Replication for Bio-inspired Delivery in Unstructured Peer-to-Peer Networks

Anita Sobe
Institute of Information Technology
Alpen-Adria Universität Klagenfurt
Austria
anita.sobe@aau.at

Wilfried Elmenreich
Institute of Networked and Embedded Systems
Alpen-Adria Universität Klagenfurt
Austria
wilfried.elmenreich@aau.at

Laszlo Böszörmenyi
Institute of Information Technology
Alpen-Adria Universität Klagenfurt
Austria
laszlo.boeszoermenyi@aau.at

***Abstract*—Many of the current bio-inspired delivery networks set their focus on search, e.g., by using artificial ants. If the network size and, therefore, the search space gets too large, the users experience high delays until the requested content can be consumed. In this paper we propose replication strategies to reduce this delay. Typical mechanisms, applied in unstructured P2P networks, such as replication at the target (owner replication) and replication on the travel path of content (path replication) are either inefficient or the user experience suffers because of the long distance between content and requester. Based on an previously introduced self-organizing hormone-based delivery algorithm, we compare seven existing and proposed replication mechanisms. We show by simulation that the exploitation of local knowledge about the desire for the requested content performs best in scale-free and random networks. These results are expected to provide a guide towards designing future self-organizing bio-inspired networks.**

I. Introduction

The emergence of large networked databases and increasingly dynamic access patterns have led to new challenges for content delivery, i. e., the combination of searching, routing and replica placement in the network. One example from multimedia is a live sports event such as the Ironman Triathlon. Visitors produce masses of multimedia content, either photos or short videos. With the advent of social networks like Flickr, or YouTube much of this content is shared. However, on site the visitors have to rely on video walls with content provided from the organizers. It is not possible to share their content at the event directly with other visitors a few kilometers away. Assuming visitors have smart phones and tablet computers, direct access to the Internet would be challenging for multimedia delivery. This was shown in a similar scenario by Stiemerling et al. in [1]. The authors propose an ad-hoc WLAN network for sharing content in combination with some nodes connected to the Internet. For our purposes, we assume an unstructured peer-to-peer network as a basis, however, how this network is built is out of scope of this paper.

A scenario like the Ironman is dynamic on several levels. First, the production of multimedia content is dynamic at several places of the network. Second, since the content is rather small, users are likely to consume a number of videos in a row. However, there is no notion of sequentiality, such as at a movie. It is not predictable which video is watched first and it is further not known if a number of users choose the same order of videos to be presented.

Such dynamics can be best handled by robust and adaptive systems [2]. However, current delivery systems concentrate either on search or on transport. One example for the first case is ant-based search, such as applied for peer-to-peer by Michlmayr in [3]. It allows for the adaptive handling of network changes, but content may not be found or can only be delivered with high delays if the search space is too large. BitTorrent is an example for outsourcing search to other facilities and to concentrate on the delivery of content. If BitTorrent is used a visitor of a live sports event might need another near visitor with the required content to experience the needed QoS for multimedia presentation.

Our idea is to combine search and transport and handle dynamics by introducing a self-organizing delivery algorithm. Our algorithm introduced in [4] is inspired by ant-based search as introduced as SemAnt in [3] and hormone-based task allocation as described by Brinkschulte et al. in [5]. SemAnt is an implementation of Ant Colony Optimization [6] for keyword based query routing in peer-to-peer systems. Brinkschulte et al. propose a hormone-based system for distributed task allocation. A hormone indicates the goodness of a node for executing a task. The current hormone value of a node is called *eager value*. This value can be increased by the *accelerator value* or decreased by a *suppressor value*. The eager value is recalculated based on the suppressors and accelerators and disseminated to the network. The node with the highest eager value executes the task and broadcasts suppressors to avoid duplicate execution of a task.

We adopted the keyword search from SemAnt and introduce one type of hormone by keyword. The hormone value expresses the current demand (the “goodness”, as adopted from Brinkschulte et al.) for the corresponding keyword. So, the search is done by spreading hormones over the network. Accelerators represent a periodical increase of hormones by the requester until the content is delivered. Suppressors are implemented by an evaporation mechanism. If the content is placed at fixed nodes, such as assumed by SemAnt, the search space gets very large. In order to reduce the search space and to limit the necessity of global knowledge, in our work, content is attracted by hormones and transported hop-by-hop (such as in Freenet [7]). On the intermediate nodes the content can be replicated. In our former implementation the replication mechanism was limited, i.e., a video or photo was only replicated if it is currently consumed by the client. In this paper we evaluate existing and proposed replication methods that support our main goal – the placement of content where it is needed, but *before* it is requested. Since not only the production of content is dynamic, but also the consumption, the popularity of content is dynamic as well. The provided discussion should not only target our proposed algorithm, but should also be a design guide for future bio-inspired delivery systems.

II. Related Work

In this section we give a short overview of existing replication strategies in unstructured P2P networks. We focus on algorithms that aim at improving search efficiency. The following categorization is based on the surveys from Androutsellis et al. [8] and Yamamoto et al. [9].

Owner Replication. The content is replicated at the requester's node [10] and is also referred to as passive replication. Typically, this replication technique is used in file sharing systems based on BitTorrent [11], because no intermediate peers exist during download.

Path Replication. In a multi-hop network where content is transported indirectly such as in Freenet [7], it is possible to cache a replica of the content in transmission in each intermediate node. Since the intermediate nodes are acting as caches, the path replication is also referred to as cache-based replication. It is assumed that intermediate nodes provide storage space for replicas even if they are not interested in the content. Path replication leads to a high number of unused replicas.

Therefore, an improved approach replicates the content on an intermediate node according to a fixed replication rate (*path random replication* [9]). The advantage of this approach is a compromise between a higher replica usage and limited hop distance to other replicas. The difficulty of this approach is to specify a suitable replication rate for each file in advance, which is hard if the files are not known at system startup. Therefore, an alternative is to specify a node specific replication probability, where nodes decide ad-hoc if a file is replicated or not. The replication probability is dependent on the peer's resource status and optionally refers to the replication rate, too. The authors in [9] refer to this strategy as *path adaptive replication*.

Active Replication. The goal is to place the right number of replicas at the right locations *before* they are requested. Researchers investigated therefore the optimal number of replicas. In [12] and [10] the authors investigate *random, proportional, and square root replication*. When applying random replication a uniform number of replicas are created. Proportional replication creates replicas proportional to their query rate. The authors showed that square root replication determines the optimal replication rate r_i for object i, which is calculated as $r_i = \lambda\sqrt{q_i}$, with $\lambda = R/(\sum_i \sqrt{q_i})$, where q is the query rate and R is the number of object replicas in the system. Square-root replication does not consider the location of replicas. All strategies require global knowledge on the number of currently existing replicas and the current query rate for each of the replicas.

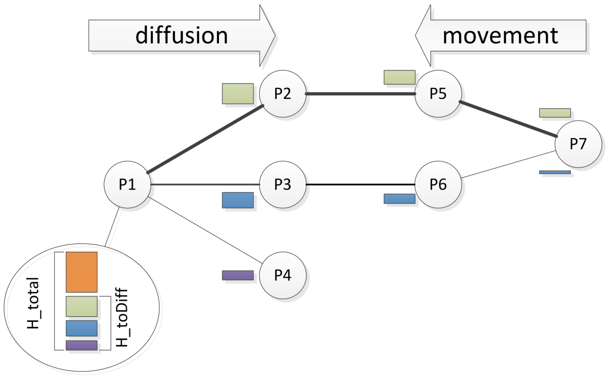

Fig. 1. Interplay of diffusion and unit movement in hormone-based delivery

To reach square-root replication with limited knowledge researchers proposed *Pull-then-Push Replication* introduced in [13]. This replication method consists of two phases. The first phase regards the search of the content, with any existing algorithm. The second phase regards the replication of content to the neighbor nodes. To reach square root replication, the authors suggest that for the pull and push phase the same algorithms are used, because the number of replicas should be equal to the number of nodes visited during search. The authors evaluated typical algorithms, such as flooding and random walks. Their focus is set on robustness even on updates. As multimedia content is usually not updated after creation and this algorithm does only consider the number and not the location of content, this approach is out of scope of this paper.

For our evaluations we refer to replication mechanisms that do not need global knowledge, i.e., owner replication, path replication, and path adaptive replication.

III. Hormone-based Delivery

The proposed algorithm is distributed and self-organizing and allows to handle the complexity of requests and the search for units in the network with comparably simple decision algorithms based on local knowledge.

The idea of our algorithm [4] is to guide units to the right places by spreading hormones, in opposite to the ant algorithm, which guides queries through the network.

The algorithm consists of two phases: search and delivery. Search consists of the hormone creation for indicating the demand for a content, and hormone diffusion to spread the demand over the network. The delivery involves the movement of units guided by the corresponding hormone. The stronger the hormone on a node the more likely the unit will move towards this node.

A simple example shown in Figure 1 depicts the general behavior of the algorithm. Assuming a network consisting of 7 peers, and P_1 wants to have a unit that is located at P_7.

P_1 only knows its direct neighbors P_2,P_3 and P_4. Peer P_1 creates periodically new hormones for the desired unit. A unit is attracted by a higher hormone level, therefore it is necessary to get a hormone trail with the highest hormone level at the requester. So, each node forwards only parts of its hormones until the corresponding unit is found or if no hormone is left to forward. A node differentiates its neighbors by their provided QoS (e.g., the lowest link load, or lowest round trip time). The neighbor with the best provided QoS gets the highest hormone value (in the figure marked with thicker lines).

If a unit is found the second phase starts and the unit will be transported towards the requester. In the depicted example the unit will be moved from P_7 to P_5,P_2 and finally to P_1.

The negative impact (suppressor value) is implemented by periodical hormone evaporation, which is needed to reduce hormones on alternative paths. On the movement path the hormones are deleted on the current peer to avoid the attraction of replicas.

The hormone-based delivery creates a feedback loop between network conditions. The network conditions influence hormone diffusion, the hormone diffusion influences unit movement, which in turn creates network traffic and changes the network conditions.

Multiple requests for different units lead to a set of different hormones being handled in parallel by the network. Requests for the same unit result in a superimposed hormone landscape for that unit. In this case, a unit might be attracted by two hormone trails. Without replication the unit has to move first to the first requester and afterwards to the second requester. Which requester gets the unit first depends on the strength of hormone reaching the unit. In order to avoid such detours, an intelligent replication mechanism has to take care of this issue.

IV. Hormone and Popularity-based Replication Strategies

The basis for our proposed replication mechanisms is owner replication, since the units are consumed for some time and therefore need to be replicated to be further usable by other nodes. Units replicated at the requester can only be supportive for the immediate neighborhood. To serve future requests, replicas should also be created on the delivery path. If the hormone level of a neighbor attracts a stored unit, the peer has to decide whether to move or to copy the unit. The simplest solution would be to apply path replication, but then the utilization of replicas would drop and the storage space is not used efficiently. Therefore, the goal is to find a replication mechanism that balances replica utilization and delay without the need of global information.

Beyond owner replication, path replication, and path adaptive replication we evaluate the following four replication mechanisms. These mechanisms exploit local knowledge on popularity and hormone levels collected from neighbors.

1) **Simple Hormone**. If a unit is requested by peers from opposite parts of the network, the unit has to move first to one requester and afterwards to the other requester. This can lead to long traveling paths, which can be avoided by replicating a unit if more than one neighbor holds hormones for it. Note that it is not possible to differentiate if hormones on the neighbor are created by different peers. Thus, in a situation as depicted in Figure 1 the replica would be created unnecessarily.
2) **Local Popularity**. Each node uses the local request history of the corresponding content to decide if it is likely to be requested again in the future. If the rank of a content is among the best 30 % the corresponding unit is kept. So popular units are more likely to be replicated, but popularity information from neighbors is ignored. The communication effort is minimized.
3) **Neighbor Popularity Ranking**. After collecting the popularity ranks for a content from the neighbors, the peer decides if it is worth to replicate the corresponding unit. The ranks are aggregated to a region rank (introduced in [14]), which is calculated as follows:

$$R = \frac{1}{n} \sum_{1}^{n} \ln(r_i)$$

   If the region rank R is lower than a given threshold (e.g., the best 30 % at all neighbors) the unit is replicated. n represents the number of neighbors and r is the rank of the specific unit at this neighbor. To reduce the impact of peak ranks (e.g., one unit is best ranked at two nodes, but worst ranked on the third node) the logarithm is used. The cooperation of neighbors is advantageous if their taste diverges.
4) **Neighbor Hormone Ranking**. Analogue to the popularity ranking the units can also be ranked by their hormone values at the neighbors. The higher the hormone value for a unit on a neighbor is, the better is the unit's rank. The collected ranks can be aggregated as before and if the region rank is lower than a given threshold (e.g., the best 30 %), the unit is replicated.

V. Evaluation Settings

We implemented a time-triggered discrete simulator. In such a simulation, the peer's actions (requests, hormones, unit movement) are guided by a cyclic schedule, which gives a comprehensive overview of the processes and dependencies of the simulated system [15].

The detailed settings are described in the following.

A. Network Topology

We assume for small overlay networks of 50 nodes a connected Erdős-Rényi random graph with a diameter of 6. For larger networks, e.g., with 1,000 nodes, we assume a scale-free network topology. To generate such a network the Eppstein Power Law Algorithm [16] is used. The algorithm gets as input a random graph and by repetitively removing and adding edges a power law distribution is reached. The network diameter of the scale-free graph is 13. The bandwidth was set to 100 Mbit/s. Note that further bandwidth scenarios and parameter studies are target of future work.

B. Initial Storage

Each node creates units up to a predefined initial storage limit. At the beginning only one instance of each unit exists. We expect that in a scenario with 50 motivated persons, each person is contributing with equal probability. In a scenario with 1,000 visitors we expect that there are few highly motivated persons and a high number of less motivated persons. We further assume that each person is represented by one peer. We generate 5,000 units for the 50 peers scenario, and 15,000 units for the 1,000 peers scenario.

The average size of a unit is 2.6 MB, whereas the maximum size is 16 MB and the minimum size is 190 KB, with a playback bit rate of 1 Mbit/s. These sizes are the result of an analysis of a use case event performed at our university, where visitors were encouraged to contribute their videos and photos taken at the event.

C. Request Generation

Units are considered to be of different content types, which we mapped to a three dimensional array (in a real setting the array can also have more dimensions). Each dimension contains an integer index of a different category (e.g., swim, run, bike, people,...). Due to this mapping the similarity can be calculated by applying the Euclidean Distance. One of such arrays may describe a number of units. How many units are mapped to one content array is dependent on the content's popularity (Zipf-like distribution). Content arrays describe video units like keywords.

A user has a specific taste described by a content array. Based on this taste requests for similar content are generated. A request consists of a number of content arrays and is fulfilled if for one content at least one unit can be presented. Hormones are generated for each of the content arrays and not for the distinct units. We further implemented a taste change, i.e., if a user likes the content just watched, the taste for future requests might be similar to the currently watched unit.

In this paper we do not consider any order of the units, thus, if a requested unit arrives, it is presented to the user. Further possible patterns, such as presentation of one unit after another (sequential) or all starting at the same time are (parallel) are out of scope of this paper, but are under investigation.

Additionally, we introduce a deadline for each unit, until which it has to be delivered. The deadline is dependent on the size, the link bandwidth and a constant multiplier (the maximum number of hops a unit can travel). If a deadline is missed, no further hormones for that unit are created to stop attracting content.

A request is considered as failed if none of the requested units could fulfill their deadline. A user can only submit one request at a time. If this request is fulfilled or failed, a new request will be generated.

D. Simulation Parameters

The delivery algorithm is a very adaptive, robust and flexible method, however, depends on proper parameter settings for amount of hormone creation, diffusion factor, migration threshold, etc. These parameters and thresholds are highly dependent on each other. E.g., if a low number of hormones is created and forwarded, a high evaporation rate might slow down the movement of the corresponding unit. Therefore, we used an evolutionary algorithm for generating a parameter set such as found in [17]. The fitness function was chosen to maximize the number of successful requests. The optimized parameter set is used for both random and scale-free network scenarios.

hormone initial	3.95
hormones increased p. sec	4.39
% hormones diffused p. sec	45
hormones evaporated p. sec	0.16
migration threshold	0.23
% of storage used to trigger cleanup	60

TABLE I
PARAMETER SETTINGS

In Table I the resulting parameters are shown. On creation the hormone value is high, leading to a wider travel range. The diffusion of 45 % of the hormones supports longer travel distances, too. The evaporation rate is in comparison to the creation number rather low, which means the hormones last for some time. The evaporation hormones are a fixed value subtracted from the current hormone level. The migration threshold describes the minimum hormone difference between two nodes to make a unit move. In this case the difference is very low in comparison to the creation amount of hormones. This leads to a very dynamic behavior of the units. The clean-up is triggered by a node if its current storage level exceeds 60 %.

E. Metrics

We want to evaluate the request fulfillment on the one hand and the utilization of replicas on the other hand. The fulfillment of requests is represented by the *delay*. The delay is measured from the request time of a unit until the arrival of that unit on the node. A delay of 0s̃ means that the unit was already on the node. The delay is presented as cumulative distribution function (CDF) over the simulation time. The *deadline missed rate* represents the rate of units (not requests), for which the deadline is missed. If a unit missed its deadline, the delay is calculated as deadline minus request time (max. delay). The *request failed rate* indicates requests from which all units missed their deadline.

A unit is presented for some time, and we measure the rate of units that currently started with presentation. The more unit presentations started in comparison to the number of their replicas, the better the *unit utilization*. The utilization and the request failed rate will be depicted as box plot with 1.5 interquartile range whiskers.

VI. EVALUATION

We conducted extensive simulations for the random and the scale-free network topology. We performed each simulation in 10 runs for 500 simulated seconds. Each run started with a different seed for the random number generator. The random number generator has an impact on the network topology and the request generation and anything further that needs random input. The results of these runs are averaged. Since the performance difference of each run is negligible, we reduced the evaluation runs for the scale-free network. The delay stabilizes after an initial simulation time of app. 100-200 seconds, a run time of 500 seconds for the random network and 700 seconds for the scale-free network is therefore sufficient.

A. Scenario 1: 50 nodes random network

For this scenario we first show the impact of the single replication methods on utilization and delay. Afterwards we evaluate the robustness of selected algorithms in case of peer churn.

1) Impact of Replication Methods: In this part we evaluate the impact of replication, which takes place in a best effort manner until no storage space is available anymore. A full storage space on a peer prevents forwarding of further units. The goal is therefore to find an intelligent replication mechanism.

In Figure 2 the delay development for all replica mechanisms is depicted. Hormone ranking (hranking) outperforms path replication (path), although path replication generates more replicas. The placement of hormone ranking is more efficient, leading to less overloaded nodes and therefore more requests can be fulfilled. Local popularity (pop) has a lower delay than popularity ranking (pranking), because the best 30 % for popularity may contain more units than the best 30% of popularity ranking, thus popularity ranking creates less replicas. An adaptive threshold for popularity ranking might lead to better results. The path adaptive replication mechanism (path_adapt) shows that random decisions can also lead to good results. The hormone replication mechanism (hormone) tries to replicate if there are currently further requests for that unit from somewhere else. This leads to a low number of replicas if the hormones do not reach the current location of the unit. This graph shows that the number of replicas has a high impact on the service quality. If too many replicas are created, the storage used inefficiently and if full, it blocks further transport. If the number of replicas is too low, the delay is high, because of long distance transport.

A more detailed view on the storage efficiency is shown by the utilization rate. Note that it is collected only once, at the playback start. Figure 3 shows interesting results. Owner replication has the best utilization, which is explainable by the low number of replicas. On the first sight one might say that the more replicas created the lower the utilization. But the hormone ranking mechanism creates less replicas than the path replication mechanism. The difference in utilization can be explained by comparing the delay of these replication mechanism. The hormone ranking replication has a far lower delay than the path replication. Since a request is sent after another is fulfilled or failed a lower delay means that more requests can be submitted and therefore more replicas can be generated, which results in lower utilization. In general the utilization is a metric that has to be evaluated in combination with delay and request failed rate.

The request failed rate is depicted as box plot in Figure 4. It shows that hormone ranking and path replication perform 50% better in comparison to owner replication. This indicates that the placement of units is well developed. The outliers (marked with x in the figure) are experienced during the first few seconds of the simulation, when the unit placement is random.

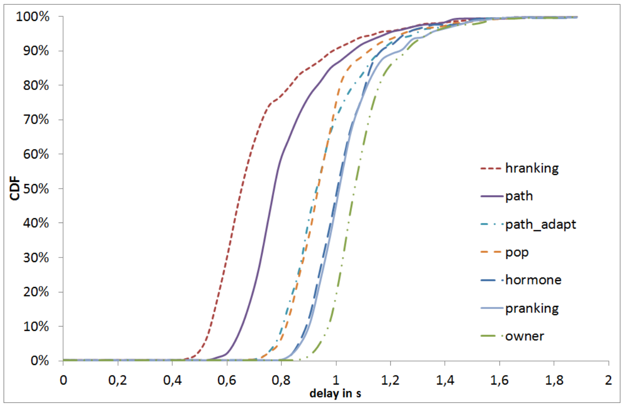

Fig. 2. Delay distribution in the best effort scenario

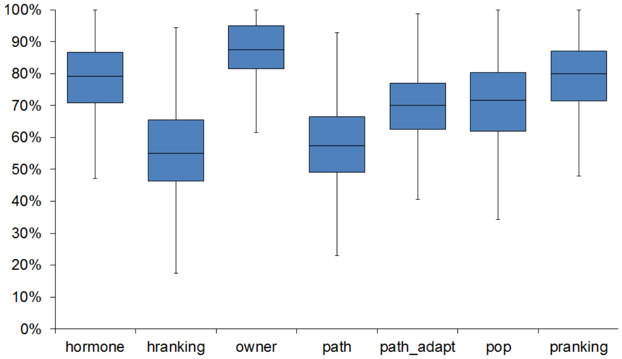

Fig. 3. Utilization comparison

After that the failed requests go fast down to approximately 5-10 %.

An ideal replication mechanism would be lowest in delay and best at utilization. Until now, none of the described replication mechanisms matches this pattern. Thus, there are further strategies necessary. As an example, clean-up mechanisms can be used (such as least recently used) to increase the utilization, by taking care of unneeded units.

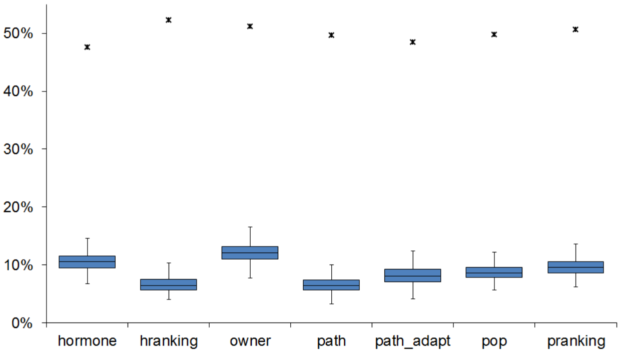

Fig. 4. Failed Request Rate

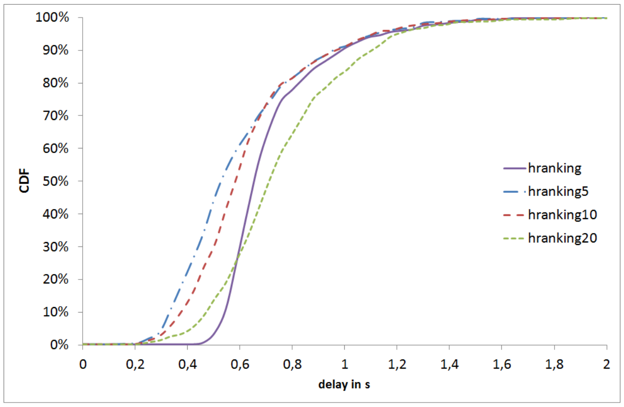

Fig. 5. Delay distribution of Hormone Ranking if 5,10, 20 nodes fail

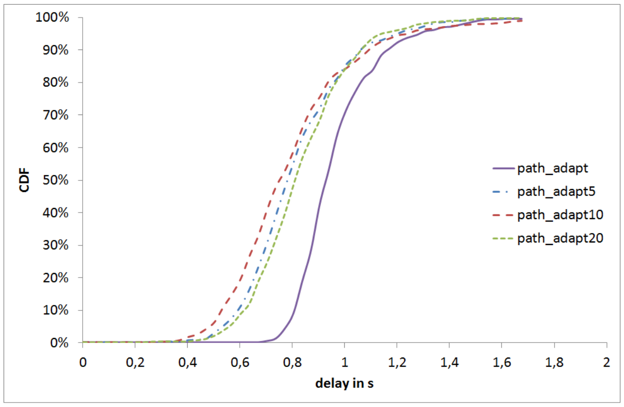

Fig. 6. Delay distribution of path adaptive if 5, 10, 20 nodes fail

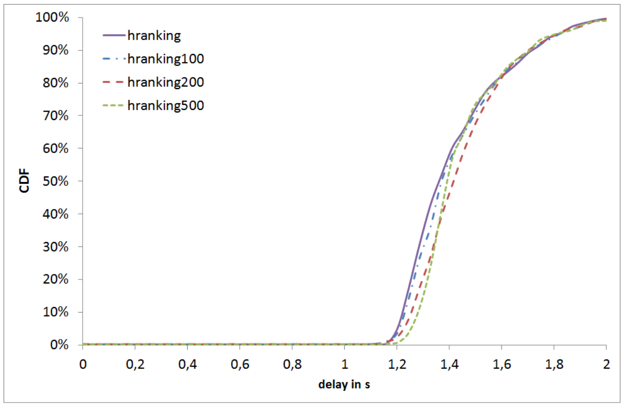

Fig. 7. Delay Distribution of Hormone Ranking if 100, 200, 500 nodes fail

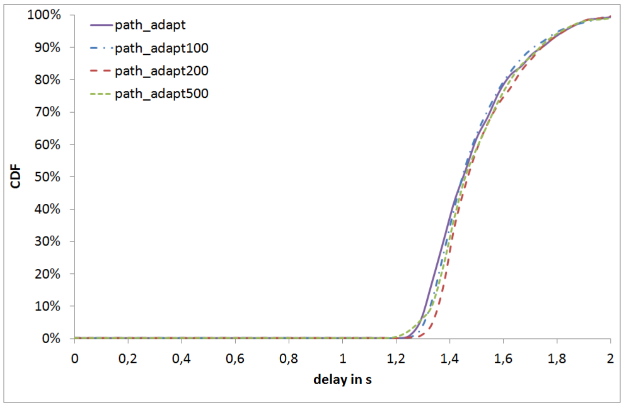

Fig. 8. Delay distribution of path adaptive if 100,200, 500 nodes fail

2) Impact of Peer Churn: We decided to simplify the peer churn scenario to periodical random peer deletion. In case of peer addition and removal the hormone trail is disrupted. By removing nodes from the network such trail disruptions can be simulated. Since hormones are spread over the neighborhood, in both cases a unit can travel over alternative paths. The probability that a request cannot be fulfilled because no fitting unit exists is minimized because of keyword search and replication.
In this section we investigate two of our before mentioned replication mechanisms to show their robustness. We select hormone ranking and path adaptive replication. The first, because it shows the best results in the best effort scenario and the second because it performs similar to path replication, but offers better utilization. Additionally, path adaptive replication is applicable to any bio-inspired delivery approach without further knowledge of the implemented system.
Figure 5 shows the delay distribution of the hormone ranking algorithm. One can see that the replication algorithm and the delivery algorithm are capable of handling loss. At the first sight an interesting delay scenario is shown. The delay of the churn scenarios for 5 and 10 nodes is lower than in the non-churn case. In the non-churn case nodes consume replicas and units until their storage space is full. Therefore, such nodes may block the transport of a unit. If such a node is removed from the network, units have to move over alternative paths. The 20 nodes failure is still capable, however, the delay starts to drop. Path adaptive replication seems to be more robust against churn than hormone replication, which is shown in Figure 6. However, the delay of the 5 nodes failure is higher than the 10 node failure case. This indicates that units are removed from the network for which no replicas exist or nodes that block the network are not removed.
For both replication algorithms a clean-up mechanism would reduce the number of nodes with filled storage and could result in a lower delay. Additionally, the utilization is expected to increase.

B. *Scenario 2: 1000 nodes scale-free network*

In this section we evaluate the applicability of our delivery algorithm for scale-free networks. Again, we use hormone ranking and path adaptive replication. It is shown that the parameters for the 50 node network also work for the 1,000 peer network. This can be explained by the low diameter of the 1,000 nodes network. If the diameter is larger, more hormones have to be created to reach a high number of nodes. The dependency between network diameter and parameters can be solved by adaptively learning how many hormones should be created and spread.
In Figure 7 it is depicted that the delay is increased by around 500 ms in comparison to the small network for the hormone ranking algorithm. Furthermore, if 100, 200 and even 500

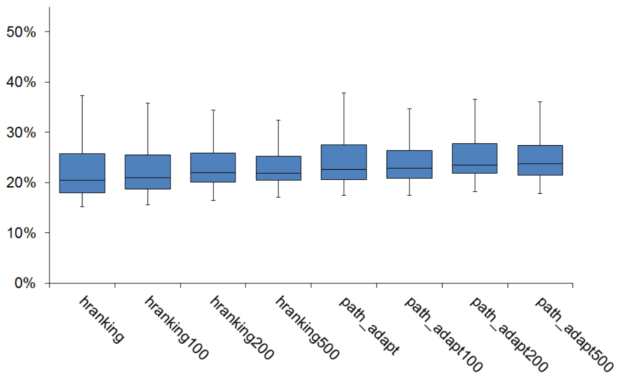

Fig. 9. Failed request rate in case of peer churn

nodes fail over time the delay does not increase considerably. Note that also high degree nodes may fail, because the nodes leaving the network are chosen randomly. Figure 8 shows that with this network structure the algorithm is robust against peer churn. In these two graphs the problem with blocked nodes does not appear. Because of the high number of nodes the unit distribution is more efficient. Both algorithms show a slight increase of request failures in the presence of peer churn (see Figure 9). In comparison to the small network the request failure rate doubles, which can be explained by the larger network diameter.

VII. Conclusion and Future Work

In this paper we compared a number of existing replication from the area of unstructured P2P networks. We evaluated their applicability for a self-organizing bio-inspired delivery algorithm. The delivery algorithm is targeted at highly dynamic networks and uses multi-hop transport of requested content for adaptive replication to reduce search space and improve the robustness. The basis for all our replication mechanisms is owner replication, since we assume multimedia content is consumed for a while and should be available during this time also for other peers.

Uninformed replication such as path replication is resource-wasteful. Path adaptive replication showed to be a good compromise. Thus, for networks without knowledge about current demands, path adaptive replication can be recommended. The evaluation of local popularity replication and popularity ranking showed that popularity aging has to be considered since in dynamic environments the popularity can change quickly. Therefore, we address the hormone as dynamic and latest information about popularity of a content. By additionally including the neighborhood into the decision, the performance increases as well. A clean-up mechanism would further reduce the delay by limiting the number of nodes that block delivery. The mechanisms are as applicable for small random networks as for large scale-free networks. Furthermore, the delivery algorithm in combination with replication is robust against peer churn up to 50 %.

Future work targets the parameters of the delivery algorithm. If the number of peers is unknown in advance there is performance gain potential if the hormones created and diffused as well as the thresholds can be optimized over time. For this an impact study of the single parameters is planned. Additional work regards the investigation of different user request models. We want to define disadvantages and advantages of our algorithm in comparison to direct-download (1) if peers are clustered according to their interests and (2) if the interests are widely spread.

Acknowledgements

The results presented in this paper are part of the research efforts for the SOMA (Self-organizing Multimedia Architecture, http://soma.lakeside-labs.com/) and MESON (Modeling and Engineering of Self-Organizing Networks, http://meson.lakeside-labs.com) project.

References

[1] M. Stiemerling and S. Kiesel, “Cooperative p2p video streaming for mobile peers,” in *Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on*. IEEE, 2010, pp. 1–7.

[2] P. Harsh, R. Chow, and R. Newman, “Gray networking: a step towards next generation computer networks,” in *Proceedings of the 2010 ACM Symposium on Applied Computing*, 2010, pp. 1323–1328.

[3] E. Michlmayr, “Ant algorithms for self-organization in social networks,” Ph.D. dissertation, Technical University Vienna, Austria, 2007.

[4] A. Sobe, W. Elmenreich, and L. Böszörmenyi, “Towards a self-organizing replication model for non-sequential media access,” in *Proceedings of the 2010 ACM workshop on Social, adaptive and personalized multimedia interaction and access*. ACM, 2010, pp. 3–8.

[5] U. Brinkschulte, M. Pacher, and A. Von Renteln, “Towards an artificial hormone system for self-organizing real-time task allocation,” *Software Technologies for Embedded and Ubiquitous Systems*, pp. 339–347, 2007.

[6] F. Dressler and O. B. Akan, “A survey on bio-inspired networking,” *Computer Networks*, vol. 54, no. 6, pp. 881–900, Apr. 2010.

[7] I. Clarke, O. Sandberg, B. Wiley, and T. Hong, “Freenet: A distributed anonymous information storage and retrieval system,” in *Designing Privacy Enhancing Technologies*. Springer, 2001, pp. 46–66.

[8] S. Androutsellis-Theotokis and D. Spinellis, “A survey of peer-to-peer content distribution technologies,” *ACM Computing Surveys*, vol. 36, no. 4, pp. 335–371, Dec. 2004.

[9] H. Yamamoto, D. Maruta, and Y. Oie, “Replication Methods for Load Balancing on Distributed Storages in P2P Networks,” *The 2005 Symposium on Applications and the Internet*, pp. 264–271, 2005.

[10] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker, “Search and replication in unstructured peer-to-peer networks,” in *Proceedings of the 16th international conference on Supercomputing*. New York, New York, USA: ACM, 2002, pp. 84–95.

[11] B. Specification, “Bittorrent Specification.” [Online]. Available: http://www.bittorrent.org/beps/bep_0003.html

[12] E. Cohen and S. Shenker, “Replication strategies in unstructured peer-to-peer networks,” in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, Oct. 2002, pp. 177–190.

[13] E. Leontiadis, V. Dimakopoulos, and E. Pitoura, “Creating and maintaining replicas in unstructured peer-to-peer systems,” in *Euro-Par 2006 Parallel Processing*. Springer, 2006, pp. 1015–1025.

[14] A. Sobe and L. Boszormenyi, “Non-sequential Multimedia Caching,” *2009 First International Conference on Advances in Multimedia*, pp. 158–161, Jul. 2009.

[15] W. Elmenreich, G. Bauer, and H. Kopetz, “The time-triggered paradigm,” in *Workshop on Time-Triggered and Real-time Communication Systems*, 2003.

[16] D. Eppstein and J. Wang, “A steady state model for graph power laws,” *Arxiv preprint cs/0204001*, 2002.

[17] W. Elmenreich and G. Klingler, “Genetic evolution of a neural network for the autonomous control of a four-wheeled robot,” in *Sixth Mexican International Conference on Artificial Intelligence, Special Session*. IEEE, 2007, pp. 396–406.