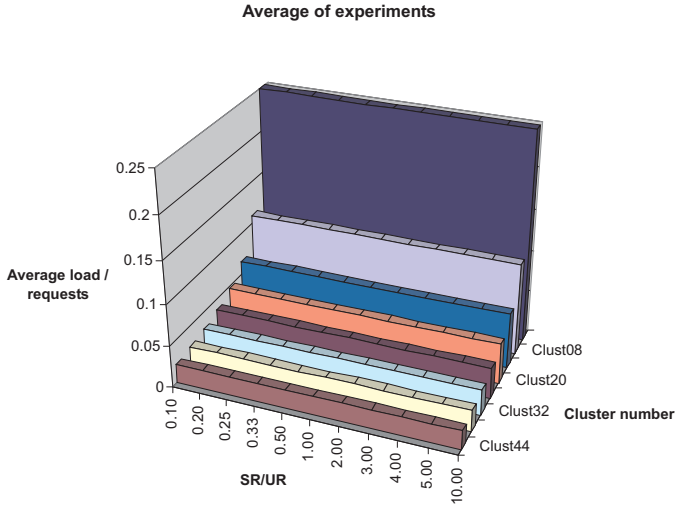

Fig. 4. Square root law

The network traffic generated by a PIC-like protocol largely depends on the ratio of the search and update requests (SR/UR). However, at a constant network size (node number), the number of the clusters and the number of nodes in each cluster can be calculated to generate the least traffic. We have found the following results.

If the SR/UR ratio is known and constant, the perfect node distribution among the clusters is also known. If SR/UR is low, there are many updates, and only a few searches, so many small clusters generate lower traffic. If SR/UR is large, there are many searches: a small number of large clusters are useful for this type of user behavior. Simulation results can be seen on *Fig. 5a*. The 'optimal' cluster size (where the traffic is the lowest) can be also calculated. *Fig. 5b* shows the results by Maple.

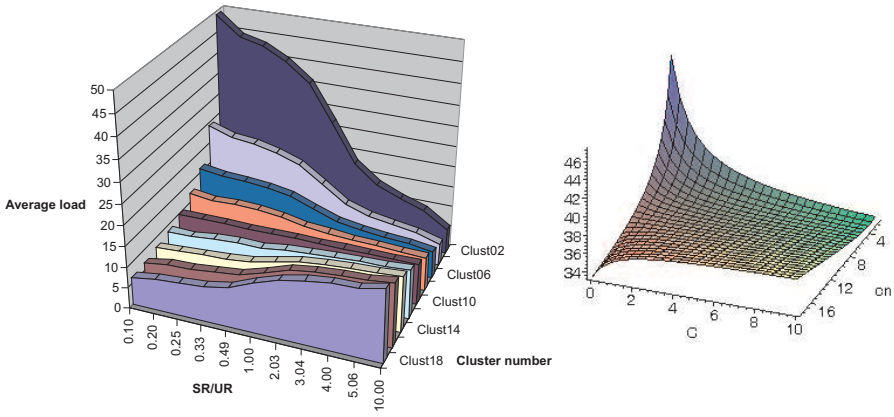

Fig. 5. Average network load in case of various SR/UR rates and cluster sizes a) based on simulations b) calculated with Maple

When SR/UR is equal in all nodes, which means that the users behave equally, the optimal cluster size can be derived easily. The following notations are used: a is the average load of a node, s is search requests (in the system), u is index update requests, n is the number of the nodes in the network, r is the square root size, so $\sqrt{n}$, c_s is the (average) size of a cluster and c_n is (average) number of clusters.

In a PIC system with a uniform $\frac{s}{u}$ (SR/UR) ratio the average load can be calculated as:

$$a = \left(\frac{s(c_n - 1) + u(c_s - 1)}{n} \right) \tag{1}$$

because the search requests are forwarded to all other groups and the index updates are forwarded to all other members of the cluster.

We can also assume that the requests in the system grow linearly with the

number of nodes present in the system, so:

$$s + u = Kn \tag{2}$$

(In our simulations K was 10.)

With naming the SR/UR ratio to C

$$C = \frac{s}{u} \tag{3}$$

and using that

$$c_s c_n = n \tag{4}$$

one can find based on (1) that

$$a \approx \left(\frac{KCr}{C+1}\right)\left(\frac{r}{c_s} + \frac{c_s}{Cr}\right)$$

and the optimal cluster size is

$$c_{s_{opt}} = r\sqrt{C}$$

The square root law can also be derived from (1) and (2).

When the SR/UR ratio is equal only in each group, with a little more complicated derivation the same minimum value can be found for the cluster size.

4.2. Adopting Clusters

As we have the network traffic as the function of the SR/UR ratio and cluster size, every node can calculate this and can choose the optimal node distribution, if the SR/UR ratio is measured. So the change in the SR/UR ratio is handled by the network with adapting to it and rearranging the clusters. Simulations are also made to see how often clusters have to change for following the changing of the user behavior, but not to generate too much traffic with the adaptation.

Using an adopting algorithm, the $AvgLoad$ can be reduced reasonably where the SR/UR ratio was very high or very low. Simulation results can be seen on *Fig. 6*.

4.3. Network Load Comparison

It is hard to compare the PIC network to other P2P networks, because its main usage area is not usual: as described in section 1.4, our goal was to achieve 100% hit ratio in a network with moderate size, and in a closed user group environment. Though, it can be useful to compare the generated network load in a PIC network to a commonly used protocol, to be able to see its usability in the aimed area.

As an example, we have compared the network load generated by PIC to a Gnutella network described in [1]. As 100% hit ratio practically can not be achieved in a Gnutella network, 95% hit ratio was chosen based on [1]. We can see in [1], that the best analyzed network topology (i.e. with lowest bandwidth needs) fulfilling this hit ratio criteria (Semi-Random Mesh, with a node degree of 2.8) needs approximately 570 packets to be sent per query with the introduced optimization method called "Adaptive TTL", and 910 with the original Gnutella implementation. In contrast the PIC network needs $\sqrt{n} - 1$ packets to be sent per request in a square root cluster sized network, which means 31 packets for either a search or an update request in a 1024 node network. This can be further reduced if SR and UR is not equal, e.g. by adopting the cluster size dynamically like in 4.2 (as we can see on *Fig. 6*).

Although the previous calculations show very promising results, we can only conclude that if our needs are close to the ones described in 1.4, the PIC network is a very good candidate to be used. If somebody wants e.g. to handle very large networks, but the 100% hit ratio is not needed at all, then probably there are better solutions for that purpose.

5. A P2P Application Using PIC

In collaboration with the Nokia Research Center Budapest, we have developed a demonstration application using the PIC protocol. The application was written in Java (MIDP 2 profile for wireless devices). The test has been performed using J2ME Wireless Toolkit (WTK) 2.0 and Nokia Developer's Suite for J2ME 2.0 (NDS). Recently it was also successfully demonstrated on the Nokia 6600 platform.

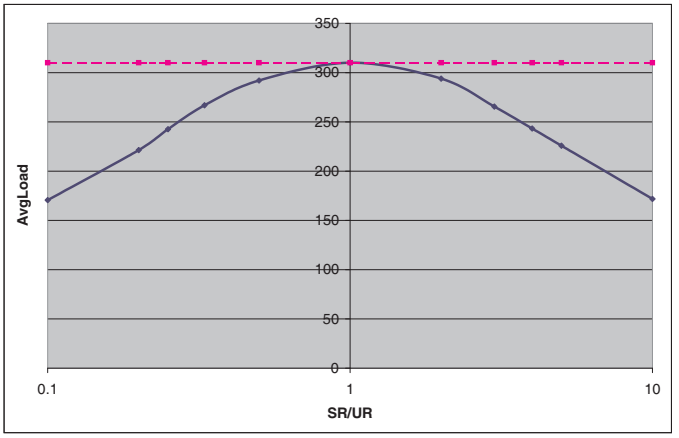

Fig. 6. Average network load for adopting clusters (straight) compared to square root sized network (dashed) in a 1024 node network

For index update topology the deterministic Ring topology has been selected, because it was easy to implement, balances index update load well and unexpected disconnections from the network could be easily detected by co-cluster nodes. The NSLs for the queries are implemented using acknowledged UDP. The registration process to the user group is done via a separate TCP connection. The initial IP address discovery is solved using a separate technology (patent pending).

Fig. 7. NDS emulator: main screen & network events, query, content handling, net status, image capture

Fig. 7 presents some of the capabilities of the application: creation and sharing of text files, pictures (support of sound is also a possible and planned feature), search for content and displaying of network information. The application runs on the mobiles without any central support, there is no need of any infrastructure beside the regular GPRS service offered by the GSM operator.

6. Conclusions and Further Work

In this article several types of peer-to-peer systems were analyzed, with special attention to the needs of a mobile environment. For a given realistic mobile peer-to-peer scenario the PIC protocol was introduced.

We have simulated PIC performance for uniform traffic and identified a number of topologies. We have shown that traffic is uniform for all SR/UR ratios in the case when the cluster size is the square root of the number of nodes. We have also developed a demo application using PIC and tested it in the real environment.

Further work will focus on four theoretical directions: identifying the optimal cluster size from the viewpoint of traffic minimization, including the link management traffic in the model, the analysis of non-uniform cluster size and non-uniform node traffic case; definition and analysis of node profiles, and the case of inactive nodes.

Acknowledgement

Thanks to Nurminen J. K., Bakos B. and Farkas L. from the SAM group in the Nokia

Research Center, and Charaf H. and Csúcs G. from the Applied Informatics group in the DAAI department in TUB. Special thanks to Lóránt Farkas and Gergely Csúcs.

References

[1] CSÚCS, G. – NURMINEN, J. K. – BAKOS, B. – FARKAS, L., Peer-to-peer Protocol Evaluation in Topologies Resembling Wireless Networks. An Experiment with Gnutella Query Engine, *ICON*, 2003.

[2] COOPER, B. F. - GARCIA-MOLINA, H., Modeling and Measuring Scalable Peer-to-peer Search Networks, *Proc. SIGCOMM*, 2002

[3] SCHOLLMEIER, R., A Definition of Peer-to-Peer Networking for the classification of Peer-to-Peer Architectures and Applications, *P2P*, 2001

[4] ORAM, A.(edited by), Peer-to-Peer – Harnessing the Power of Disruptive Technologies, *O'Reilly*, 2001

[5] SCHMIDT, R., Gridella: an open and efficient Gnutella-compatible Peer-to-Peer System based on the P-Grid approach, *EPFL Technical Report*, 2002

[6] STOICA, I. - MORRIS, R. – KARGER, D. – KAASSHOEK, M. F. – BALAKRISHNAN, H., Chord: a scalable peer-to-peer lookup service for internet applications, *SIGCOMM*, 2001

[7] ABERER, K., P-Grid: A self-organising access structure for P2P information systems, *CoopIS*, 2001

[8] GOLD, R. - TIDHAR, D., Towards a Content-based Aggregation Network, *P2P*, 2001

[9] SHIN, K. – LEE, S. – LIM, G. – YOON, H. - MA J. S., Grapes: Topology-based Hierar-chical Virtual Network for Peer-to-peer Lookup Services, *ICPPW*, 2002

[10] MAROSSY, K. – CSÚCS, G. – NURMINEN, J. K. – BAKOS, B. – FARKAS, L. Peer-to-peer content sharing in wireless networks, *PIMRC*, 2004 (submitted, accepted)

[11] MAROSSY, K. - CHARAF, H., Peer-to-Peer Systems in Mobile Environment, *MicroCAD*, 2004