Biology-Inspired Optimizations of Peer-to-Peer Overlay Networks

Sven Apel, Erik Buchmann
Department of Computer Science
Otto-von-Guericke-University Magdeburg, Germany
{apel,buchmann}@iti.cs.uni-magdeburg.de

ABSTRACT

The aim of this article is to examine the relationship of large-scale Peer-to-Peer (P2P) overlay networks and certain biological systems. In particular, we focus on organization mechanisms that are crucial to adjust and optimize the behavior of large-scale P2P systems in the face of a dynamic environment. We propose to adopt concepts and mechanisms of biological systems in order to extend their capabilities to cope with environmental changes, e.g. a highly dynamic network topology. We introduce the notion of *organic P2P overlay networks* that adopt behavioral and structural characteristics of biological systems. We present a framework that poses as a basis for understanding, investigating, and implementing organic P2P overlay networks. Using a case study we analyze an organic P2P overlay network, AntCAN, that utilizes ant colony optimization to improve the query processing in the face of varying query distribution patterns. Experiments confirm the functional efficiency of this self-organization mechanisms as well as the applicability of our proposed framework.

1. INTRODUCTION

Peer-to-peer (P2P) overlay networks will gain momentum in the near future. This is because of beneficial properties as scalability, reliability, and (economic) efficiency. A major characteristic of P2P overlay networks is the absence of a central coordinator and no global view. Peers are individual entities that act autonomously based on behavioral rules which are predefined by a communication protocol.

Promising application scenarios of P2P overlay networks can be found in the domain of cooperative information systems, networked embedded systems, and Internet computing. These scenarios have in common that the environment is highly dynamic and subject to frequent fluctuations. This does not only concern the network characteristics, e.g. topology, latency, throughput, but merely the whole underlying infrastructure, e.g. available resources, the formation of the particpants, the configuration of peers, etc. Furthermore, the user requirements may evolve over time, especially as with reactive systems that run contineously without shutting down. Software for such scenarios have to adapt to these dynamic changes and evolving requirements in order to work efficiently [19].

Due to their technical and geographical size as well as their high degree of distribution these scenarios require an underlying P2P infrastructure. However, the absence of a central instance demands for coordination and adaptation mechanisms different from the ones used in common distributed systems. The high degree of distribution and the autonomous behavior of the individual peers yield a lot of potentials to achieve *self-organization*. Leading researchers in robotics, artificial intelligence, cybernetics, and organic/autonomic computing proclaim a new kind of self-organizing systems: *organic systems* [9, 22, 19]. Such systems consist of myriads of individual entities that form with their local behavior and activities the overall system. The behavior of the entities is a result of a small, simple set of rules and their limited view of the world. Several researchers see analogies to biological systems, e.g. bee hives, swarms, flocks, ant colonies, the structure of the human brain, etc.

For our part we see analogies between the vision of dynamic self-organizing large-scale P2P overlay networks and biological organic systems, e.g. cells or ant colonies. In this contribution we examine this analogy in order to (1) open a new direction to understand large-scale distributed systems as organic systems, (2) to learn from organic systems, e.g. the organization, adaptation, and healing mechanisms, and (3) to improve the capabilities of large-scale P2P overlay networks in adapting to a highly dynamic environment and evolving requirements.

To address these issues, we introduce a general framework that describes self-organizing P2P overlay networks. We introduce the notion of *organic P2P overlay networks*, general definitions, their analogies to biological systems, their potential benefits, and the challenges to implement organic P2P overlay networks. Furthermore, we present a case study, AntCAN, a *Content-Addressable Network (CAN)* [26] enhanced by a self-organization mechanism based on our view and our framework of organic P2P overlay networks. In this study we focus on the optimization of CAN query processing in order to cope with changing and fluctuating query patterns. We present a mechanism that borrows ideas from ant colonies and swarms to minimize the average query path length. Our experiments confirm that this mechanism improves the overall performance of the CAN. It serves as an example of an organic P2P overlay network that is based on our introduced notion and framework.

2. TECHNICAL BACKGROUND

P2P overlay networks (a.k.a. *P2P data structures*, *distributed hash tables*, *structured P2P overlay networks*, cf. [3]) are a prominent example of P2P systems. They address a core issue of data management research. In particular, P2P overlay networks promise Internet-scale data management: they cope with huge sets of *(key,value)*-pairs, high numbers of parallel transactions, and scores of users. Due to their consequent decentralized organization P2P overlay networks are a promising experimental ground for self-organization mechanisms.

In contrast to popular filesharing systems that use *flooded requests*, P2P overlay networks are based on a *document routing model*. Here, a query is a point in a key space and the query result is the value corresponding to the query point.

The global key space is partitioned in zones, and each zone is assigned to a certain peer. A peer that obtains a query first checks if it can answer it from its own zone. Otherwise, it forwards the query to a peer whose distance to the query point is minimal. This reiterates until the query arrives at the peer that can answer it.

Variants of P2P overlay networks mainly differ in the topology of the key space, contact selection, and routing path selection, i.e., the metric used to find the peer with the minimal distance to the query point.

In *Content-Addressable Networks (CAN)* [26] the key space is a torus of d dimensions. Each peer maintains a contact cache containing at least $2\dot{d}$ neighbors of its zone in the key space. CAN uses the Chessboard distance to determine the peer a query is forwarded to. *CHORD* [34] organizes the data in a circular one-dimensional key space. Messages are forwarded from peer to peer in one direction through the cycle until the peer whose ID is closest to the query key is reached. The contacts are chosen according to its distance, i.e., in a system containing n peers, each peer maintains $log(n)$ contacts in the distance 2^{k-1} with $1 \leq k \leq log(n)$. *Pastry* [28] manages its data in a Plaxton Mesh. The forwarding algorithm is similar to the one of Chord: each peer keeps a table containing $log_{2^b}(n) \cdot (2^b - 1)$ contacts determined by common prefixes of the peer-IDs, while b is an exogenous tuning parameter. *P-Grid* [2] is based on a distributed search tree. Each peer is addressed with a binary string representation of the path from the root to the peer on the leaf of the virtual tree. For each level of the tree, each peer maintains a reference to another peer in the same subtree that branches to a different subtree in the deeper levels. [3] features a detailed survey of these and further approaches.

All of these approaches focus on data management issues. Optimizations are always specific to the certain approaches and are not connected to a broader unifying framework. This complicates the general understanding of optimizing large-scale decentralized distributed systems. Thus, programmers run the risk of reinventing the wheel, again and again. However, current approaches are based on recurring concepts and building blocks, e.g., contact caches and forwarding rules. This allows us to develop a generic framework that connects them to other domains, e.g. biological systems.

3. THE NOTION AND FRAMEWORK OF ORGANIC P2P OVERLAY NETWORKS

This section introduces our notion of self-organizing / organic P2P overlay networks. "Self-organization" refers to the organization concern whereas "organic" expresses the relation to biological systems. We present a framework that describes self-organization in the context of P2P overlay networks. Then, we compare P2P overlay networks to biological systems and explain their potential benefits. The goal of this convergence is to exploit useful characteristics, ideas, and mechanisms of biological systems to improve large-scale P2P overlay networks in optimizing themselves, e.g. coping with a huge number of parallel transactions. The analogy to biological systems can help to understand the decentralized organization and the absence of direct control [21]. Finally, this section discusses design and implementation issues.

3.1 Self-Organization in P2P Overlay Networks

Since P2P overlay networks have no global view or central coordinator, but instead consist of autonomously acting peers, the notion of self-organization must be defined in terms of the participating peers. Thus, we perceive the self-organization of the overall P2P system as the self-adaptation of the individual peers.

Besides the basic data management capabilities of a peer, we introduce the following new properties: Each peer introspects its environment. For introspection a peers uses sensors to observe external characteristics and uses monitors to observe its own behavior and state (see Figure 1). External characteristics are for instance the network latency, topology, and available primary and secondary memory. Internal characteristics are, among others, the number of unanswered queries, frequently used contacts, or the number of incoming messages per time interval. In [4] we give a comprehensive list of internal and external characteristics and discuss the role of their indicators inside a peer.

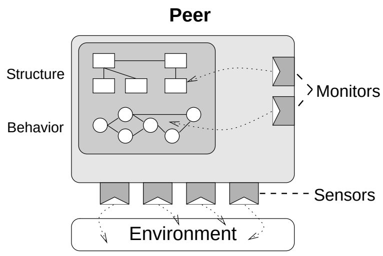

Figure 1: Using monitors and sensors to introspect internal and external characteristics.

Introspecting these characteristics, a peer waits for predefined events. Such events could be simple changes of parameters, the exceeding of a threshold, or more complex context changes, e.g. the connectivity degree, which can also be expressed in terms of parameters and thresholds. When such events occur a peer reacts by adapting its behavior and internal state. This process of introspection and adaptation can be expressed by event-condition-action rules or state machines [4].

In addition to thie simple introspection and adaptation scheme, peers have to coordinate themeselves in a decentralized way by exchanging information. Such pieces of information are load information, security certificates, content replicas, contacts, reputation information, etc. (see [10] for further scenarios). In order to realize such coordination we propose a swarm-like meta-data dissemination to deliver coordination information from peer to peer (see Figure 2).

Each peer emits coordination information by attaching meta-data items to messages that are sent out anyway, e.g. forwarded query requests. Thus, with each incoming message each peer receives a set of meta-data items. The peers cache these items temporarily. When a peer sends a message it selects a set of cached meta-data items and attaches them to the outgoing message. The policies which items are cached and which ones are selected for a certain outgoing message are powerful tuning parameters for organic P2P overlay networks [10].

In [10] we explain that such decentralized dissemination

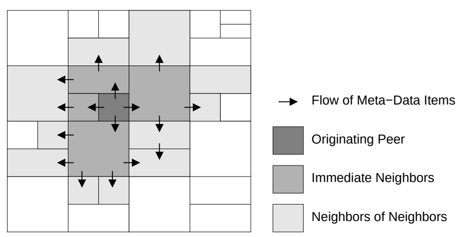

Figure 2: Swarm-like meta-data dissemination.

mechanisms are more efficient than common mechanisms (e.g. peers send dedicated messages to well known sets of addressees) with regard to resource consumption, network traffic, time to delivering information, etc. In [4] we show that a coordination mechanism is essential for certain kinds of self-organization mechanisms, e.g. load balancing, reputation mechanisms, healing mechanisms, etc.

In a nutshell, we perceive self-organization in P2P overlay networks as self-adaptation (introspection and adaptation) of the individual peers, but all peers in concert (coordination).

3.2 Relationship to Biological Systems

Our investigations in optimizing large-scale P2P overlay networks have revealed that there are several similarities between organic systems, known from life sciences and self-organizing P2P overlay networks. The definition of Kelly regarding mandatory characteristics of organic systems [21] can be applied one-to-one to self-organizing P2P overlay networks, i.e. they hold for both, biological organic systems and organic P2P overlay networks:

- There is no central coordinator or global view.
- The individual entities that constitute the global system act autonomously.
- The individual entities are highly connected.
- The causal relations and dependencies between the individual entities are of a complex nature.

Organic systems have several benefits compared to common centralized engineer or computer systems: They are adaptive, evolve over time, are fault-tolerant and reliable, and have no limited pre-defined behavioral patterns that degrade their flexibility.

Since our notion of self-organizing P2P overlay networks has these commonalities with biological systems, we should learn from biological systems in order to exploit their capabilities. Several studies have revealed that a lot of mechanisms of P2P overlay networks directly correspond to mechanisms of biological systems [6, 7, 4]. So, the mentioned mechanism for disseminating coordination information in form of meta-data through the P2P network is similar to marching ants that emit pheromones or cells that emit enzymes to notify their immediate environment. The mechanisms to constrain the life time and the scope of meta-data items correspond to biological mechanisms where the cells destroy unused enzymes and ant pheromones evaporate over time.

Another example is a load balancing mechanism, e.g. [16]: A peer observes its own load and the load of its immediate neighbors. In the case of an overload-state it moves its data to its under-loaded neighbors. The load information is distributed – as with pheromones and enzymes – in a swarm-like manner. Each peer acts according to a simple set of rules that defines which reactions take place in the case of which events occur. The global order – a balanced load – *emerges* from the individual rule-based behavior of the participating peers. However, a prerequisite is an adequate set of rules and well-adjusted parameters settings, e.g. cache size, load threshold, number of neighbors, etc. The *emergence of order* is a major characteristic of organic systems [21].

A main goal of engineering organic systems is to find a set of rules and a set of parameter settings to switch between order and disorder, e.g. a balanced and an unbalanced load, under certain conditions. A small modification in these sets of rules and parameters can lead to massive behavioral changes of the overall system. We understand this switching as *phase transition*: Phase transition phenomena are defined as phenomena where a macroscopic parameter (global parameter) of the system changes significantly when a microscopic parameter (local parameter) approaches an uncertain critical point [33]. We argue that finding appropriate rules, parameters, and phase transitions is a main task of designing, engineering, and tuning organic P2P overlay networks.

3.3 Potential Benefits and Limitations

The goal of our work is to improve the self-organization capabilities of P2P overlay networks. Common P2P overlay networks have several drawbacks when coping with highly dynamic environments, e.g. the Internet:

Static structure: Overlay networks use key spaces with predefined properties, e.g. number of dimensions, distance function, topology (hyper cube, tree, ring, etc.) Furthermore, the partition of the key space is fixed during runtime, except in the case of leaving and joining peers. It would be better to be flexible: Rearranging the key space partitioning at runtime would allow to cope with load fluctuations or unreliable peers.

Predefined behavior: With common approaches, the behavior of peers is based on a simple but static protocol that concerns only the main data management tasks. It does not take other dynamic properties into account. Examples for that kind of properties are load distribution, overlay/physical network topology, or the average query response time. Several studies (e.g. [13, 7, 11]) indicate that altering the behavior based on the environment would be effective, e.g. adapting the forwarding policy, the neighbor and contact selection, or the replication degree, etc.

The advantages of exploiting mechanisms and borrowing ideas from biological systems are straightforward. Biological systems are extremely successful in coping with scaling volumes of information, load, or participants. Reynolds states: *"There is no evidence that the complexity of natural flocks is bounded in any way. Flocks do not become 'full' or 'overloaded' as a new bird joins. When herring migrate toward their spawning grounds, they run in schools extending as long as 17 miles and containing millions of fish [27]."* We perceive these circumstance and the close relation of organic P2P overlay networks to biological systems as a chance to improve the capabilities of P2P overlay networks in managing frequent fluctuations of the environment and evolving requirements.

Besides these potential advantages also some drawbacks arise: Organic systems cannot be fully controlled, their behavior is not really predictable, and their adaptation and evolution cannot be completely formally specified, verified, and validated (in a usable and practicable way). This is because the interaction pattern between the individual entities becomes by orders of magnitude more complex than the behavior and structure of these entities stand-alone. The overall global behavior emerges from the simple individual behaviors and their complex interactions.

It is well known that an important indicator for organic systems is the circumstance that the prediction of their behavior is more time-consuming and complex than running the systems themselves [21]. This is a clear limitation of our view on organic P2P overlay networks. However, we argue that current approaches to manage and specify large-scale distributed are very limited. In our opinion it is an inherent property of organic P2P overlay networks to be less predictable. That means it is not a matter of the view but of the complexity of the application scenario.

Mary Shaw argues that thinking in specifyable programs is nowadays – in the time of heterogeneous distributed systems consisting of myriads of components – not the right direction [31]. Practical systems would be better served by development models that recognize the variability and unpredictability of the environment.

Our goal is to tame the low predictability by defining lower and upper bounds, working with statistical models and guarantees, and exploiting emergent behavior in defined boundaries and under certain circumstances.

3.4 Design and Implementation Issues

Developing organic P2P overlay networks is challenging. During the development process the programmer has to do several tasks concerning different fields. Our framework for describing organic P2P overlay networks helps to order the tasks:

1. The programmer analyses potential application scenarios in order to determine which kind and degree of self-organization is needed. This includes the specification of observed characteristics and triggered reactions.
2. The programmer specifies a set of rules for adapting the behavior of the individual peers.
3. Furthermore, the programmer chooses a coordination mechanism to disseminate meta-data over the network (see [10]).
4. Afterwards follows the implementation of rules by adding introspection, adaptation, and coordination code. [10, 4] propose several techniques to ease this step.
5. After the implementation the programmer experiments to get feedback for adjusting rules and parameters (to find potential phase transitions) in order to achieve the desired global behavior.

The remaining article applies our framework and these steps to a case study, an ant colony optimization mechanism to improve the query processing of CAN in dynamic environments.

4. CASE STUDY: ANTCAN

In their original proposals the P2P overlay networks reviewed in Section 2 use a deterministic ruleset to choose its contacts. This leads to a scalability of typically $log(n)$, i.e., obtaining an arbitrary query result invokes $log(n)$ peers on average, while the system contains n peers.[1]

However, deterministic rules leave aside that

- peers want to obtain queries at different rates,
- some keys are more popular than others and
- peers can be less reliable.

In the following we want to show that extending a CAN by a self-organization mechanism for contact selection reduces the number of peers invoked for each query. The envisioned mechanism is supposed to optimize the query processing in face of a highly dynamic environment, e.g. with unreliable peers or with limited connectivity.

We call our CAN enhanced by this mechanism AntCAN. Its core idea is to adopt the path selection strategy from ant colonies for CANs. Basically, each ant marks its way to a food source with pheromones. After some time, efficient paths, e.g., routes with shortcuts, 'smell' more than inefficient ones. Since ants tend to follow those routes that have the strongest odor, they usually find the most efficient paths. Pheromones disappear over time, thus unneeded paths lose their marking.

The 'ant routing algorithm' (a.k.a. Ant Colony Optimization [14]) is very similar to the forwarding problem in P2P overlay networks: in both cases we have small entities that are fully autonomous. Each ant as well as each peer has a very limited point of view, i.e., a global view to the world is not available. And the problem is beyond the horizon of a single entity, but it can be solved *cooperatively*!

In order to adopt the ant system to a CAN, we require additional data structures (see Figure 3). In the original proposal, each peer knows its immediate neighbors in the key space only. Now the peers have the following new properties:

- Peers are able to keep a *limited number of other contacts* in addition to its neighbors in their contact caches.
- Each contact is assigned with a *pheromone value.*
- In addition, each *query result contains the list of peers* which have forwarded the query.

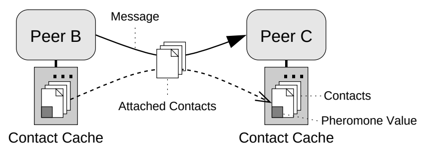

Figure 3: CAN with contact cache.

Thus, the query forwarding algorithm considers not only the neighbors, but the set of additional contacts as well. The crucial question that arises now is: how obtains each peer an optimal set of contacts? Here, ant colony optimization comes into play as follows:

The query answer carries not only the answer itself, but also information about all peers that have forwarded the query message before. A peer that obtains information about the path on that way now updates its contact cache: If a contact is already present in the cache, its pheromone value is increased. Previously unknown peers are added and marked with a base pheromone value. If the number of peers in the contact cache exceeds a certain limit, the contact with the least pheromone value is removed. Finally, the pheromone values of all contacts are decreased by a small amount, as with biological systems. The peers remove contacts with a pheromone value below zero.

From a different point of view, the messages containing queries and query answers are our 'ants'. They mark their ways with pheromones, which disappear over time. The major difference to the behavior of ant colonies in the nature is that here the ants carry information about the path travelled. This is why we want the ants to make 'jumps' over longer distances. In other words, in certain situations our messages shall be forwarded directly to remote peers instead of being handed over from neighbor to neighbor.

4.1 Connection to the Framework

This simple self-organization mechanism can be classified into our framework as follows:

Introspection: Each peer observes incoming messages. With our algorithm, the messages contain contact information of other peers. In this way each peer becomes aware of its environment.

Adaptation: By observing the message traffic each peer builds up a contact cache. A peer uses this cache to adapt its behavior, in particular to modify the forwarding policy. Each outgoing message is passed not to the nearest neighbor, but to the nearest peer in general.

Coordination: The attached contact information corresponds to the swarm-like coordination concern. Peers may use different policies for selecting and attaching their contacts to messages.

To realize the pheromone-based forwarding each contact is marked with a pheromone value. Pheromone values are a second characteristic that is introspected by the peers. They

[1] For CAN with d dimensions and n peers the average path length is $\frac{d}{4}\sqrt[d]{n}$. With $d = \frac{log_2(n)}{2}$ CAN have also a scalability of $log(n)$ [26].

change over time and lead on the one hand to preferences in choosing paths through the P2P network and on the other hand to the automatic removal of less used contacts.

A further improvement of this mechanism would be the introspection of other characteristics, e.g. avergage query results per time unit, average query response time. This information can be used to adjust the cache size, the pheromone life time, etc., which are free parameters. A limited cache size saves resources; but an undersized cache reduces the positive effect on the query path length. However, in our first experiments we focus on the simple variant of the AntCAN, i.e. without additional introspection of such characteristics.

4.2 Evaluation

Our AntCAN forwarding algorithm offers three optimization parameters:

1. the number of additional contacts (c),
2. the increment of the pheromone values for the contacts already present in the contact cache (i), and
3. the decrement value that makes the pheromones disappear (j).

To simplify the experimental setup we use the increment parameter also as base for initial pheromone values.

We now determine the extent for these parameters by means of experiments. We use a Java-based CAN implementation of our own that is fully operational and allows us to run experiments with a large number of peers on a Linux cluster consisting of 32 hosts equipped with 2 GHz CPUs, 2 GBytes RAM and 100 MBit Ethernet each. [12] gives a detailed description of our experimental setup.

Given this runtime environment, we ran a four-dimensional CAN consisting of 100,000 peers. In order to have meaningful results we experimented with 5,000,000 queries. One setup simulates the worst case. Here, the query keys are equally distributed over the key space, i.e., there is no key that is more popular than others. Two setups with Gaussian-distributed query keys cover more realistic cases. The mean value in both Gaussian setups is the center of the 4d-hypercube of the key space, i.e., more queries go to the nodes in the center than the peers at the 'sides'[2]. Examples for this setting come from the WWW. Here, *www.google.com* and URLs nearby (e.g., *www.google.com/search?q=something*) are much more popular than, for example, *www.dogbreed.com*. The variances of the Gaussian setups are the diameter of the key space divided by 3 and 100, simulating an environment containing a moderate and an extreme hot spot. Figure 4 illustrates the key distribution for one dimension of the key space.

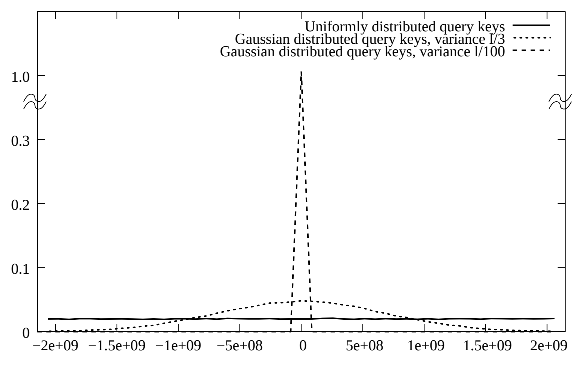

Figure 4: Distribution of query keys used in the experiments.

[2]The key space is a torus, thus 'sides' means the ranges in the key space where the keys wrap between 0 and 1.

Additional Contacts. First we want to examine the number of additional contacts. We anticipate that even a small number of additional contacts decreases the average path lengths, and the ant algorithm performs better in the setting with the gaussian-distributed query keys. We set $i = 1$ and $d = 0.1$ arbitrarily, and ran a series of experiments with different sizes for the contact cache. Figure 5 now graphs the results of these experiments. The x-axis shows the number of additional contacts, i.e., 0 stands for the original CAN proposal where only the neighbors are kept in the contact cache[3]. The y-axis is the average path length.

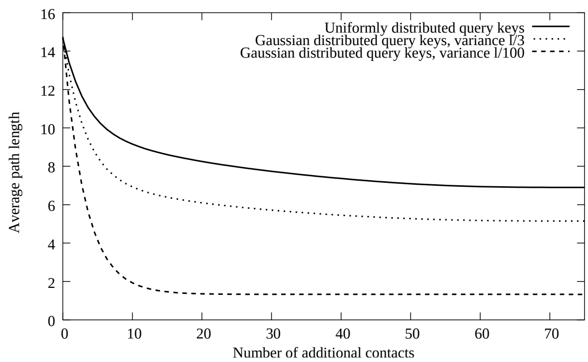

Figure 5: How does the number of additional contacts affect the average path length?

The experiments confirm our expectations. Obviously, having no additional contacts in the cache leads to the average path length given by the original CAN proposal ($d/4 \cdot n^{1/d}$). But having 8 additional contacts reduces the path length by 1/3 in the worst-case setting with equally distributed queries! In settings with gaussian distributed query keys, the path length is reduced further. In addition, Figure 5 tells us that a very large cache size does not help very much, i.e., the marginal utility falls with increased cache size. A contact cache containing 8 neighbors plus 20 additional contacts seems to be viable for our settings. A dynamic cache size that adapts to the current workload in order to find a trade-off between minimal average path length and contact cache size is part of further work.

Increment/Decrement for the Pheromone Value. We now determine the extent of the increment i and decrement j for the pheromone values. Clearly, the values depend on each other, e.g., an increment of 5 and a decrement of 1 yields the same result as an increment of 0.5 and a decrement of 0.1. Therefore, we set the decrement to the arbitrary value $j = 0.1$, and vary the increment. The number of additional contacts is set to $s = 50$.

Figure 6 shows the results of our experiments. Again, the y-axis graphs the average path lengths. The x-axis is the extent of the increment i for the pheromone value. It is obvious that $i < j$ leads to a setting where any additional contact is immediately removed. Therefore, all three graphs start at an average path length comparable to the case without additional contacts. The figure tells us that peers can always better off by using increments substantially larger than the decrement value. In particular, our observations have shown that the average path length is minimal if the ratio of increment / decrement meets the number of additional contacts, i.e., $i/j = s$. In our experiment, $i_{opt} = s \cdot j = 50 \cdot 0.1$. Larger ratios yield no effect.

[3]Basically, each peer in a 4-dimensional key space keeps track of at least 8 neighbors.

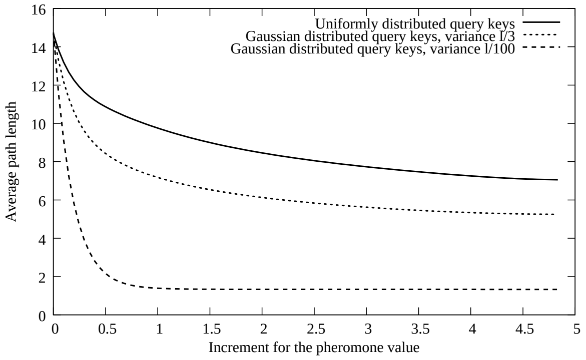

Figure 6: Relation between the increment of the pheromone value and the average path length.

5. RELATED WORK

Several work tries to adopt concepts and mechanisms of biological systems, e.g. swarm-intelligence, ant colony optimization, etc. Since this field of research has an interdisciplinary touch we review only close related and representative work. We distinguish between mechanisms adopted from biological systems and programming models to develop organic computing systems.

Biological Mechanisms. The *BISON* project [20] explores biology-inspired techniques for self-organization in dynamic networks. Their results and findings are an important basis for our project to provide a general framework for modular self-organization mechanisms for P2P overlay networks. The *AntHill* project [7] has a similar focus. It aims on design, implementation, and evaluation of P2P applications based on ideas such as multi-agent and evolutionary programming, however, without considering overlay structures.

Kulendik et al. proposes a distributed control and synchronization mechanism that exploits collective coordination and decision making [24]. Furthermore they propose to utilize available processes inside a target system to coordinate the behavior. Although they focus on chemical and physical effects this idea is related to our coordination concern that exploits in the case of AntCAN the existent message delivering mechanism of CAN.

Arabshahi et al. present an adaptive routing mechanism for wireless ad-hoc networks that uses swarm intelligence [5]. They discuss two specific algorithms that show promising results in this respect [8].

Di Caro et al. present *AntNet*, an adaptive approach to routing tables learning in packet-switched communication networks [15]. They show that their algorithm based ant colony optimization outperforms common internet routing algorithms.

Schoonderwoerd et al. apply ant colony optimization to improve load balancing in telecomunication networks [29].

However, all these approaches propose algorithms for specific problems in different fields. We have introduced a general framework for understanding, investigating, and implementing organic P2P overlay networks. Future algorithms can profit from this basis.

Programming and Architectural Models. Shaw proposes a new kind of programming model that takes dynamic behavior of nowadays software systems into account [30]. The model is based on a process-oriented view and feedback loops that reflect temporal concerns. Our proposal of introspection, adaptation, and coordination goes one step beyond towards massive distributed systems, but not as deep into the field of programming paradigms.

Furthermore, Shaw states that engineering nowadays and future complex systems requires a different view on software [31]. She proposes to softening precision in order to avoid brittleness.

George et al. present a novel programming model that is inspired by biological cell systems [18]. Basically, they utilize a state machine approach to describe local actions in case of errornous system behavior. Our approach of introspection and adaptation has a similar model based on events, states, and actions.

The early pioneer work on reflection and reflective software architecture is a basis for our view on introspection and self-adapting peers [25, 32, 23].

In [4] we have shown how novel design and implementation techniques can help to build flexible self-organizing overlay networks. In [10] we discuss the importance of meta-data dissemination as decentralized coordination mechanism and its efficient modular implementation.

Last but not least the great visions of autonomic [17], proactive [35], and organic computing [1] signpost to a new understanding of software systems. With our approach regarding P2P overlay networks, we go one step into this direction.

6. CONCLUSION

The AntCAN study shows that ideas from biological systems can be exploited to optimize the behavior of large-scale P2P overlay networks in the face of a dynamic environment. Our framework poses as a basis for describing the function, the architecture, and the implementation of self-organization mechanisms on top of P2P overlay networks. We argue that a significant part of self-organization mechanisms for P2P overlay networks can be expressed using three standard concerns: introspection, adaptation, and coordination. The strict decentralized architecture and the swarm-like information distribution opens the door to adopt mechanisms and algorithms of biological systems. In the case of AntCAN, we borrowed ideas from ant colonies, in particular the mechanism of dispersing pheromones to signpost paths. Indeed, one can say that another synthetic algorithm could achieve the same results; but we argue that the strict adherence to the decentralized organization paradigm and the biology-inspired view of autonomous rule-based peers leads to a deeper comprehension of organic P2P overlay networks.

We argue that our framework in general as well as the optimization of the query processing are applicable to other P2P overlay networks because they do not depend on proprietary mechanisms but on general concepts of overlay networks.

Experiments have shown that an implementation of a biology-inspired algorithm optimizes the behavior of a P2P overlay network, especially under real-world circumstances. Furthermore, the results show that even in this simple case study several parameters decide over the overall system behavior, in particular the contact cache size and the pheromone increment/decrement values. We perceive these parameters as tuning leverages to adapt a P2P overlay network to a specific application scenario. In future work we want to investigate a mathematical analysis and the runtime adaptation of these tuning-parameters.

7. REFERENCES

[1] European Organic Computing Initiative. http://www.organic-computing.de.

[2] K. Aberer. P-Grid: A Self-Organizing Access Structure for P2P Information Systems. In *Proceedings of 6th Conference*

on Cooperative Information Systems, 2001.

[3] S. Androutsellis-Theotokis and D. Spinellis. A Survey of Peer-to-Peer Content Distribution Technologies. *ACM Computing Surveys*, 36(4), 2004.

[4] S. Apel and K. Böhm. Self-Organization in Overlay Networks. In *Proceedings of 1st CAISE'05 Workshop on Adaptive and Self-Managing Enterprise Applications*, 2005.

[5] P. Arabshahi et al. Adaptive Routing in Wireless Communication Networks using Swarm Intelligence. In *Proceedings 9th AAAI Communications Satellite Systems Conference*, 2001.

[6] O. Babaoglu et al. Design Patterns from Biology for Distributed Computing. In *Proceedings of the European Conference on Complex Systems*, 2005.

[7] O. Babaoglu, H. Meling, and A. Montresor. Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems. In *Proceedings of the 22th International Conference on Distributed Computing Systems*, 2002.

[8] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artifical Systems*. Oxford University Press, 1999.

[9] R. A. Brooks and A. M. Flynn. Fast, Cheap and Out of Control: A Robot Invasion Of The Solar System. Technical Report AIM-1182, MIT, 1989.

[10] E. Buchmann, S. Apel, and G. Saake. Piggyback Meta-Data Propagation in Distributed Hash Tables. In *Proceedings of Conference on Web Information Systems and Technologies*, 2005.

[11] E. Buchmann and K. Böhm. FairNet – How to Counter Free Riding in Peer-to-Peer Data Structures. In *Proceedings of 9th Conference on Cooperative Information Systems*, 2004.

[12] E. Buchmann and K. Böhm. How to Run Experiments with Large Peer-to-Peer Data Structures. In *Proceedings of 18th Parallel and Distributed Processing Symposium*, 2004.

[13] G. Canright, A. Deutsch, and T. Urnes. Chemotaxis-Inspired Load Balancing. In *Proceedings of the European Conference on Complex Systems*, 2005.

[14] A. Carbonaro and V. Maniezzo. The Ant Colony Optimization Paradigm for Combinatorial Optimization. *Advances in Evolutionary Computing: Theory and Applications*, 2003.

[15] G. D. Caro and M. Dorigo. Ant Colonies for Adaptive Routing in Packet-Switched Communications Networks. In *Proceedings of 5th Conference on Parallel Problem Solving from Nature*, 1998.

[16] K.-H. Deutinger. Modulare Mechanismen zur Lastbalancierung in Peer-to-Peer Systemen. Master's thesis, Department of Computer Science, University of Magdeburg, Germany, 2005.

[17] A. Ganek and T. Corbi. The Dawning of the Autonomic Computing Era. *IBM Systems Journal*, 42(1), 2003.

[18] S. George, D. Evans, and L. Davidson. A Biologically Inspired Programming Model for Self-Healing Systems. In *Proceedings of 1st Workshop on Self-Healing Systems*, 2002.

[19] F. Heylighten and C. Gershenson. The Meaning of Self-Organization in Computing. *IEEE Intelligent Systems*, May/June 2003.

[20] M. Jelasity, A. Montresor, and O. Babaoglu. A Modular Paradigm for Building Self-Organizing Peer-to-Peer Applications. In *Engineering Self-Organising Systems: Nature-Inspired Approaches to Software Engineering*, 2004.

[21] K. Kelly. *Out of Control: The New Biology of Machines, Social Systems, and the Economic World*. Basic Books, 1997.

[22] J. O. Kephart and D. M. Chess. The Vision of Autonomic Computing. *IEEE Computer*, 36(1), 2003.

[23] G. Kiczales and J. D. Rivieres. *The Art of the Metaobject Protocol*. MIT Press, 1991.

[24] O. Kulendik, S. Kornienko, and P. Levi. Collective Decision Making using Natural Self-Organization in Distributed Systems. In *Proceedings of Conference on Computational Intelligence for Modelling Control and Automation*, 2001.

[25] P. Maes. Concepts and Experiments in Computational Reflection. In *Proceedings of Conference on Object-Oriented Programming Systems, Languages and applications*, 1987.

[26] S. Ratnasamy et al. A Scalable Content-Addressable Network. In *Proceedings of SIGCOMM Conference*, 2001.

[27] C. W. Reynolds. Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics*, 21(4), 1987.

[28] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In *Proceedings of Conference on Distributed Systems Platforms*, 2001.

[29] R. Schoonderwoerd et al. Ant-Based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, 5(2), 1997.

[30] M. Shaw. Beyond Objects: A Software Design Paradigm Based on Process Control. *ACM Software Engineering Notes*, 20(1), 1995.

[31] M. Shaw. "Self-Healing": Softening Precision to Avoid Brittleness. In *Proceedings of 1st Workshop on Self-Healing Systems*, 2002.

[32] B. C. Smith. Reflection and Semantics in a Procedural Language. Technical Report 272, MIT Laboratory for Computer Science, 1982.

[33] H. E. Stanley. *Introduction to Phase Transition and Critical Phenomena*. Oxford University Press, New York, 1971.

[34] I. Stoica et al. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of SIGCOMM Conference*, 2001.

[35] D. Tennenhouse. Proactive Computing. *Communications of the ACM*, 43(5), 2000.

8. AUTHOR INFORMATION

Sven Apel received the diploma in computer science from the University of Magdeburg, Germany. Afterwards, he worked as IT consultant at the METOP research institute at Magdeburg, Germany, Currently, he is a PhD student and research associate at the Institute of Technical and Business Information Systems at the University of Magdeburg. His main research interests are distributed systems and novel software engineering methods to develop distributed systems.

Erik Buchmann received the diploma in business information technology from the University of Magdeburg, Germany. Currently, he works as a PhD student and research associate at the Institue of Technical and Business Information Systems at the University of Magdeburg. His main research interests are Peer-to-Peer systems and their optimizations.