Overlay Bandwidth Management: Scheduling and Active Queue Management of Overlay Flows

Kálmán Graffi, Konstantin Pussep, Sebastian Kaune, Aleksandra Kovačević, Nicolas Liebau, and Ralf Steinmetz
KOM - Multimedia Communications Lab, Technische Universität Darmstadt
Merckstr. 25, 64283 Darmstadt, Germany
Email: {graffi,pussep,kaune,sandra,liebau,steinmetz}@kom.tu-darmstadt.de

***Abstract*— Peer-to-peer and mobile networks gained significant attention of both research community and industry. Applying the peer-to-peer paradigm in mobile networks lead to several problems regarding the bandwidth demand of peer-to-peer networks. Time-critical messages are delayed and delivered unacceptably slow. In addition to this, scarce bandwidth is wasted on messages of less priority. Therefore, the focus of this paper is on bandwidth management issues at the overlay layer and how they can be solved. We present HiPNOS.KOM, a priority based scheduling and active queue management system. It guarantees better QoS for higher prioritized messages in upper network layers of peer-to-peer systems. Evaluation using the peer-to-peer simulator PeerfactSim.KOM shows that HiPNOS.KOM brings significant improvement in Kademlia in comparison to FIFO and Drop-Tail, strategies that are used nowadays on each peer. User initiated lookups have in Kademlia 24% smaller operation duration when using HiPNOS.KOM.**

Keywords: P2P, overlay, bandwidth, scheduling, AQM, QoS

I. Introduction

The influence of the Peer-to-Peer (P2P) communication paradigm increased over the last years. P2P solutions provide a feasible strategy to balance load in a large scale system. Another trend in communication networks is the mobility of the users and their network devices. Cell phones, PDAs and mobile computers are common and widely in use. These devices, if participating in P2P networks, introduce a wide variety of device capabilities and connection types. Most of the current research in P2P network [1] neglects the effects of peers' bandwidth characteristics, although bandwidth is considered as the scarcest resource in the network. In [2] it is shown that even overprovisioning does not help.

In fact, the strategy of bandwidth utilization can be crucial, as the breakdown of Gnutella [3] in 2002 shows (see [4]). In Gnutella an exponential number of messages is produced which the network was incapable to process, resulting in congestion and the collapse of the network. In future, Next Generation Networks (NGN) are evolving combining various types of networks with an all-IP paradigm. Having a wider range of devices with diverse connection types in the system, solutions are needed to manage bandwidth and to handle congestion in P2P overlays.

Peer-to-Peer applications are build according to the architecture presented in Figure 1. An overlay is created and maintained by periodically exchanging messages with other peers in the overlay. Furthermore the overlay provides specific services that can be used by user applications. Common services are lookup, search and store operations. However, once a communication partner is identified in the network using the overlay operations, direct P2P communications is initiated. In the case of limited bandwidth capacity, various strategies can be applied in order to limit the negative effects of traffic peaks.

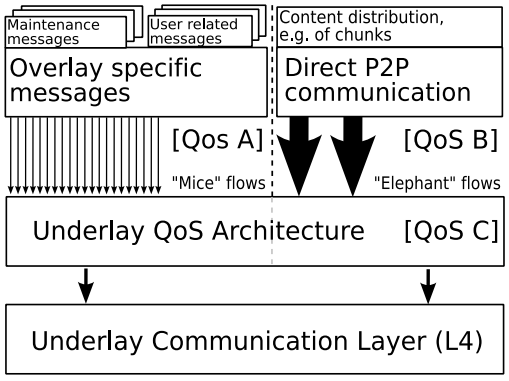


Fig. 1. Layers of the P2P QoS architecture.

Bandwidth limitations are likely to occur, because of two reasons. Download bandwidth is often larger than upload bandwidth, so that peers tend to retrieve more data than they can send out. Furthermore computing power of network devices is assumed to be sufficient, incoming traffic can be processed fast enough. Thus, the output link remains as bottleneck, congestion may occur.

In order to provide quality of service (QoS) to both kind of communication (overlay-specific and direct P2P communication), we investigated on the characteristics of corresponding flows. Direct P2P communication is used for long-term transmission (*elephant* flows), e.g. of chunks. In contrast to this, P2P overlays contain only short-term flows (*mice* flows), with only few messages exchanged with every contact.

Furthermore, Esten et al. affirm in [5] through thorough measurement of P2P traffic this observation. In order to manage the available bandwidth, we present a classification of (quasi) quality of service providing mechanism.

The field denoted by (QoS B) contains strategies to manage the direct P2P communication. BitTorrent [6] for example decides which peers to serve with chunks. This has effect on the scheduling of the *elephant* flows. As *elephant* flows are steady background traffic, several approaches exist to provide lower than best effort service [7] or alternative best effort

0742-1303/07 $25.00 © 2007 IEEE
DOI 10.1109/LCN.2007.13

IEEE computer society

[8] service. These solutions can be applied at lower layers, at (QoS C) in Figure 1.

What remains open and is uncovered in P2P research is the provision of QoS for overlay-related messages, for the *mice* flows. This research field is located at the point (QoS A) in Figure 1. Providing guarantees to overlay flows is challenging and cannot be done on the underlay (QoS C), as the content and relevance of the overlay messages have to be known.

Scheduling and Active Queue Management (AQM) mechanisms are widely discussed in the research community. In this paper we investigate how the solutions for the network layer can be applied in P2P overlays, i.e. on the application layer. We show that solutions based on constant message flows cannot be applied directly, as the concept of data flows is not applicable in P2P overlays. Further, we show that scheduling and AQM solutions that are independent of flows can significantly increase the quality of service provided by the overlay. We introduce multi-dimensional message priorities and show that priority-based schedulers and AQM mechanisms can give guarantees with respect to transmission delays and loss avoidance.

We implemented both a First-In-First-Out (FIFO) and Drop-Tail based buffering strategy and a priority-based strategy called HiPNOS.KOM (Highest Priority First, No Starvation) in PeerfactSim.KOM [9], a simulator for large-scaled P2P systems. PeerfactSim.KOM focuses on multi-layer interdependencies in P2P systems instead of isolated overlay evaluation. The effects of the buffer strategies have been evaluated for Kademlia [10] in the version of [11]. We show that using HiPNOS.KOM, P2P systems can provide guarantees in terms of lower delays and lower drop ratios for higher prioritized messages. In both cases the average hop count for messages in the overlay was between 6.3 and 6.48. The hop count was independent of the used bandwidth management mechanism.

The operation duration of user initiated lookups is decreased by 24% when using HiPNOS.KOM in comparison to the state-of-the-art FIFO mechanism. This performance gain comes with very low computational costs for performing scheduling calculations.

The paper is structured as follows: in Section II we summarize the problem statement and present criteria that are relevant to measure the quality of solutions. In Section III we present our investigations on the premises needed to apply scheduling and AQM mechanisms in P2P networks. We present HiPNOS.KOM that provides QoS guarantees on the processing of prioritized messages. In Section IV we present the evaluation setup we used and motivate the metrics. The evaluation of our assumptions shows the significant beneficial effects of bandwidth management in P2P systems. In Section V we briefly discuss solutions presented in literature that deal with efficient bandwidth utilization in P2P networks. We also present in that section scheduling and AQM mechanisms that have been proposed for other network layers. Finally, we conclude in Section VI that the proposed priority-based scheduling and AQM strategies increase the QoS provided by the P2P overlay.

II. Problem Statement

In this section, we present a simple network model in order to describe the problem of bandwidth management and show criteria which give us insights in the status of the system. We consider a network of N peers, each peer p has a download bandwidth capacity of D_p and an upload bandwidth capacity of U_p. Messages in the P2P overlay are either periodically generated for maintenance reasons or related to the user's behavior; we name these sets M_m and M_u. A message m is considered to have a message-type specific size m_s. There exist three cases in which a peer has to transmit a message: when the peer is initiating the message, forwarding the message or replying to the message. Let r_i^p, r_f^p and r_r^p be the rates for messages that are initiated, forwarded and replied by peer p. Please note that the messages may be either element of M_m or M_u.

Congestion occurs if the bandwidth required for transmitting the messages of $M_m \cup M_u$ exceeds the peer's available bandwidth. The amount of upload bandwidth required by a peer p is $B_u^p = r_i^p m_s + r_f^p m_s + r_r^p m_s$. Note that in order to simplify the formula we model the message-type related message size only with one parameter, the average message size m_s. The main aspects in the formula are the three message rates.

The parameter r_i can be adapted by each peer individually, r_f is closely related to the complexity of the overlay operations. For example a search operation in Gnutella would cause $O(N)$ nodes out of N nodes forwarding the query. A lookup operation in Chord causes only $O(log(N))$ peers to forward the lookup message. Finally, r_r is mainly influenced by the topology, i.e. the number of contacts a peer has, and by the popularity of the content the peer stores. It is obvious that if $B_u^p > U_p$ holds peer p is constantly congested.

In this context we identified two problems that have to be addressed. In the case when congestion occurs, messages cannot be sent and are stored in the buffer of the congested peer. When bandwidth is available again the peer can choose which message to transmit next, this process is called *scheduling*.

The second problem that needs a solution is given by the limitation of the buffer size in each peer. If the transmission rate of a peer is smaller than the arrival rate of new messages, the size of the buffer increases constantly. Due to the limited buffer size in reality the peer has to choose which packets to drop in case of buffer overflow. This problem is called *active queue management* (AQM).

We identified two criteria that are relevant when processing messages.

1) Delay - The variety of message types used in overlays comes with a variety of demands on the delay of message transmission. Whereas rarely sent maintenance messages are not time critical, user related communication is expected to be processed as fast as possible.
2) Loss - Various message types have differing relevance for the functionality of the overlay network. Messages that are delay critical or tightly coupled to user request

should not be dropped in case of congestion. Optional messages or messages of less interest should be considered for dropping.

III. Solution

In this section we present our solution for the scheduling and AQM problem in P2P overlays. We demonstrate that flows in the common sense do not exist and therefore messages should be grouped by message priorities. Furthermore, we introduce a static priority scheduler and AQM mechanism called HiPNOS.KOM that provides guarantees in context of delay and loss. In the following, we point out in which layer our solution is to be placed.

A. Placement on Layer Model

Current research on P2P overlays applies a layer model consisting of the overlay layer, which provides all P2P related intelligence, communicating directly with the transport layer of the ISO/OSI model. Dabek et al. introduced in [12] the *Key-Based Routing* layer (KBR), which is placed between the two layers described above. Its main task is to translate the peer IDs used in the overlay layer to IP addresses and ports used in the transport layer. Furthermore, it provides a pool of possible IP contacts that may be used as contacts in the overlay layer. Still KBR does not take any action in processing of messages passed between the other two layers.

For that reason we introduce a new layer below KBR, which we call the *Network Wrapper*. Scheduling and AQM mechanisms for overlay messages are placed in this layer, as it interacts tightly with the transport layer. The content of messages is not relevant to the Network Wrapper as only QoS information regarding scheduling and AQM are of interest for this layer. This QoS information should be passed via meta-data. Furthermore the Network Wrapper provides information to the KBR layer about which contacts on the transport layer are congested. By this, the KBR layer can focus on the selection of the most appropriate peers from the pool of contacts for the overlay layer. In Figure 2 we show the layer model containing the KBR and Network Wrapper layer.

Having identified the layer of interest we present in the next subsection the results of our investigations on message flows.

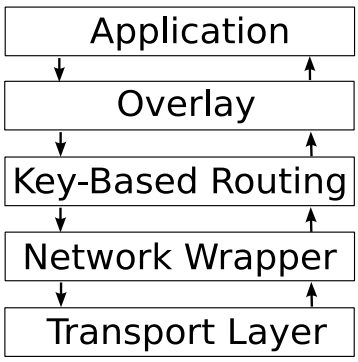


Fig. 2. The P2P layer model including the Network Wrapper layer for overlay independent bandwidth management

B. The Term Flow in Context of P2P Overlays

In order to use scheduling and AQM mechanisms in P2P systems we researched what kind of *flows* in P2P overlays exist, and whether flow-based mechanisms to handle bandwidth can be applied. In research on queue management the term *flow* describes periodically occurring events or messages initiated by a known instance that need to be processed. For CPU cycle scheduling and message scheduling between known endpoints it is easy to define flows. In CPU cycle scheduling jobs related to a single process are defined as a flow. In packet scheduling packets related to a communication path between specific end points are defining flows. In contrast to this, in P2P networks a flow is hard to define, as it is needed that events related to a flow are periodically occurring. Traffic in P2P overlays is very dynamic. Each peer can initiate key-based search or DHT-based lookup queries for any object stored in the network. The repliers of the queries are typically not known in advance.

Concluding, we state the hypothesis that the number of requester-replier pairs in the overlay is large and that a specific combination is not reoccurring periodically. Overlay traffic consists only of *mice-flows*.

We show in Section IV that peers in Kademlia have in average a high number of contacts in the P2P overlay, but the number of messages per contact is significantly small. Therefore we conclude that messages from one specific initiator are not periodically received and the term *flow* in its common sense cannot be applied to P2P overlays.

This observation causes that solutions relying on the existence of flows cannot be used to solve the scheduling and AQM tasks in P2P overlays. Packets cannot be grouped by their initiator or by their destination, only packet specific details can be used for grouping. Either the message type or individual (multi-dimensional) packet priority classes can be used. In our solution we use prioritized messages considering delay and loss requirements.

Before presenting our priority based scheduler HiPNOS.KOM, we subsume the design goals for our solution.

C. Requirements for a Solution

One important requirement for a solution on the overlay layer is that it should be independent of a specific overlay. This goal aims at the re-usability of the solution approach.

Message priorities are motivated by the diversity of applications that may use a P2P overlay. Numerous message types exist in the implementation of an overlay, their relevance for the functionality of the system differs. In addition, the orders of the user may have differing relevance as well. In order to model the relevance of a message with respect to delay and loss a solution has to take into account multi-dimensional message priorities. We use the term *message class* to describe all messages with the same priority. We state in the following requirements the design goals of our solution.

The average queue delay $D_Q^{avg}(i)$ for messages of priority class i should be smaller than the average queue delay $D_Q^{avg}(j)$

of lower prioritized message classes j of the set of delay priorities P_D:

$$\forall i, j \in P_D \; with \; i > j \; : \; D_Q^{avg}(i) < D_Q^{avg}(j) \qquad (1)$$

Another requirement is that starvation of lower prioritized messages does not occur. Every message has to be processed in a reasonable time.

For the decision about which message to drop out of a full queue we stick to the claim that no message of importance should be dropped if a less important message exists in the queue:

$$\forall i, j \in P_L \; with \; i > j \; : \; Loss_{avg}(i) < Loss_{avg}(j) \qquad (2)$$

D. HiPNOS.KOM - High Priority First, No Starvation

From the requirements stated above we conclude HiPNOS.KOM, a priority scheduler for P2P networks. HiPNOS.KOM is placed in the Network Wrapper, managing the messages passed from the layers above for transmission using the layers beneath. Each peer is expected to have a buffer B to store messages for the case that the transmission channel is in use. Further we assume that a peer p can estimate its own available upload capacity U_p. There exist feasible mechanisms to achieve this task, several approaches are presented in [13]. Messages in the P2P system are marked with a two-dimensional priority value (P_D; P_L), characterizing their criticality regarding delay and loss. It is assumed that the upper layers mark these packets according to the relevance for the system. This assumption is valid, as higher layers know the semantics of the packets and are able to give additional information on it to the lower layers.

As a message $m_{P_D(m),P_L(m)}$ is passed from the higher layers to the Network Wrapper, HiPNOS.KOM follows the following strategy:

- Only if the buffer is empty and bandwidth is available, $m_{P_D(m),P_L(m)}$ is transmitted. Otherwise $m_{P_D(m),P_L(m)}$ is stored in the buffer.
- If the buffer contains messages and upload bandwidth is available the following steps are processed:
  1) The message with the highest delay priority $P_D(\cdot)$ is chosen.
  2) As tie-breaking rule the buffer-insertion time is considered in order to choose the message that is longer in the buffer.
- In order to avoid starvation of lower prioritized messages, the delay priority of all messages in B is increased every time unit Δt.

For active queue management HiPNOS.KOM uses a simple priority based mechanism. If the size of B reaches a predefined threshold value and a new message arrives, a message m^* is dropped. In order to calculate m^* let $AT(m)$ be the arrival time of message m in the buffer. We define the subset of messages B_{min} in the buffer with minimal loss priority as $B_{min} = m \in B$ with $P_L(m) = min\{P_L(x) \mid x \in B\}$. The message m^* that is dropped is then calculated by $m^* = m \in B_{min}$ with $AT(m) = min\{AT(x) \mid x \in B_{min}\}$.

Higher prioritized messages are always considered more relevant than lower prioritized ones. One may argue that in a congested scenario this may lead to the case that messages of specific message types with low loss priority are never processed. This case only leads to problems in the network when the relevance of the messages is estimated wrong on the higher layers. HiPNOS.KOM supports dynamic priority changes of messages types. Higher layers are assumed to modify the priority setting of messages that are passed to the Network Wrapper in order to adapt to the network characteristics. The Network Wrapper itself is incapable to determine the relevance of the semantics of messages.

HiPNOS.KOM is designed to provide service guarantees on message transmission to higher layers. The functionality of the overlay layer should not be thwarted by incapabilities of the layers below. This aspect is very important if a P2P network contains numerous devices with low bandwidth capabilities.

Regarding the complexity, HiPNOS.KOM can be implemented demanding O(1) processing time using hashmaps (to identify the queue per priority class) and calendar queues (for enqueue and dequeue operations regarding the arrival time of the messages). The storage demand of HiPNOS.KOM is O($|B| + max\{|P_D|, |P_L|\}$) where $|B|$ is the size of the buffer and P_D and P_L the set of priority classes regarding delay and loss. The buffer-size is limited with a predefined threshold.

In the following section we present the evaluation on the applicability of scheduling and AQM mechanisms in P2P overlays. Therefore we laid in Section III the foundation of bandwidth management in P2P systems. We identified that the term *flow* needs to be adopted for P2P overlays, as it is inapplicable in its common sense. We introduced message priorities and priority classes to model the requirements of messages in terms of delay and loss. HiPNOS.KOM is a simple scheduling and AQM mechanism that considers the characteristics of P2P networks and provides differentiated service for the priority classes. In the next section we show that our hypothesis on the inapplicability of the term *flow* in its common sense is true, and that HiPNOS.KOM has a significant beneficial effect on the processing of the messages at a very low cost.

IV. Evaluation

In order to evaluate our approach we chose PeerfactSim.KOM[1] [9], a large-scale P2P simulator. It comes with the already implemented P2P overlays Gnutella [3], Kademlia [11], Chord [14] and Globase.KOM [15]. We extended the simulator with the Network Wrapper that manages the bandwidth management and message transmissions, and various metrics described in the following subsection.

We compare the quality of HiPNOS.KOM to the reference strategies of nowadays: the FIFO scheduler and the DropTail AQM mechanism presented in Section V. Although these two mechanisms are very simple, they are state-of-the-art mechanisms to control the bandwidth utilization of outgoing

[1] http://www.peerfactsim.com

messages. We used Kademlia in the version [11] as P2P overlay in order to compare the effects of bandwidth strategies used in the Network Wrapper. Kademlia is currently one of the most used overlays in P2P applications.

A. Evaluation - Simulation Setup

In the following we present the simulated scenario and the metrics we used.

1) Metrics: We used following metrics to measure the quality of the system:

- The number of *contacts per peer and the number of messages per contact* gives us information on the applicability of the term *flow* in P2P systems.
- The metric *average delay per message priority (delay)* shows how the Network Wrapper supports the processing of relevant messages. Delay is measured hop-to-hop in the overlay.
- The metric *average loss rate per message priority (loss)* shows which message classes are dropped, when the transmission channel of the peer is congested.

2) Scenario: Our simulation setup consists of 10,000 peers with heterogeneous bandwidth capabilities. In [16] Saroiu et al. give a measurement study on the bandwidth capacities of peers in P2P overlay networks. We use the bandwidth distribution presented in Table I based on their work:

Fraction	Download capacity	Upload capacity
10%	64 kbps	64 kbps
15%	784 kbps	128 kbps
15%	2048 kbps	304 kbps
30%	3076 kbps	1024 kbps
20%	10240 kbps	2048 kbps
10%	20480 kbps	10240 kbps

TABLE I
THE UPLOAD/DOWNLOAD CAPACITY DISTRIBUTION

We limit the size of a peer's queue to 10 messages in order to investigate the effects of strategies handling congestion. All peers join at the beginning of the simulation. The joining phase is long enough to give each peer enough time to join (0.5 seconds per peer).

After joining each peer performs several store and lookup operations for randomly chosen objects. During the simulation time it is taken into account that peers may fail and churn exists. The user initiated message load M_u is defined by the parameter r_i^p. The size of the message load M_m is related to the overlay maintenance demanded by the chosen P2P overlay. Each overlay comes with a set of message types. We do not define for each of them a specific priority, but we give random priorities to each message individually. We do this in order to have messages with a wide range of priorities, so that the effects of the strategies implemented in the Network Wrapper can be analyzed in more detail.

We modified the bandwidth management strategies in our scenario, either using FIFO with Drop-Tail or HiPNOS.KOM. Each scenario is simulated 20 times so that we can use a confidence interval of 95%.

B. Evaluation - Results

In the following subsection we present the results of the simulations. We show that flows in common sense do not exist as only few messages are sent per contact. Further, we show that HiPNOS.KOM provide better service for higher prioritized messages in context of both delay and loss.

1) The Applicability of the Term Flow: In Figure 3, we examine the number of messages per contact in relation to the number of contacts each peer in Kademlia has. This figure shows us both, from how many peers a single peer receives messages and how many messages are received on average. We see that one single peer (out of N) has contact to a huge number of other peers. Messages could be grouped by their initiator, but this would result in O(N) flows with only few messages per flow. In our scenario the average number of contacts a peer has is 1437. Further we see that a peer approximately receives only 1.4 messages per contact. These simulation results match the traffic measurements of real systems presented in [5]. Discussions [5] considering the applicability of per-flow mechanisms in context of *mice* flows, come to the conclusion that it is infeasible to hold state for all flows. Considering these two observations, we state that maintaining a buffer per source-identified queue does not make sense as only very few messages are received from a huge number of contacts. Typically, source-destination pairs are used to group messages to flows. Using the source-destination pair of a message is not applicable in our case, as we get even more flows (O(N^2)), with even less messages each. The term $flow$ cannot be applied in its classical meaning to P2P networks. As mentioned in Section III we suggested to group messages by their priority classes.

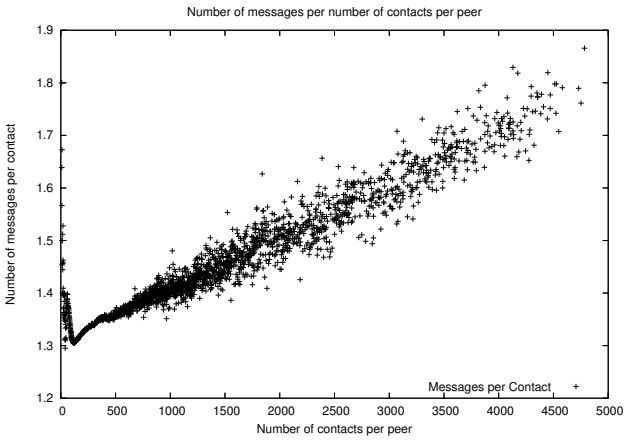


Fig. 3. Relation between number of contacts per peer and the number of messages per contact

2) Comparison of FIFO with Drop-Tail and HiPNOS.KOM: Figure 4 shows the performance of FIFO in combination with Drop-Tail and HiPNOS.KOM regarding delay in a P2P network with high traffic load. We use the bandwidth distribution presented in Table I.

Figure 4 shows the average end-to-end delay of different priority classes. We used one byte per priority (delay, loss), so that the range is from -128 to 127. The higher the number, the higher the priority. As FIFO and Drop-Tail do not consider priorities, the graphs corresponding to them are predominantly constant. The average delay of the messages processed with HiPNOS.KOM decreases linearly as the delay priority of the messages increases. In total, both approaches provide a similar overall average delay, but HiPNOS.KOM guarantees a faster processing of messages that are more relevant to higher layers. Here again, HiPNOS.KOM enables an additional functionality by fulfilling the delay related Equation 1 shown in Section III.

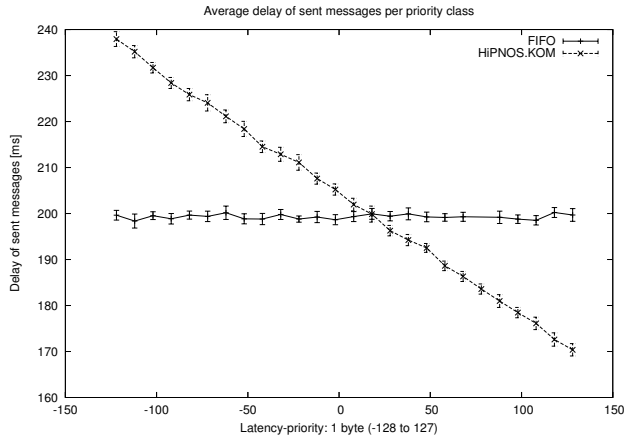


Fig. 4. Delay of message transmission in Kademlia in relation to the delay priority $P_D(m)$ of the message m.

In Figure 5 we examine the number of dropped messages in relation to the loss-priority of the messages. The high traffic load leads to the case that peers are not able to process every message, as there are more messages incoming or generated than messages can be transmitted. In our scenario approximately 3% of the messages are dropped. The Drop-Tail strategy treats every message class equally and drops loss-critical messages in the same amount as non-loss-critical messages. HiPNOS.KOM in contrast drops the messages according to their priorities. The number of messages dropped decreases with the increase of the loss-priority value. We observe that HiPNOS.KOM approximates the loss related Equation 2 presented in Section III, which models the ideal case. As we mentioned, this aspect is very relevant in real P2P systems. In critical situations like in the join process messages should not be dropped. Using HiPNOS.KOM the Network Wrapper can provide guarantees that highly important messages are only dropped, when there is no other way.

C. Benefits in the System Performance

We compared the operation duration in a P2P system using FIFO and HiPNOS.KOM. We assigned the highest delay and loss priority to all value lookup and reply messages in the system. The other messages were marked with low priorities. The total operation duration for user initiated actions, like

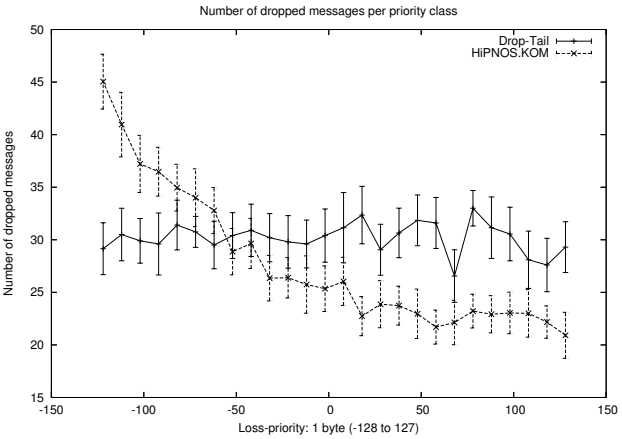


Fig. 5. Number of dropped messages in Kademlia in relation to the loss priority $P_L(m)$ of the message m.

value lookup, was while using FIFO 1.684 seconds. In contrast to the average operation duration of 1.282 seconds when using HiPNOS.KOM. By applying scheduling and AQM mechanisms on application layer the performance of the system could be improved by 24% with minimal additional costs.

V. Related Work

In this section we discuss the state of the art solution for the scheduling and AQM problem and give a brief overview on other solutions presented in literature.

State of the art P2P overlay implementations do not consider the problems occurring from bandwidth congestion. A common approach to implement P2P overlays is to focus on the overlay layer and to leave details on message transmission for the lower layers. Typically messages are created in the application and passed to the network handler of the operating system. TCP provides congestion avoidance strategies, but does not consider message priorities. In order to provide guarantees in terms of delay and loss, mechanisms have to be implemented tightly interacting with the overlay layer.

Resulting from the arguments above, the state of the art mechanisms used for queue management on overlay layer are First-In-First-Out (FIFO) and Drop-Tail, we present both in the following.

First-In-First-Out Scheduling: The First-In-First-Out principle processes the incoming messages ordered by there arrival time. Its implementation complexity is O(1). Coming with its simplicity FIFO is the dominant scheduling mechanism currently used in P2P overlay implementations. Messages are passed from the overlay layer to the transport layer and transmitted when bandwidth is available. It is obvious that this strategy does not provide any guarantees on the delay of a message transmission.

Drop-Tail AQM: The simplest way to handle network congestion is called Drop-Tail. It means that new packets are enqueued as long as there is place for them in the queue, which is limited in length. The queue may get full because the sending rate of the output link is smaller than the arrival rate

at the input link. In this case new packets are dropped. Drop-Tail does not provide any guarantees that highly prioritized messages are transmitted.

In contrast to these two simple mechanisms other more technically mature mechanisms exist. In [17] we analyzed twenty-two scheduling mechanisms discussed frequently in the literature. Our investigation shows that the majority of the scheduling approaches assume the existence of message flows defined by sender-destination pairs. As we have shown in Subsection III-B flows in this sense cannot be assumed in P2P networks. In addition to the taxonomy on scheduling mechanisms we analyzed in [18] fifteen common AQM approaches. Our taxonomy on these AQM mechanisms reveals that many of them rely on flows as well.

Based on the taxonomies stated in [17] and [18] we present in the following the scheduling and AQM solutions that are independent of sender-destination pair based flows.

A. Related Work on Scheduling and AQM Mechanisms

For ease of presentation the following overview on existing solutions adopts the terminology to P2P overlays, although the original papers had been proposed for another field of application. In general we changed the term *flow* to *message class*, which describes the set of existing message types or message priorities in the system.

1) Scheduling using Fair Queuing (Round Robin): Nagle proposed 1987 in [19] a simple scheduling mechanism for packet switches in which each flow is assigned to a queue of its own. Adapted to P2P overlays this means that to each message (priority) class a queue of its own is assigned. Messages are transmitted using the round robin principle to choose the next queue to be serviced. This approach can be implemented very efficiently, as no further computation is needed. Round Robin is a feasible solution for providing fairness among several message classes. However, if message priorities have to be considered, additional control parameters, like weights, need to be introduced.

2) Scheduling using Weighted Round Robin: Classical round robin provides an equal share of service to all message classes in the system. Weighted Round Robin (WRR), presented in [20] by Katevenis et al., introduces for each class i weights w_i, which define the amount of share they receive. The round robin share for each class i is $\frac{w_i}{\sum_{j \in F} w_j}$; this is also the fraction of the total service provided for message class i. This approach is capable of considering prioritized messages by adopting the share of service a message class i receives to its priority. Congestion may occur, but each message class receives a certain share of bandwidth. The drawback of this approach is that it only controls the maximum rate of service a specific message class receives. WRR controls how much throughput is guaranteed to a class. This is independent to the requirements of having low delays.

3) Scheduling using Core-Stateless Fair Queuing: Core-Stateless Fair Queuing (CSFQ) is introduced in [21] by Stoica, Shenker and Zhang. Their main goal is an efficient fair queuing algorithm with strong complexity reduction. This is achieved by introducing two types of devices: edge and core routers. Core routers are surrounded by edge routers so that all traffic coming from the rest of the network has to pass an edge router before coming to a core router. Edge routers estimate the traffic at the edge of this network island and label packets with the rate of their flows. Core routers use these labels to calculate a minimum service rate for all flows. Upon congestion packets that exceed a specific threshold above the minimum service rate are dropped.

An extension to CSFQ is Weighted CSFQ [21]. Each flow i is assigned a weight w_i that has impact on the share the message class receives. The higher the weight of a class, the smaller the probability that packets of this class are dropped. Flow i with weight w_i receives in the time interval $[t_1, t_2]$ not more share than $w_i \cdot \alpha \cdot (t_2 - t_1)$, where α is calculated dynamically as the maximum service share for all classes.

A further improvement of CSFQ presented in [22] is Self-Verifying Core-Stateless Fair Queuing (SV-CSFQ). The authors argue that the concept of having edge and core routers is not applicable, because it is infeasible to isolate an island of core routers by surrounding them with edge routers. Therefore, they suggest in [22] to use only one kind of routers, which periodically check the validity of packet labels. In case of inappropriate labels, packets are relabeled and the service rate is adapted. Please note that in P2P systems it is possible to have an island of core routers. Each peer is an core router as it has to forward messages for other nodes, but also an edge router as it may initiate overlay specific actions.

4) AQM using Random Early Detection: Sally Floyd and Van Jacobson present in [23] a mechanism called Random Early Detection (RED) that aims at congestion avoidance. Their work is motivated by the goal to keep average queue sizes in routers small. This is done by dropping or marking packets with a probability related to the position of the messages exceeding a certain threshold in the queue. System-wide parameters Q_{min} and Q_{max} define the threshold boundaries of the queue size. Q_{min} defines the minimum queue length at which no packets are dropped, as the weighted average queue size Q_{avg} exceeds Q_{min} the dropping probability increases with increasing Q_{avg} and $count$, up to the maximum dropping probability P_{avg}^{max}. $count$ is the number of packets since the last dropped packet of the same flow.

There exists a wide range of AQM mechanisms based on RED. ATM-RED [24] takes the characteristics of ATM networks into account. Adaptive-RED [25] adapts the target queue length to meet delay and throughput requirements. Stabilized-RED [26] considers the bandwidth share of the flows in order to increase the diversity of flows in the queue. Fair-RED [27] measures the utilization of bandwidth per flow in order to impose on each flow a loss rate that is related to its bandwidth utilization. RED with Preferential Dropping [28] maintains a dropping history in order to identify flows that utilize bandwidth excessively. Flows with a high number of previously dropped packets are preferred for dropping. The main idea of *Choose and Keep Packets from Responsive Flows* (CHOKe) [29] is to compare an arriving packet with n random

packets in the queue. All randomly picked packets having the same flow identifier like the arriving packet are dropped. If they differ, a strategy similar to RED is used. Exponential-RED [30] uses an exponentially increasing dropping probability. This is done by using a primal-dual algorithm, known from optimization theory, in order to compute the optimal dropping parameters for RED.

B. Related Work on Bandwidth Management in Peers

In this subsection we discuss approaches on managing bandwidth in P2P networks that are discussed in literature. However, related work on this topic is hard to find. Hoßfeld et al. observe P2P systems in networks with limited bandwidth capabilities like UMTS [31] and GSM with GPRS [32]. They focus mainly on UMTS and GPRS specific issues and not on issues arising for P2P networks resulting from peers with limited bandwidth.

Some investigations on bandwidth issues in P2P overlay multicast trees have been presented in [33], [34], [35], and [36]. The focus in these papers is on scheduling of multimedia streams in P2P networks. P2P multimedia streaming uses scheduling to decide which peer shall receive the next chunk of data. These data distribution strategies are applied on top of the overlay layer. They differ from the assumptions and requirements stated for the Network Wrapper layer. Therefore, the listed approaches cannot be applied for our problem statement.

Chawathe et al. present in [37] numerous enhancements for Gnutella in order to improve its quality, especially with respect to scalability. The authors consider issues arising from congestion of the network and overloaded peers. Original Gnutella does not scale as queries are flooded through the network and therefore the message load is exponential in relation to the number of peers in the system. The authors suggest improvements on the search algorithm of Gnutella and besides that, a flow-control mechanism based on tokens. Each peer should generate tokens in the rate it can process query messages. These tokens are propagated to the peer's neighbor nodes. Each query that comes from such a peer requires the sending of a token as well in order to be processed. By adapting the rate by which tokens are generated a peer can control the number of queries it has to process. This solution proposed in [37] is well applicable in unstructured P2P networks, as user generated messages dominate the traffic load and maintenance messages are rare. In structured P2P networks maintenance messages are dominating the overlay traffic load, so that controlling the number of messages a peer is willing to receive, may have undesirable effects. However, the solution provides only a mechanism to reduce the incoming traffic, but no further differentiation on the priority of incoming messages. Our problem statement and solution focuses on the control of the outgoing traffic.

VI. Conclusion

This paper identifies that scheduling and bandwidth management are necessary for current and upcoming P2P overlays in order to preserve the functionality of the P2P system in networks with low average bandwidth capabilities. The issues we observed for congested P2P networks are that time-critical messages may be delayed, delivered unacceptably slow and scarce bandwidth is wasted on messages that are not important. The trend for mobility and ubiquitous computing leads to a wide range of small devices with in general limited bandwidth capabilities. Network congestion can occur in these cases easily.

We proposed the Network Wrapper layer which is located directly above the transport layer and manages the outgoing message queue of each peer in order to apply specific scheduling and AQM strategies for P2P overlay traffic. We show that message flows in the common sense do not exist in the overlay and therefore common scheduling and AQM approaches cannot be applied directly.

Further, we introduced message priorities and HiPNOS.KOM: a scheduling and active queue management mechanism following a *Highest Priority first, No Starvation* policy. HiPNOS.KOM enables the Network Wrapper layer to provide assertions to higher layers that the importance of messages is considered. The statements defined by Equation 1 and 2 express that more important messages receive in any case better quality of service than less important messages. In the Section IV we have shown that these equations are fulfilled for HiPNOS.KOM in Kademlia. This relevant functionality comes with a low computation complexity for HiPNOS.KOM. The FIFO and Drop-Tail strategy provides for all messages in the system the same quality which is not beneficial for the functionality of the overlay.

We laid in Section III the foundation of bandwidth management for P2P overlay traffic. We identified that the term *flow* needs to be adopted for P2P overlays, as it is inapplicable in its common sense. We introduced message priorities and priority classes to model the requirements of messages in terms of delay and loss. HiPNOS.KOM is a simple scheduling and AQM mechanism that considers the characteristics of P2P networks and provides differentiated service for the priority classes. Due to the separation of the network layers from the overlay layer by introducing the Network Wrapper, the solution presented in this paper is applicable for any implementation of a P2P overlay. HiPNOS.KOM provides a mechanism to cope with traffic overload in a way that is best for the overlay.

In the future we want to focus on the development of dynamic strategies to determine priorities for overlay message classes according to the state of the network. With this we look forward to increase the robustness of overlays. Our second research goal in future is to identify security issues of and find solutions to the problem that (misbehaving) peers can choose priorities for their messages by their own.

Acknowledgment

The authors would like to thank Kyra Wulffert for helping on implementing the simulations.

REFERENCES

[1] K. Graffi et al., "Peer-to-Peer-Forschung - Überblick und Herausforderungen," *it-P2P*, vol. 46, no. 3, 2007.

[2] K. Pitter, K. Kelly, and M. Quiner, "Battle of the bandwidth," in *SIGUCCS '04: Proc. of the 32nd annual ACM SIGUCCS conference on User services*. New York, NY, USA: ACM Press, 2004, pp. 1–3.

[3] R. Fernando, A. Bordignon and G. H. Tolosa, *Gnutella: Distributed System for Information Storage and Searching. Model Description*, http://www.gnutella.co.uk/library/pdf/paper_final_gnutella_english.pdf, 2000.

[4] Gnutella: To the Bandwidth Barrier and Beyond, http://www.xml.com/pub/r/662, 2000.

[5] C. Estan and G. Varghese, "New directions in traffic measurement and accounting: focusing on the elephants, ignoring the mice," *ACM Trans. Comput. Syst.*, vol. 21, no. 3, pp. 270–313, 2003.

[6] B. Cohen, "Incentives build robustness in bittorrent," in *Workshop on economics of peer-to-peer systems*, 2003.

[7] R. Bless, K. Nichols, and K. Wehrle, "RFC 3662: A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services," 2003.

[8] P. Hurley, J. Boudec, P. Thiran, and M. Kara, "ABE: providing a low-delay service within best-effort," *IEEE Network*, vol. 15, no. 3, 2001.

[9] A. Kovačević et al., "Benchmarking Platform for Peer-to-Peer Systems," *it-P2P*, vol. 46, no. 3, 2007.

[10] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *IPTPS*, 2002.

[11] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," http://pdos.csail.mit.edu/petar/papers/maymounkov-kademlia-lncs.ps.

[12] F. Dabek et al., "Towards a Common API for Structured Peer-to-Peer Overlays," in *IPTPS*, 2003.

[13] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *ACM SIGCOMM IMC*, 2003.

[14] I. Stoica et al., "Chord: A scalable peer-to-peer lookup service for internet applications," in *ACM SIGCOMM*, 2001.

[15] A. Kovačević, N. Liebau, and R. Steinmetz, "Globase.KOM - A P2P Overlay for Fully Retrievable Location-based Search," in *IEEE P2P*, 2007.

[16] S. Saroiu, P. K. Gummadi, and S. Gribble, "A Measurement Study of Peer-to-Peer File Sharing Systems," in *SPIE MMCN*, 2002.

[17] K. Graffi, N. Liebau, and R. Steinmetz, "Taxonomy of Message Scheduling Strategies in Context of Peer-to-Peer Scenarios," Technische Universität Darmstadt, Germany, Tech. Rep., 2007.

[18] K. Graffi, K. Pussep, N. Liebau, and R. Steinmetz, "Taxonomy of Active Queue Management Strategies in Context of Peer-to-Peer Scenarios," Technische Universität Darmstadt, Germany, Tech. Rep., 2007.

[19] J. Nagle, "On Packet Switches with Infinite Storage," *IEEE Trans. Comm vol. 35, No.4*, Apr. 1987.

[20] Katevenis M. and Sidiropoulos C. and Courcoubetis C., "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, 1991.

[21] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: achieving approximately fair bandwidth allocations in high speed networks," in *ACM SIGCOMM*, 1998.

[22] I. Stoica, H. Zhang, and S. Shenker, "Self-verifying CSFQ," in *INFOCOM*, 2002.

[23] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, 1993.

[24] V. Rosolen, O. Bonaventure, and G. Leduc, "A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic," *SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 3, pp. 23–43, 1999.

[25] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive red: An algorithm for increasing the robustness of red's active queue management," ICSI Center for Internet Research, Tech. Rep., 2001.

[26] T. J. Ott, T. V. Lakshman, and L. H. Wong, "SRED: Stabilized RED," in *INFOCOM*, 1999.

[27] D. Lin and R. Morris, "Dynamics of Random Early Detection," in *SIGCOMM*, 1997.

[28] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling high-bandwidth flows at the congested router," ACM ICNP, 2001.

[29] R. Pan, B. Prabhakar, and K. Psounis, "CHOKe - a stateless active queue management scheme for approximating fair bandwidth allocation," in *IEEE INFOCOM*, 2000.

[30] S. Liu, T. Basar, and R. Srikant, "Exponential-RED: a stabilizing AQM scheme for low- and high-speed TCP protocols," *IEEE/ACM Trans. Netw.*, vol. 13, no. 5, pp. 1068–1081, 2005.

[31] T. Hoßfeld, K. Tutschku, and F.-U. Andersen, "Mapping of File-Sharing onto Mobile Environments: Enhancement by UMTS," in *IEEE MP2P'05, in conjunction PerCom'05*, Mar. 2005.

[32] T. Hoßfeld, K. Tutschku, and F.-U. Andersen, "Mapping of File-Sharing onto Mobile Environments: Feasibility and Performance of eDonkey with GPRS," in *IEEE WCNC*, 2005.

[33] A. Bharambe et al., "The Impact of Heterogeneous Bandwidth Constraints on DHT-Based Multicast Protocols," in *In The Fourth International Workshop on Peer-to-Peer Systems*, 2005.

[34] R. Rejaie and A. Ortega, "Pals: peer-to-peer adaptive layered streaming," in *ACM NOSSDAV*, 2003.

[35] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Supporting heterogeneity and congestion control in peer-to-peer multicast streaming." in *IPTPS*, 2004.

[36] C.-C. Yeh and L. S. Pui, "On the frame forwarding in peer-to-peer multimedia streaming," in *ACM P2PMMS*, 2005.

[37] Y. Chawathe et al., "Making Gnutella-like P2P systems scalable," *ACM SIGCOMM*, 2003.