GKP+08] Kalman Graffi, Sebastian Kaune, Konstantin Pussep, Aleksandra Kovacevic, Ralf Steinmetz; **Load Balancing for Multimedia Streaming in Heterogeneous Peer-to-Peer Systems.** In: 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '08), ACM SIGMM, May 2008

Load Balancing for Multimedia Streaming in Heterogeneous Peer-to-Peer Systems

Kalman Graffi, Sebastian Kaune, Konstantin Pussep, Aleksandra Kovacevic, Ralf Steinmetz
Technische Universität Darmstadt, Multimedia Communications Lab KOM
Merckstraße 25, 64283 Darmstadt, Germany
{graffi[1],kaune,pussep,sandra[1],steinmetz}@kom.tu-darmstadt.de

ABSTRACT

Multimedia streaming of mostly user generated content is an ongoing trend, not only since the upcoming of Last.fm and YouTube. A distributed decentralized multimedia streaming architecture can spread the (traffic) costs to the user nodes, but requires to provide for load balancing and consider the heterogeneity of the participating nodes. We propose a DHT-based information gathering and analyzing architecture which controls the streaming request assignment in the system and thoroughly evaluate it in comparison to a distributed stateless strategy. We evaluated the impact of the key parameters in the allocation function which considers the capabilities of the nodes and their contribution to the system. Identifying the quality-bandwidth tradeoffs of the information gathering system, we show that with our proposed system a 53% better load balancing can be reached and the efficiency of the system is significantly improved.

1. INTRODUCTION

In recent years online communities emerged that had a strong focus on streaming of user generated multimedia content. Starting with Last.fm [2], a platform mainly for audio streaming, and YouTube [3], a platform for streaming of video clips, a wide set of other platforms emerged. Common to new platforms is that the multimedia content is mainly user created and provided.

Streaming of multimedia content states strict requirements on the quality of service (QoS) provided by the system, as it requires the contribution of various resources, ranging from bandwidth to online time. As the multimedia content is usually large, stream providers have to invest extensive amount of bandwidth and online time.

These costs can on the one hand be provided by a server farm architecture, which results in significantly high monetary costs, this is currently the case in the popular multimedia streaming platforms. On the other hand, the costs for service provisioning can be beard by the participating nodes, the user's devices. Nodes at the edge of the Internet often have considerable amount of resources like bandwidth or storage space available, that can be used. Distributed content creation and sharing has been long researched in the peer-to-peer (P2P) community. Whereas file sharing was the main application in the P2P domain, multimedia streaming may become the next significant application area for P2P-based solutions. Applying the P2P paradigm, architectures can even scale to millions of users by distributing the load in the system on the peers.

Load balancing is an important design goal in creating a distributed multimedia streaming platform, which relies on the contribution of the participating nodes. Once multimedia content is published in a distributed network, users consume and redistribute the content. Having various streaming provider for the same content leads to the question how to maintain the information of the providing peers in a distributed system and how to allocate the requests of content consumers to content providers. The redistribution of the content should be balanced on the participating peers so that the costs for the system, from which all participants benefit, are shared. Taking the heterogeneity of the nodes into account can result in a load balanced system which does not stress participants excessively but achieves to fulfill the quality of service requirements stated by the users of the multimedia streaming system.

The contribution of this paper is both a load-balanced architecture for P2P-based multimedia streaming and a stream provider selection mechanism, which can be applied on any distributed hash table (DHT). Having the multimedia content split up in content blocks, we assign for each block a responsible peer in the DHT. This peer maintains a list of peers providing the specific content block. Requests for this block are assigned by the DHT node to the providing peers by using a scoring function, which we present in this paper. The scoring function determines the quality of a peer, considering its capabilities (heterogeneity) and its contribution to the system (load balancing). We show that in comparison to a stateless dispatching, our solution results in a profit of up to 109% (measured by the scoring/cost function) and is capable to decrease the standard deviation of the load in the system by 53%.

The paper is structured as follows. First, we discuss solutions present for P2P-based multimedia streaming and P2P-based content delivery networks in Section 2. In Section 3 we present our solution for DHT-based multimedia stream-

[1]Authors supported by the DFG Research Group 733, "QuaP2P: Improvement of the Quality of Peer-to-Peer Systems" [1].

NOSSDAV '08 Braunschweig, Germany

ing using the terminology and models introduced in the same section. In the evaluation Section 4 we present our simulation setup and discuss the results intensively. The paper closes with a conclusion in Section 5.

2. RELATED WORK

A majority of existing multimedia streaming systems apply the client/server paradigm, where only servers or server farms provide the content, which results in scalability issues. An increasing number of requests can only be compensated by extending the amount or capabilities of the servers.

To disburden the servers, solutions for P2P-enhanced multimedia streaming has been deployed, like Kontiki [4], Octoshape [5] or BitTorrent DNA [6]. In these users that already received the content are helping to redistribute it. In contrast to these approaches, which assume that the multimedia content is generated only by one participant, upcoming multimedia streaming applications have to face the challenge of all participants in a distributed network creating, providing and consuming multimedia content.

There are two kinds of multimedia streaming applications: live streaming and video-on-demand (VoD) streaming [7]. The main difference is that a VoD system does not care about the freshness of the content, the videos are already pre-encoded and can be played asynchronously. Therefore the chunks can be distributed in any order and even be preloaded and cached in the network. Further, in VoD users can seek forward and backward, which again encourages some kind of pre-caching mechanisms. We focus on P2P-based multimedia-on-demand streaming, as the freshness of movie clips or audio streams is rarely of importance in current multimedia streaming platforms like YouTube or Last.fm.

Typical solutions for P2P VoD are either push-based application multicast trees or pull-based mesh systems. In a push-based solution [8] the peers are organized in application level multicast trees with the content source as root, which pushes the data towards the leaves. Contrariwise, in a pull-based system [9] a peer actively requests parts of data from available sources, which typically results in a mesh topology. The main benefit of pull-based solutions are lower costs, as the multicast tree maintenance is expensive, and the higher flexibility in the source selection. Different strategies for source selections can be applied, like the scoring function proposed in this paper.

In [10] the authors propose a video-on-demand solution which distributes the load for maintaining the content block information as well. However, content provider related information is not maintained by a dedicated peer, but inquired by the content requester any time a request is stated. This results in an increased traffic overhead.

For multimedia content distribution, currently BitTorrent [11] is a very popular tool. This system is mainly used for the distribution of stored content, such as large video files or software updates. In BitTorrent, files are broken into chunks and as soon as one peer has downloaded its first piece, it can start acting as a providing peer. However, there exist a few drawbacks as follows: First, when a new peer joins a torrent, it contacts the centralized tracker to obtain a list of peers being already involved in the distribution process. As this subset of peers is randomly chosen, it considers neither the heterogeneity of peers nor does it apply any load-balancing mechanism. Furthermore, the centralized tracker presents a single point of failure making the system not scalable. Another system on multimedia content distribution is Avalanche [12] that uses network coding [13]. This system utilizes also a centralized server (similar to BitTorrent) providing a subset of peers already in the system. Having the same drawbacks as mentioned before, it is likely that this system will also benefit from the mechanisms outlined in this paper. The utilization of an underlying DHT distributes the load from a single tracker to the peers in the DHT and annihilates with this the single point of failure.

3. OUR SOLUTION

In this section we present our solution for DHT-based multimedia streaming. First, we introduce a model for formally describing the distributed multimedia streaming scenario, then we present a DHT-based architecture which maintains the information of multimedia content providers. In Subsection 3.3 and 3.4 we present two solutions how to assign a multimedia stream provider to a stream requester. The first solution uses a scoring function to determine which stream provider to choose. This scoring function optimizes the decisions for load balancing considering the heterogeneity of the peers. The second (stateless) solution chooses randomly a stream provider.

3.1 Model of the P2P Streaming Scenario

In this section we describe the scenario more formally and introduce the terminology used in the succeeding sections. Let C be a streamable multimedia content, which is split up in m blocks: $C_1, C_2, ..., C_m$. The multimedia content blocks are stored in a network in which participants provide each other the desired content. Let P be the set of participants in the network, then we define the set $W_i \subseteq P,\ i \in \{1, ..., m\}$ for the participants that want to obtain block C_i and the set $H_i \subseteq P,\ i \in \{1, ..., m\}$ for the participants that already have the block C_i.

The question we focus on in this paper is: What is the best strategy for matching peers from W_i to H_i according to a scoring function. The scoring function should consider the load of the specific peer (both in regard of local and contribution load), its online time and its capabilities. This paper does not focus on the overlay specific routing and characteristics. With applying a cost function in the matching function, the quality of the decision can be optimized. The quality of the matches is measured by the load distribution in the system: How many service requests has been processed by which peer. An optimal solution results in a minimal standard deviation in the load.

3.2 Distributed P2P-based Multimedia Streaming Architecture

In this subsection we present our architecture for multimedia streaming. One design goal is to disburden the content creator by utilizing the bandwidth capabilities of content consumers as well in the distribution process. We developed a system architecture taking following design goals into account:

- Load balancing: Goal of the architecture is to provide a load balancing in the selection of stream providing peers.
- Low overhead: The costs for the system, measured in additional traffic, have to be low (in comparison to streaming traffic).

- Easy deployment: The mechanism has to be applicable in a mixed environment with peers supporting and not supporting the mechanism.
- Overlay (DHT) independence: The solution should be applicable on any DHT.

Our architecture assumes that the underlying multimedia streaming network provides the functionality of a Distributed Hash Table (DHT) [14]. To any content block C_i, which represents an object in the DHT, a peer can be identified, which is responsible for this content block. Messages addressed to a specific object identifier are routed in the DHT to the responsible peer. We do not state further requirements at the DHT, which makes our architecture generally applicable on any DHT.

The peer being responsible for a specific content block C_i is called R_i, it maintains a list of all peers in H_i. Besides maintaining H_i, the responsible peer R_i also receives requests for C_i, it decides to which peer in H_i to assign the streaming tasks. The contact address of this C_i providing peer is then replied to the requesting peer. We present two approaches in the next two subsections regarding whether further information on the peers in H_i is maintained by R_i or not. Both approaches could be used in a mixed scenario, which makes the architecture easy to deploy. Using further information enables R_i to come to optimized decisions, while increasing the (traffic) costs for the architecture. We assume that the providing peers cooperate and announce their content blocks C_i at the corresponding R_i. Security and incentive issues are not covered in this paper due to the lack of space.

Consuming peers (in W_i) retrieve the multimedia stream content block by content block (C_i) by requesting the contact information of streaming peers (in H_i) from the corresponding peers R_i. While consuming the multimedia file, the block index i is increased and successive blocks are (pre-) loaded. Focus of our investigations is which information to consider and according to which optimization goal to chose a streaming provider.

3.3 Assignment Using a Scoring Function

Each responsible peer R_i maintains the information of the offered content blocks (C_i). This information contains in a block-centric view the contact addresses of peers in H_i. Additionally, R_i maintains the state and information of all peers in H_i as well. The peers in H_i periodically announce their state at the corresponding R_i. Following information vector IV_p is maintained per peer ($p \in H_i$) in dependency of the time $t \in T$:

- Active Tasks $I_p^{AT}(t)$: Number of tasks already performed for the system. This parameter can be used to optimize the system regarding load balancing.
- Local Tasks $I_p^{LT}(t)$: Estimation of local load (e.g. number of active processes in relation to the computing power). With this parameter, the system can adapt to the heterogeneity of the peers. Assigning fewer tasks to weaker peers keeps the system stable.
- Bandwidth quality $I_p^{Bq}(t)$: This parameter shows the network conditions of the providing peer. Peers with low bandwidth capabilities are identified and the system can adapt to unburden them. Coping with bandwidth heterogeneity is a key question in upcoming mobile streaming applications.
- Online Time $I_p^{Ot}(t)$: Uptime of the corresponding peer. Considering this parameter, the power of peers staying only online for a short time [15] can be used more intensively which benefits peers that stay online longer. This incites the peers to remain in the network.

Two Approaches for Information Updates

There are two approaches for building a system, that relies on distributed information. Either information is used proactively or reactively. In a push model peers update their information in specific intervals proactively at the peers responsible for blocks they provide. This can lead to a high traffic overhead in a system, in which queries are rare. In a pull model peers send their status information reactively on demand. This solutions results in increased query costs, but decreases the update costs. Our solution follows the proactive approach, as the distributed multimedia streaming scenario states a high frequency of queries. The overhead can be adapted by tuning the parameter t_Δ, which is the frequency in which updates are transmitted by peer p to R_i for all $i \in \{1, ..., m\}$ with $p \in H_i$.

Task Assignment for Streaming

We assume that all peers $p \in P$ are connected to R_i for all $i \in 1, ..., m$ with $p \in H_i$. In a frequency of t_Δ each peer p transmits its updated information vector IV_p to R_i. When a peer $k \in W_i$ wants to obtain a specific content block C_i it sends a query to R_i asking for a peer providing C_i. The peer R_i determines from all peers in H_i one peer to recommend, using a scoring function $c_s(p,t) : P \times Time \rightarrow \mathbb{R}$, which calculates the costs for choosing a specific peer. The scoring/costs of a peer is value depicting the adequacy of the peer for being recommended. The peer with the lowest costs is chosen by being considered to be the fittest. By taking the peer characteristics into account the most suitable peer for providing the requested multimedia stream can be determined.

We define $c_s(p,t)$ (and s) in the following way:

$$c_s(p,t) := s(I_p^{AT}(t), I_p^{LT}(t), I_p^{Bq}(t), I_p^{Ot}(t)) \quad (1)$$
$$:= \alpha_1 \cdot I_p^{AT}(t) + \alpha_2 \cdot I_p^{LT}(t) + \alpha_3 \cdot I_p^{Bq} + \alpha_4 \cdot I_p^{Ot}(t)$$

After calculating the costs for each peer providing the desired content, the peer R_i sends a message back to the querying peer, recommending the most suitable multimedia stream provider. The streaming of multimedia content from a peer in H_i to a peer in W_i generates much more load on the peers (in H_i) than assigning the request by R_i to a peer in H_i. The balancing of the (higher) streaming load compensates for the dispatching load on the DHT peers. In Section 4 we investigated the impact of the choice of α_1 to α_4. The results show that by tuning these parameters, the load distribution in the system can be decreased by 53%. The scoring function can be extended by various more parameters, for example taking QoS requirements into account as well. In addition, overlay bandwidth management mechanisms [16] can be used to increase further the provided quality of service.

3.4 Stateless Assignment

We present a stateless solution that is similar to current approaches in P2P-based multimedia streaming [17]. This solution assumes the existence of a responsible peer R_i per

content block in the DHT as well, which maintains the information about the offered content blocks C_i. The stateless solution stores no additional information on the peer characteristics. Peers requesting a specific content block (C_i) contact R_i using the DHT and request the address of any peer $p \in H_i$. The peer R_i responds with the network address of a peer chosen randomly from the set of the peers offering C_i. After this step, peer R_i updates its internal information on H_i and W_i. This stateless solution results in less traffic overhead, is easy to deploy and provides load balancing as well.

3.5 Summary

We presented a DHT-based architecture for multimedia streaming. The multimedia content is split up in content blocks, for which individually a responsible peer in the DHT is assigned. This peer maintains a list of peers providing the specific content block. Providing peers periodically update their status information at these responsible peers. Based on a scoring function, the responsible peer calculates which peer should be utilized to stream a requested content block. As the scoring function considers the status information of the peers, the load of the peers can be balanced and the heterogeneity of the peers taken into account. In the next step, we present the evaluation of our architecture and the parameters of the scoring function.

4. EVALUATION

In this Section, we present the performance measurement of our architecture. First, we describe the simulation setup and the used metrics, before we present and discuss the simulation results.

4.1 Simulation Setup

Our scenario is inspired by the multimedia streaming requirements of today's platforms. In today's content distribution networks like BitTorrent [11] there are only tens to one hundred peers requesting a file [18] at a time. We therefore focus on the streaming strategy for multimedia content up to 100 participants, the simulation setup consists of 25, 50, 75 and 100 peers. For each request a peer in the network is chosen, which then states a query for one chunk it is looking for.

We chose the P2P simulator PeerfactSim.KOM [19] for evaluating our architecture, it focuses on inter-dependencies between various layers in a P2P systems. We extended the simulator with both solutions and adapted the user layer to define the content preference distribution of the peers. Focus of the evaluation is the load balancing of the multimedia content provisioning peers. We investigate the load on the peers resulting from the request allocation strategy.

4.2 Metrics

For the rating of the quality of the solutions we have chosen metrics focusing on the obtained load balancing and the traffic overhead generated. We measure the *load of providing peers* in form of number of requests allocated to them by a peer R_i in the DHT responsible for a content block. The distribution of the allocated requests shows how well the system is balanced in terms of load. We use *the standard deviation of the load distribution* as a metric for fairness in request assignment. Further we use *the difference between the costs* for using the solution based on the scoring function and the stateless solution as an indicator for the impact of the scoring function parameters α_1 to α_4. In order to investigate the impact of the update frequency t_Δ we measure the *average error rate in relation to the update interval.* We identify the tradeoff between the error rate and the traffic overhead for keeping information up-to-date. We define the term *Profit* for a metric M as the ratio of additional costs for the stateless solution (RND) in comparison to the solution using the scoring function (SF):

$$Profit_M = \frac{M_{RND} - M_{SF}}{M_{RND}} \quad (2)$$

By measuring the profit, we identify the quality gain when using the scoring function.

4.3 Simulation Results

Before presenting the simulation results, we give a brief outline on the structure of this section. We present in the next subsection the impact of the parameters α_i on the costs for task assignment in the system. With this evaluation step, we identify a setting for the α_i to use for the next investigations on the effects of an increasing number of participants and requests in the system. We show in Subsection 4.3 that with increasing number of peers our solution results in an increasing profit in comparison to the stateless solution. In this subsection we present the correlation of the traffic costs on the freshness of the information and show that the error rate and traffic costs are nearly linearly anti-proportional. The freshness of the information and with this the traffic overhead can be adapted to any specific scenario linearly.

Parameters α_i in the Scoring Function $c_s(\cdot,\cdot)$

In order to evaluate the load distribution on the peers, we first investigated parameters in the scoring function $c_s(\cdot,\cdot)$ defined in Equation 1. We varied α_1 and α_2 in the scoring function, as they represent the impact of the number of active and local tasks. Table 1 shows five setups for α_i and how they affect the profit of the system according to the scoring/cost function. We focused on the variation of the parameters concerning load-balancing. The impact of these parameters on the function is as important as the impact of the parameters concerning the heterogeneity of the peers. We therefore varied the impact of the load balancing parameters α_1 and α_2, which have a sum of 50% in total. The parameters α_3 and α_4 modeling the heterogeneity of the peers are both set to 25%. However, these parameters can be tuned as well in order to meet the requirements of a given scenario.

Figure 1 shows the task allocation distribution for a content block using this five setups in a scenario with 100 peers and 100 service requests in total. The Figure shows that with increasing impact of α_1 the deviation in the load distribution decreases. The parameter α_1 represents the number of allocated tasks to a peer. By giving more impact on this parameter, load balancing is improved at the expense of the heterogeneity of the peers having less effect on the task allocation.

	setup1	setup2	setup3	setup4	setup5
Active Tasks (α_1)	5%	15%	25%	35%	45%
Local Tasks (α_2)	45%	35%	25%	15%	5%
Profit	65.14%	62.29%	56.26%	62.15%	76.88%

Table 1: Impact of α_i in $c_s(p,t)$ on the Profit

Variation in the Number of Peers and Requests

With the variation of α_i, we identified a suitable parameter setting with $\alpha_1 = 45\%$, $\alpha_2 = 5\%$, $\alpha_3 = 25\%$ and $\alpha_4 = 25\%$. Based on these values we investigated the impact of the number of peers and number of requests in the system. We varied the number of peers from 25, 50, 75 to 100 and investigated the profit in a system with 25 requests (see Table 2). We also investigated the profit of our solution in comparison to the stateless solution in a system in which the number of peers and requests are equal, e.g. 50 peers and 50 requests. The profit of our solution (by applying the scoring/cost function to the chosen peers) is depicted in Table 2. The table shows that our solution outperforms the stateless solution by at least 36%. With increasing number of peers the profit grows to 109.84%, i.e. the decisions resulting from the reference solution cost 109.84% more in relation to the results of our solution. With increasing number of peers and requests, the profit increases as well.

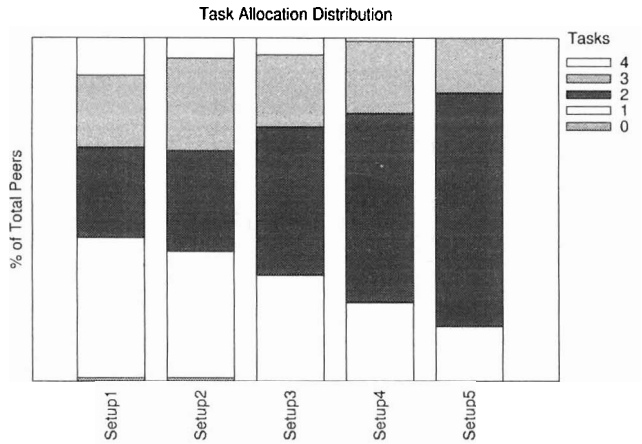

Figure 1: Impact of α_i on Load Balancing

	25 peers	50 peers	75 peers	100 peers
25 resource requests	36.69%	66.25%	94.63%	109.84%
# of requests = # of peers	36.69%	62.89%	74.40%	76.88%

Table 2: Profit with Varying Number of Peers

We investigated the load distribution in the system in relation to an increasing number of peers and requests in the system. In the multimedia streaming scenario, we aimed both on taking the heterogeneity of the peers into account, but still have a load balanced system. The results of the simulations are shown in Table 3 performed for 25, 50, 75 and 100 peers and the same number of requests per setup. The number of tasks assigned to a peer is also shown in Figure 2 in the columns labeled with 0 to 7.

The figure shows that the deviation in the task distribution is smaller using the scoring function (SF) in allocating requests for multimedia content. Assigning randomly (RND) a peer providing the requested multimedia content, leads to more variation. This effect can be best seen in the two right columns of Table 3. We denoted the standard deviation σ in the number of allocated tasks and the Load Balancing Saving LBS. The mean number of allocated tasks per peer is in any case the same as the numbers of peers and requests are in all setups equal. The metric LBS is defined as the profit in the standard deviation

$$LBS = \frac{\sigma_{RND} - \sigma_{SF}}{\sigma_{RND}} \quad (3)$$

The metric LBS represents the ratio by which the deviation in the number of allocated tasks per peer is decreased when using our solution. The Table 3 shows that the Load Balancing Saving metric increases with increasing number of peers and requests. This is due to the increased number of peers the scoring function can take into account, based on a higher number of candidates better peers can be chosen using the scoring function. The table shows that the load balancing can be increased by 53%, even better results can be expected in a system and scenario that involves more participating peers.

Peers	Sol.	0	1	2	3	4	5	6	7	σ	LBS
25	RND	2	8	8	4	1	2	0	0	1.32	
25	SF	1	8	9	5	1	1	0	0	1.12	0.15
50	RND	1	24	11	6	4	4	0	0	1.35	
50	SF	0	14	23	12	1	0	0	0	0.78	0.42
75	RND	2	29	19	19	5	0	1	0	1.12	
75	SF	0	15	45	15	0	0	0	0	0.64	0.43
100	RND	11	18	45	19	3	2	1	1	1.24	
100	SF	0	17	66	17	0	0	0	0	0.58	0.53

Table 3: Task Assignment Distrib. with Varying Mechanism and Number of Peers

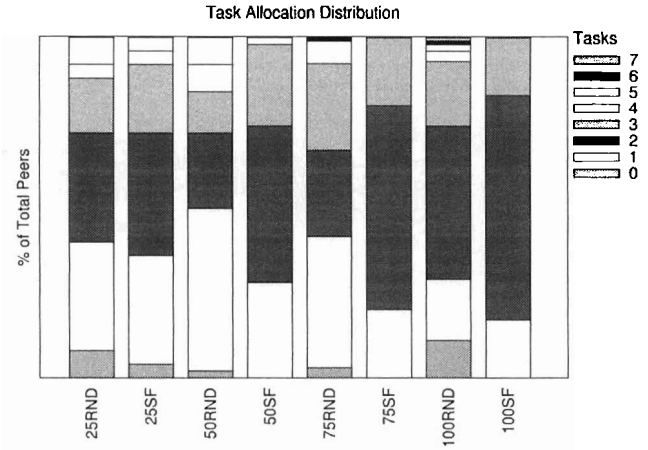

Figure 2: Load Distribution in the System with Varying Mechanism and Number of Peers

Tradeoff: Traffic Costs and Information Freshness

Having proved the performance of our solution leads us to investigating the costs. Information on the multimedia content providers has to be maintained at the peers responsible for the specific multimedia content block in the DHT. Keeping this information fresh requires frequent updates. We investigated the traffic overhead resulting from varying update intervals and how this effects the error rate of the information. The error is measured as maximum difference between the real value at a specific peer and the information about this value stored at the last update on the corresponding content block maintainer. The interval for the error rate is from 0 to 2. Figure 3 shows a scenario of 100 peers and 100 requests. The number of messages (left y-axis) resulting from various update intervals (x-axis) decreases with increasing update intervals as greater update intervals mean less frequent updates and with this less traffic overhead. The error rate of the information (right y-axis) increases with increasing update intervals. However, the normalized product of the number of messages and the error rate does not vary much, both metrics are nearly anti-proportional.

In order to result in better stream assignment decisions (measured by the scoring function), our solutions generates

traffic overhead for the information updates. However, as the information sent by the peers in H_i at R_i for updating their state consist only of a few values, the generated update traffic is small. In comparison to this small traffic overhead, the streamed multimedia content is assumed to be large in size. The scoring function leads to load balancing of the huge multimedia streams by generating small information update traffic at the DHT peers. The exact ratio of benefit in load balancing in comparison to the required update traffic depends not only on the update traffic, but also on the network size and the request frequency.

Depending on the criticality of the freshness of the information and the information type, a traffic optimal update interval can be chosen. There is inflection point in the relation between the number of messages and the error rate, so that the update interval (and with that the costs for our solution) may even adapt continuously to the requirements in a specific scenario.

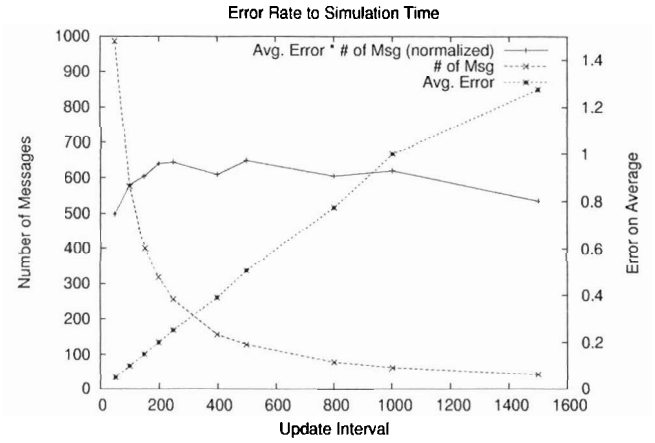

Figure 3: Tradeoff between Message Overhead and Information Quality

5. CONCLUSION

Multimedia streaming of mostly user generated data is an ongoing trend, not only since the upcoming of Last.fm and YouTube. Future applications may support streaming of haptics as well [20]. Due to the distributed creation of multimedia content we investigate in this paper, how to build an architecture that is distributed, load-balanced and takes the heterogeneity of the participating nodes into account.

The contribution of the paper is an architecture, which relies on a Distributed Hash Table, as it is common in today's P2P systems. We assume that the multimedia content is split up in content blocks. Our design is independent of any specific DHT, we assume that for any content block a peer in the DHT is responsible. This peer maintains a list of peers, that provide the content block, and responses to queries of peers asking for the specific content block. We propose to use a scoring function taking parameters for heterogeneity (bandwidth quality, online duration) and load balancing (active and local tasks) of the peers into account. This scoring function decides which peer to assign the streaming request to. We evaluate our solution in comparison to a stateless solution. Evaluation shows that our solution outperforms the reference solution by saving up to a profit of 109% and that load balancing in the system can be improved by 53%. By identifying traffic costs and tradeoffs between the error rate of the information and traffic overhead, we have shown that the performance and costs of the solution can be adapted nearly linearly to any scenario. By additionally tuning the system to support the heterogeneity of the nodes in the system our solution leads to a load balanced multimedia streaming system, which makes it applicable for a wide range of scenarios.

6. REFERENCES

[1] DFG Research Group 733, "QuaP2P: Improvement of the Quality of Peer-to-Peer Systems," http://www.quap2p.de.

[2] Last.fm - the Social Music Revolution, http://www.last.fm.

[3] YouTube - Broadcast Yourself, http://www.youtube.com.

[4] VeriSign - Kontiki Delivery Management System, http://www.kontiki.com/.

[5] Octoshape, http://www.octoshape.com.

[6] BitTorrent DNA, http://www.bittorrent.com/dna/.

[7] J. Liu, B. Li, and Y.-Q. Zhang, "Adaptive video multicast over the internet," *IEEE MultiMedia*, vol. 10, no. 1, 2003.

[8] T. Do, K. Hua, and M. Tantaoui, "P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment," in *IEEE ICC*, 2004.

[9] S. Annapureddy *et al.*, "Exploring VoD in P2P Swarming Systems," in *IEEE INFOCOM '07*, 2007.

[10] X. Liu and S. Vuong, "Supporting Low-Cost Video-on-Demand in Heterogeneous Peer-to-Peer Networks," in *IEEE ISM '05*, 2005.

[11] B. Cohen, "Incentives Build Robustness in BitTorrent," in *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[12] Ahlswede *et al.*, "Network information flow," *IEEE TIT: IEEE Transactions on Information Theory*, vol. 46, 2000.

[13] C. Gkantsidis and P. Rodriguez, "Network Coding for Large Scale Content Distribution," in *IEEE INFOCOM '05*, 2005.

[14] K. Aberer *et al.*, "The Essence of P2P: A Reference Architecture for Overlay Networks," in *IEEE P2P '05*, 2005.

[15] S. C. Rhea *et al.*, "Handling Churn in a DHT," in *Usenix Annual Technical Conference '04*. USENIX, 2004.

[16] K. Graffi *et al.*, "Overlay Bandwidth Management: Scheduling and Active Queue Management of Overlay Flows," in *IEEE LCN '07: Local Computer Networks*, 2007.

[17] P. Shah and J.-F. Paris, "Peer-to-Peer Multimedia Streaming Using BitTorrent," in *IEEE IPCCC'07*, 2007.

[18] Q. Lian *et al.*, "Robust Incentives via Multi-level Tit-for-tat," in *IEEE IPTPS '06*, 2006.

[19] A. Kovacevic *et al.*, "Benchmarking Platform for Peer-to-Peer Systems," *it - Information Technology (Methods and Applications of Informatics and Information Technology)*, vol. 46, no. 3, 2007.

[20] A. El Saddik, "The Potential of Haptics Technologies," *IEEE Instrumentation & Measurement*, 2007.