Discovering Bounds of Performance of Self-Organizing Content Delivery Systems

Tibor Szkaliczki
Computer and Automation Research Institute of the Hungarian Academy of Sciences, Budapest, Hungary
Email: szkaliczki.tibor@sztaki.mta.hu

Anita Sobe
Institute of Networked and Embedded Systems
Alpen-Adria-Universität Klagenfurt, Austria
Email: anita.sobe@aau.at

Laszlo Böszörmenyi
Institute of Information Technology
Alpen-Adria-Universität Klagenfurt, Austria
Email: laszlo@itec.uni-klu.ac.at

***Abstract*—Self-organizing methods can efficiently search, route and replicate content in complex, dynamic networks. Furthermore, they make the assumption that decisions of the nodes of the networks rely only on local information and therefore the global optimum is not known. For evaluation purposes, however, it is important to compute the global optimum to serve as a theoretical bound. In this paper we define a formal model describing the problem of content placement and use an integer linear programming (ILP) based optimization method. With this method we discover the Quality of Service (QoS) bounds of a self-organizing content delivery system. We further demonstrate how to balance between run-time complexity and accuracy of the model by applying a use case.**

I. Introduction

Delivering content over large-scale dynamic networks is challenging. If the content consists of multimedia streams, additional challenges arise, due to their timely characteristics. Furthermore, the number of devices producing and consuming multimedia content has increased. Apart from using social networks, it is very cumbersome to share content, e.g., during a social event among a large number of visitors. If we connect the visitors' devices without infrastructure, adaptive and robust measures are required to provide the quality expected by the users. Traditional, centralized solutions are not an option, due to their poor scalability.

In our previous work we developed a hormone-based delivery model [1], which showed promising results. It is robust and has a high hit rate, because of a smart replication mechanism [2]. The introduced model targets multimedia delivery in dynamic networks with quality of service (QoS) restrictions and storage limits. Due to the lack of comparable models, we evaluated our model by simulation under different QoS conditions in comparison to a shortest path routing algorithm. Another evaluation was done by applying our model to the use case event ”Long Night of Research” at the Alpen-Adria-Universität Klagenfurt, Austria[1]. The results showed the applicability of our model – we want to quantify more precisely how good our actually is. This can be best achieved if the global optimum is known.

In this paper, we define a formal model for the optimization problem and calculate the optimum by using Integer Linear Programming (ILP) techniques, in order to get an absolute bound. Content placement combines several other problems including routing, replication, scheduling, etc. The content items should be routed to the requesting nodes. Furthermore, they are replicated in the network in order to reduce the delay of serving the requests expected in the future. Transferring content items in the network should be scheduled in such a way that none of the links are overloaded.

Although the development of the optimization method was motivated by the hormone-based distribution algorithm, the recommended techniques are general and can be applied to validate other methods including distributing content items other than multimedia and also work for routing methods. The optimization approach is not suitable for real-time usage because of long running times. However, it can calculate optimal network and service metrics in acceptable time for a reasonably sized problem instance. Similar to a simulation tool, this optimization method can be applied to study the behavior of the system with different load, topology and network settings. The results characterize the content delivery system independently from the current implementation of the delivery algorithms. The developed optimization tool can help to discover the performance limits of specific content delivery problems.

The remainder of this paper is organized as follows: In Section II, an overview of the related work is presented. The optimization problem is described in Section III. Section IV introduces the developed optimization model in detail. Application steps of the model for calculating bounds for a content placement system are described in Section V. In Section VI, a case study is presented, showing how to use the model to achieve a correct bound within a long but limited running time. Section VII concludes this paper.

II. Related Work

Content delivery in dynamic networks (e.g., ad-hoc networks) combines routing, replication and search of content. Self-organizing algorithms, such as ant-based solutions are well-suited because they only require local knowledge, are robust, adaptive and scalable. An example for such a system is *SemAnt* proposed in [3], which extends AntNet [4] for query routing. A query is represented by a

[1]Documented at the project web page http://soma.lakeside-labs.com/

number of ants. Since in a peer-to-peer system the target node is typically unknown, a time-to-live (TTL) parameter for forward ants is introduced. A search procedure terminates, if a resource is found or if the TTL is reached. SemAnt has been evaluated by arguing the similarity to k-random walk [5] and shows the improvement over k-random walk. Similar to the search problem, researchers adapted AntNet for routing purposes. Hossain et al. [6] propose two algorithms to route content in resource constrained networks. *Improved AntNet* is similar to the basic AntNet, except that forward ants track nodes they have already visited, i.e., perform cycle detection. *Pharaoh* additionally introduces negative pheromones if cycles are detected to avoid unnecessary movement of other forwarding ants. Here, the researchers compare their approaches to *AntNet*. As described before, in most cases the evaluation of self-organizing systems for routing and search are compared to each other or to state-of-the-art algorithms. The global optimum that could serve as the ground truth is usually not known.
The ILP model is a common approach to describe and solve computationally hard problems including ones related to content placement (e.g., [7], [8], [9], [10]). ILP solvers can manage problems with limited sizes. Modeling a QoS-aware self-organizing system network requires a higher complexity from the model than those described in the literature.
Many routing methods build on the simplification that the delay across a link is fixed. This simplification allows to concentrate on the shortest path routing, which is a computationally easy task [11], [12]. However, this approach may lead to unstable and suboptimal routing, because in real systems, the delay depends on the load of the link and, as a consequence, on the route assignment itself. Nevertheless, the length of the shortest paths can serve as a simple lower bound for the delays.

III. Optimal Content Placement Problem

A. Overview

The basis for our proposed formal model is an artificial endocrine algorithm introduced in [1], which relies on the notion of nodes and units (atomic content objects). The main idea of the algorithm is to guide content units to the right places by spreading "hormones", which attract required units and guide them on a QoS-aware path to the requester. This is different to ant-based routing algorithms, which only guide queries through the network. The hormone-based algorithm consists of two phases: search and delivery. Search consists of the hormone creation for indicating the demand for a unit, and hormone diffusion to spread the demand over the network on multiple paths. In the delivery phase units are moved towards the place of the query, along the path marked by the hormones. The stronger the hormone on a node the more likely the unit will move towards this node. The algorithm further supports replication mechanisms based on local knowledge [2], i.e., if the hormone concentration in the neighborhood is high a replica is likely. The combination of hormone diffusion on multiple paths and replication make the system robust against churn [13].
The content distribution problem can be regarded as a multi-objective optimization problem in the general case. The optimization model serves to find lower/upper bounds for the minimum/maximum values of the examined performance metrics. A simplification is allowed only if the optimum remains a valid bound in the modeled distributed system. The optimization result serves as a valid bound if it is lower than or equal to the minimum or higher than or equal to the maximum of the modeled system. The formal optimization model complies with the same constraints as the self-organizing method, but it has an exact knowledge about the state of the entire system. Content delivery is a complex problem and can be formulated in different ways, depending on the aim of the investigation. Each decision in the model formulation may have serious effects on the accuracy and performance of the optimization. The optimization process has to balance between accuracy and running-time of the model.
For the evaluation, we assume a global view on the system. The problem input includes the network graph, initial location of the units, storage capacities of the nodes, link bandwidths, unit sizes and series of requests. The input has been generated by the simulator of the hormone-based algorithm. The optimization goal includes the average delay and the number of units that do not arrive in time (e.g., if videos are transmitted, the frames have to arrive within a given deadline to provide proper quality). In the following paragraph, the main features of the model and the described system are summarized.
The model is deterministic. Several requests arriving at the same time are processed in batch mode. Queues are formed for each link where the incoming content units can wait if the link is busy. A content unit is atomic, cannot be further split and each content unit is routed through a single path, but different units may use different paths between two nodes. If a single unit is requested by several users at the same time, then their delivery paths may share common segments.

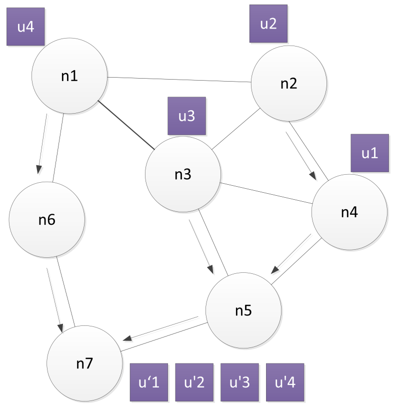

Fig. 1. Sample video delivery system with seven nodes $n_1, ..n_7$ and four units $u_1, ..u_4$. Node n_7 creates a request for a sequence of the units.

Figure 1 shows a sample network delivering four units. Initially, units u_1 - u_4 are located at different nodes. n_7 creates

a sequential request for the units (for more detail on requests, see III. B). The requested units should be forwarded towards the requesting node and played there, one after the other (u'_1 - u'_4).
The problem is computationally hard. If storage capacities are given, the problem is closely related to the Knapsack and the Bin Packing problem that are $\mathcal{NP}$-complete. Even if storage capacities are not considered (i.e., infinite storage is assumed), the problem still remains computationally difficult. The $\mathcal{NP}$-completeness can be shown with the help of the Edge-Disjoint Path problem. It follows from the proof that the content delivery problem remains computationally hard, even if each request demands only one single unit. This is the reason why we concentrate on approximating methods, which can be applied for real-life problems.

B. Components and Notation

This subsection gives an overview on the main components of the problem and the related notation.

1) Units: We name the elementary objects of the content "units". They can be either a short video or a picture, in the case of multimedia delivery. The units can be recorded and stored at any network node. The link bandwidth determines the transfer time through a link.

Unit-related notation:

- U: The set of units.
- s_0: The normed unit-size.
- $s(u)$: The size of unit u (in s_0). In our evaluation, it is an integer value.
- $t_c(u)$: The duration of copying unit u on one link (in time steps), it is an integer in our current model, otherwise it should be rounded down, in order to get a lower bound.
- $t_p(u)$: The duration of playing unit u (in time steps). In our evaluation, it is the same as $t_c(u)$. It is an integer in our model, otherwise it should be rounded up in order to get a lower bound.
- $B(u)$: Set of nodes where unit u is initially stored.

2) Time: The system state is examined in discrete time steps.

Time-related notation:

- T_{MAX}: Index of the latest time step.
- T: The set of time steps. $(1..T_{MAX})$.
- d_0: Time step length, the unit of time resolution (in sec) ($d_0 \cdot t$ gives the time from starting the system to the tth time step). d_0 is recommended to be the copying time of s_0 on a link ($= s_0/b$) because in this case the model is suitable for providing exact results.

3) Requests: The nodes continuously generate requests for one or more units. The set of units in the same request form a "composition". For a sequential composition, the playback of a subsequent unit must not start before the completion of the preceding unit. The users shall receive the first requested unit with the smallest possible delay and the rest of the units within a given deadline d_{max} (40 ms for a video with 25 fps), to ensure continuous playback. If a unit arrived with larger delay, it is considered missed. The delivery of a unit can fail either if the unit is missing from the system as a consequence of removing nodes, or the unit is present in the system but has not arrived at the requesting node before the deadline. For more details on unit compositions, see [14].

Request-related notation:

- R: The set of requests.
- $l(r)$: The number of units queried by request r.
- $I(r)$: $[0, .., l(r)-1]$ - The set of indices of the units within request r.
- $u(r, i)$: The ith unit queried by request r
- $n(r)$: The node where request r is created
- $t(r)$: The time of creating request r
- d_{max}: Maximum allowed delay between subsequent requested units (in sec). In our evaluation, it is 40 ms, i.e., less than the length of one time step: $d_{max} < \frac{d_0}{i}$ where i denotes the maximum number of units in a request).

4) Network: The multimedia content is stored on the network nodes and can be replicated or migrated from one node to another. The links between nodes are characterized by their bandwidth.

Network-related notation:

- N: The set of nodes.
- E: The links between nodes ($\subset NXN$).
- $G(N, E)$: Directed graph describing the topology of the network (here, a connected random graph).
- $N(n)$: The set of adjacent nodes of node n
- b: Bandwidth of link l (in bit/sec). In our actual evaluation, it is the same for all links.

5) Node storage: The node storage is used to store content and to hold message queues containing content to be transferred. S_n is the storage size of node n (in s_0).

IV. Integer Linear Programming Model

We derived the integer linear programming (ILP) formulation of the problem model. The presented model can be regarded as a sample that can be extended or simplified according to further problem models. We introduce the applied variables and formulas as follows:

A. Variables

We introduce several variables to be determined during the optimization process. They describe the possible states of the system and refer to the metrics to be optimized.

- $X_{u,n,t}$: Boolean, indicating whether unit u is located on node n at time step t ($u \in U, n \in N, t \in T$).
- $L_{u,n_1,n_2,t}$: Boolean, indicating whether unit u is copied/moved from node n_1 to node n_2 by time step t ($u \in U, (n_1, n_2) \in E, t \in T, t > t_c(u)$).
- $Y_{r,i,t}$: Boolean, indicating whether unit $u(r, i)$ is successfully served at time step t ($r \in R, i \in I(r), t \in T, t > t(r)$).
- $R_{r,i}$: Boolean, indicating whether serving unit $u(r, i)$ is failed, i.e. the unit is never served successfully ($r \in R, i \in I(r)$).

- $T_{r,i}$: Non-negative integer variable to show the delay in serving unit $u(r,i)$ ($r \in R, i \in I(r)$).

The number of binary and integer variables is proportional to the network size, number of units and number of time steps. In the case of distributed networks, an evaluation scenario typically contains at least 50 network nodes, several hundred units and takes at least 100 seconds. Therefore, the variable number easily achieves 100,000 in practically relevant problem instances. This is a relatively large problem size, but it is tractable due to the large computational capacities of present-day PCs and sophisticated ILP solvers.

B. Optimization Goal

The cost function consists of the number of failed unit deliveries and the total delay. We combine the two optimization goals in a weighted sum, where minimizing the number of failed unit deliveries has priority over the delay. w_R denotes the weight of the failed units in the optimization goal. It should be larger than the maximum possible value of the total delays. In this case, a solution with fewer failed units has always smaller goal value than a solution with more failed units independently from their delays.

$$\sum_{r\in R, i\in I(r)} T_{r,i} + w_R \cdot \sum_{r\in R, i\in I(r)} R_{r,i}$$

C. Constraints

The model to be evaluated is quite complex; it consists of 13 constraints. We give an overview on each of the constraints and indicate simplification possibilities as well.

Node related constraints

Storage capacity: Constraint 1 refers to the storage capacities of the nodes. It includes the storage need of the units whose copying ha started, but have not yet arrived at the node.

$$\sum_{u\in U}\Big(s(u)\cdot X_{u,n,t} + \sum_{n_s\in N(n)}\sum_{t_i=t+1}^{min(t+t_c(u),T_{MAX})} s(u)\cdot L_{u,n_s,n,t_i}\Big) \le S_n, \quad \forall n\in N, \forall t \in T \tag{1}$$

If storage capacities are omitted from the model, several simplifications can be introduced, as seen later.

Initial state: Constraint 2 specifies the initial location of the units. This is usually given as an input. If this constraint is missing, the optimization calculates an optimal initial distribution of the units.

$$X_{u,n,1} = \begin{cases} 1 \text{ if } n \in B(u) \\ 0 \text{ otherwise} \end{cases} \qquad \forall u \in U, \forall n \in N \tag{2}$$

Unit preservation: Constraint 3 ensures that each unit is preserved at least at one node until the end of the examined time period.

$$\sum_{n} X_{u,n,T_{MAX}} \ge 1, \qquad \forall u\in U \tag{3}$$

Unit replacement is unnecessary in the case of unlimited storage because it can be assumed that if a unit is present at a node then it remains there. Therefore, if the storage constraint is omitted, this constraint is omitted as well.

Network and unit transfer related constraints

Unit origin: Constraint 4 gives the precondition of locating a unit at a specific node on a specific time step. According to it, a unit can be stored on a node at a specified time step only if it was copied there at that time or it was already there in the preceding time step.

$$-X_{u,n,t} + \sum_{n_s\in N(n)} L_{u,n_s,n,t} + X_{u,n,t-1} \ge 0, \quad \forall u\in U, \forall n \in N, \forall t\in T, t>1 \tag{4}$$

Copy source: According to Constraint 5a, a unit can be copied on a link if it was present at the source node of the link.

$$-\left(t_c(u)+1\right)\cdot L_{u,n_s,n_t,t} + \sum_{t_i=t-t(c)}^{t} X_{u,n_s,t_i} \ge 0, \quad \forall u\in U, \forall (n_s,n_t)\in E, \forall t\in T, t>t_c(u) \tag{5a}$$

Constraint 5b can be simplified in the case of unlimited storage by exploiting that a unit is never removed from a node.

$$-L_{u,n_s,n_t,t} + X_{u,n_s,t-1} \ge 0, \qquad \forall u \in U, \forall (n_s,n_t)\in E, \forall t \in T, t \ge t_c(u) \tag{5b}$$

Edge capacity: At each time step, only one unit can be copied through an edge Constraint 6. In half-duplex mode, the links between two nodes have to be included in both directions.

$$\sum_{u\in U} L_{u,n_s,n_t,t} \le 1, \quad \forall (n_s,n_t)\in E, \forall (n_t,n_s)\in E, \forall t\in T, t\ge t_c(u) \tag{6}$$

Unicast: Constraint 7 should be added to the model of a specific system if multicasting is not used. Without multicasting, each unit can be forwarded from a node in only one direction at a time.

$$\sum_{n_t\in N(n)} L_{u,n,n_t,t} \le 1, \quad \forall u\in U, \forall n\in N, \forall t\in T, t\ge t_c(u) \tag{7}$$

Request related constraints

Request fulfillment - unit is present: Constraint 8a gives the most important condition of the request fulfillment at time step t, namely that the requested unit should be present at the requesting node at time step t and it should remain there while it is being played.

$$-(t_p(u(r,i))+1)\cdot Y_{r,i,t}+\sum_{t_i=t}^{t_p(u(r,i))} X_{u(r,i),n(r),t_i}\geq 0$$
$$\forall r\in R,\forall i\in I(r),\forall t\in T,t\geq t(r) \quad (8a)$$

Constraint 8b can be simplified in the case of unlimited storage by exploiting the fact that a unit is never removed from a node.

$$-Y_{r,i,t}+X_{u(r,i),n(r),t}\geq 0 \qquad \forall r\in R,\forall i\in I(r),$$
$$\forall t\in T,t\geq t(r) \quad (8b)$$

Request fulfillment - first unit has not been there before: Another important condition 9a for request fulfillment is that if the first unit of a composition is delivered at time step t, it was not present on the requesting node at the preceding time step.

$$-(t-t(r))Y_{r,i,t}-\sum_{t_j=t(r)}^{t-1} X_{u(r,i),n(r),t_j}\geq -(t-t(r))$$
$$\forall r\in R,(i=0\vee\forall i\in I(r)\text{ if }\forall t\in T,t>t(r) \quad (9a)$$

Constraint 9b appears in a simplified form in the case of unlimited storage.

$$-Y_{r,i,t}-X_{u(r,i),n(r),t-1}\geq -1$$
$$\forall r\in R,(i=0\vee\forall i\in I(r)\text{ if }\forall t\in T,t>t(r) \quad (9b)$$

Fulfillment or refuse: Each unit of a composition may be served at exactly one time step or its delivery failed (Constraint 10).

$$\sum_{t\in T} Y_{r,i,t}+R_{r,i}\geq 1 \qquad \forall r\in R,\forall i\in I(r) \quad (10)$$

Sequential composition related constraints

Relationship of subsequent units of a request: Constraint 11 describes the relationship of successful deliveries of subsequent units within a request. $u(i,r)$ denotes the ith member of a requested sequence r. Let k denote the index of the last unit within the request which was delivered before $u(i,r)$. In this case, either the kth and the ith units are subsequent in the request or the deliveries of all requested units between them have failed. A unit is either present at the destination by the end of the playback of the previous one, or it misses the deadline because the allowed inter-unit delay (d_{max}) is so small that a unit arriving one step later misses the deadline, even if the delivery of a series of units fails before it. Therefore, $u(i,r)$ can be delivered at a specified time t only if it is present at the end of the play of unit $u(k,r)$.

$$-Y_{r,i,t}-\sum_{j=k+1}^{i-1} R_{r,j}+Y_{r,k,t-t_p(u(r,k))}+R_{r,k}\geq k-i-1$$
$$\forall r\in R,\forall i,k\in I(r),i>k,$$
$$\forall t\in T,t\geq t(r)+t_p(u(r,k)) \quad (11)$$

Delay of the first unit: Constraint 12 specifies the delay for the first units (start-up delay) of the sequences which is the difference of the arrival and the request time.

$$T_{r,0}-\sum_t (t-t(r))\cdot Y_{r,0,t}-d_{max}\cdot R_{r,0}\geq 0$$
$$\forall r\in R, \quad (12)$$

Delay of the further units: Constraint 13 refers to the inter-unit delay that is either zero if the unit arrives in time or d_{max} if it misses the deadline.

$$T_{r,i}-d_{max}\cdot R_{r,i}\geq 0$$
$$\forall r\in R,\forall i\in I(r),i>0,\forall t\geq t(r) \quad (13)$$

Bounds:

$X_{u,n,t}\in\{0,1\}$, $L_{u,n_s,n_t,t}\in\{0,1\}$, $Y_{r,i,t}\in\{0,1\}$, $R_{r,i}\in\{0,1\}$, $T_{r,i}\geq 0$

V. Solution method

The recommended optimization process contains the steps as follows:

- Preprocessing
- Linear programming model generation
- Solving the linear programming model (external tool)
- Generating statistics
- Repeating the whole process with updated data if necessary

All steps except solving the linear programming model are implemented in C++.

A. Preprocessing

1) Calculating Shortest Paths for Unit Deliveries: The delay of a content unit delivery is at least the length of the shortest path between the requesting node and the closest location of the requested unit. The delay can be larger if some edges along the path are overloaded. This step determines the shortest delivery paths from the requested units to the requesting nodes and the shortest time needed to fulfill requests without missing any units, if possible. The LP model generation applies this method to reduce the size of the content delivery problem model by excluding cases from the solution space where a unit would occupy a node earlier than it is possible.

2) Filtering Requests and Units: If the optimization is restricted to a certain time interval, only those requests have to be taken into account that are created before the end of the time interval and whose delivery is not completed before the beginning of the interval. Some units may never be requested. These units do not have to be distinguished from each other – they may occupy storage and are preserved in the system, but one specific unit type is enough to denote these units.

B. Generating Linear Programming Model

1) Generating Content Placement Model: It creates the problem model from the input in the common LP format recognized by most Linear Programming solvers. The model is applicable to fulfill the optimization criteria given in the preceding section. The model generation can be called with several parameters in order to create models of different types, e.g., limited/unlimited storage.

2) Generating Unit Replication Model: The unit replication model determines the location of units by minimizing the expected value of the average delay. This model is static and can usually be solved faster than the content placement model. The replication model can stand alone or be embedded into the content placement model. Its input contains the frequency of unit requests for each node. The implemented model determines the locations of the units in such a way that the total sum of the products of the frequency of the requests and the distances from the requesting nodes to the closest locations of the requested units are minimal.

C. Solving the Linear Programming Model

The generated models can be read and evaluated by ILP solvers. This is the most time-consuming step during the whole process. We used IBM's CPLEX, which is free for research purposes [15] and performs very well in benchmark tests [16]. The solvers continuously generate upper and lower bounds for the optimal solution. If both values are equal, the optimum is achieved. The worst-case running time of the solver is exponential and may fail to find the solution even after several days. The optimization can be stopped at any time, and the current bound can be used, even if the exact optimum is unknown. At the end, the optimal solution can be written into a file that can be used for statistics generation and further optimization.

D. Generating Statistics

This step calculates the number of delivered units, their total and average delays and the number of failed unit deliveries for each time step and their cumulative values for a time period.

E. Iterating the above Steps and Updating

The Linear Programming model can be solved in a single step or in several steps processing subsequent time periods. In the multi-step approach, the problem is solved on-line, i.e., without knowledge of future requests. (We use the term on-line in the sense of algorithm theory, see [17]). It is enough to consider requests created at or before the time of the examined time period because the evaluated delivery systems are also aware of a request only after its creation. The optimization is executed for each time period with updated input. The models referring to subsequent time periods are built on each other: The solution of time period t forms the part of the input and specifies the initial location of the units at time period $t+1$. By using the multi-step approach, cases shown to be computationally intractable in a single step, can be solved because only a limited problem space has to be handled.
The ILP model without replication is useful to evaluate methods that are also working without active replication. However, a bound in the multi-step approach does not necessarily mean a bound for the entire problem because the initial location of the units coming from the result of the preceding time period may not be optimal for the next time period.
The placement and the replication models can be applied alternatively: The replication model can be used to determine the optimal initial location for a time period and then optimal routing is calculated by using the placement model.

VI. Evaluated Scenario

A. Variants of the optimization model

We used three main methods to find lower bounds for the self-organizing approach as follows:

Complete ILP model with limited storage The optimum gained by the complete ILP model provides the most accurate lower bound for the average delay in the system. It includes proactive replication as well. However, it is time-consuming to calculate and can easily lead to run out of memory or fail to find any solution even after several days.

Complete ILP model with unlimited storage The running time can be significantly reduced if the limited storage constraint is omitted. The model with limited storage provides the same optimum as the case of the unlimited storage if the maximal storage need at a node is less than the available storage capacity. Furthermore, the optimum of the case of the unlimited storage can be used as a lower bound for the limited case.

Shortest-path lengths Calculating the shortest path lengths between the requesting nodes and the closest location of the requested units serves as the most natural and the fastest way to get lower bounds for the minimal delays. The shortest path algorithm takes the duration of copying a unit along a link as the length of the link. The algorithm allows simultaneously overlapping routes. The shortest path length between two nodes is always smaller or equal than the delay along the route because the algorithm considers only a subset of constraints of the content placement problem and it ignores some constraints including the Edge Capacity 6, the Storage Capacities 1 and the Unicast Constraints 7. If the generated shortest paths have a large overlap, the result may be much smaller than the real optimum of the content placement. However, if the load on the links are low, the method may find even the exact optimum.

B. Scenario parameters

We set up a scenario in order to validate the hormone-based algorithm with the presented approach. The scenario evaluates how well the approach handles a high number of concurrent requests from all nodes. The network topology is represented by a connected Erdős-Rényi random graph consisting of 50 nodes. Each unit has the same size – 125KB. In the beginning, 1,849 units are created. Each node provides 900MB storage for replications and unit transfers. At the start of the examined period, each client requests units at the same time. Then clients generate sequential requests continuously: if one request is

TABLE I
COMPARING DIFFERENT METHODS ON ONE TIME STEP

Method	running time in s	total delay in s	missed units
ILP with unlimited storage	463.21	141	2
ILP with limited storage and without replication	3244.94	141	2
ILP with limited storage and replication	-	-	-
Shortest path lengths	0.016	131	2

fulfilled the next one is sent by the client. The length of a request varies randomly from 1 to 9 units. The bandwidth between two nodes is set to 1 Mbit/s. We apply a random churn model, i.e., at every time step a node is removed or another added with a 10 % probability. If a node is added, it is empty, but it starts immediately requesting units. The transport of one unit from one node to another takes one second, which is selected as the length of a time step. The length of the examined period is set to 100 s, which is enough for the hormone-based algorithm to stabilize the delay. We selected this set up because this scenario is large enough for evaluating the self-organizing algorithm. The discrete time step does not cause inaccuracy in the optimization model in case of the above selection of the parameter values.

The tests were run on a 2 x Intel Quad-Core 2.33 GHz (8 core) computer with 10 GB memory. IBM ILOG CPLEX V12.3.0.0 was used as ILP solver under Ubuntu Linux 12.

C. Comparing the variants of the optimization model

In this section, we compare the running times and the results of different variants of the optimization approach. The examination is restricted to the requests created at the initial time steps and within the first 10 sec of the simulation time, because this is period with the highest activity of the nodes.

The running times in Table I show that omitting the storage size accelerates the speed seven times, in the given scenario. We received the same accuracy (total delay in s) at both methods even if the storage capacity was decreased near to the initial size occupied by units on the nodes because the version with limited storage was able to get enough storage for routing by dropping the instances of the currently not requested units. Running the model with limited storage and replication was aborted because it was unable to find the optimum even after one day. The shortest path approach is by far the fastest, but has the lowest accuracy (the ILP provides a tighter bound).

D. Performance

We examined how the running time of the optimization method depends on different parameters. A series of problems with different sizes are generated. Their inputs are based on the one applied in the previous subsections but the networks contain only some of the nodes of the original network topology. The number of nodes varies from 10 to 50 and the number of units is the same at each cases. Table II shows that the running times rise quickly as the number of nodes increases. Running the model with limited storage and

TABLE II
RUNNING TIMES IN SECONDS AS A FUNCTION OF THE NUMBER OF NODES

Number of nodes	10	20	30	40	50
ILP with unlimited storage	1.29	12.17	159.77	321.04	463.21
ILP with limited storage and without replication	1.28	15.3	451.03	1607.54	3244.94
ILP with limited storage and replication	9.17	3168.12	41675.2	-	-

TABLE III
RUNNING TIMES IN SECONDS AT DIFFERENT TIME STEPS

Time step	0	4	5	8	9	10
Running time	463.21	136.84	75.66	24.98	21.84	16.69
Number of units	308	287	268	164	142	110
Total delay	141	52	22	7	7	3

replication is able to solve problems containing less than 40 nodes within a reasonable time.

Table III shows the running times at different iterations of the algorithm within the first 10 time steps. A time step was skipped if no new request arrived. The number of units includes only the units routed at the specific time step. The running times may significantly vary although a similar number of units are handled, the reason could be that the running time depends on the calculated delay as well, i.e., it takes more time to find longer routes in ILP.

E. Evaluating the hormone-based algorithm

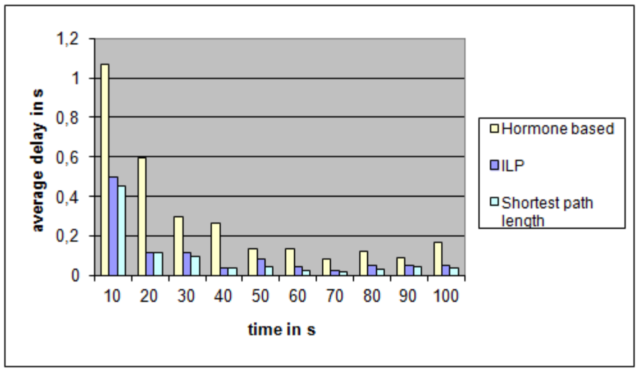

Fig. 2. Delay comparison between the hormone-based, the ILP optimization and the shortest paths method, including churn. The average delays are calculated for units delivered within 10 second intervals.

In the evaluation, the parameters specified in Subsection VI-B are used for all methods. Although some input data (including the network, the initial location of units and the request history) is generated randomly, the comparison of the methods is performed in a deterministic way because all methods use the same input. The maximum storage need of the model with unlimited storage is smaller than the available storage capacity, thus it is not necessary to compare the results with the limited storage case of the ILP. We applied the multi-step variant of the ILP approach for the first 100 seconds (which is the time the hormone-based algorithm takes to stabilize) and iterated the ILP optimization for each time step.

The ILP solver took almost 40 minutes to calculate the evaluation data for the entire period. In comparison, the hormone-based algorithm took less than a minute. In Figure 2 it can be seen that in the beginning the hormone-based algorithm is far from the optimum, because the replication of units takes time. Later, the difference between the average delays and the optimum reduces as the units are distributed more widely and the load of the links decreases. If there is no loss, the delay converges to 0 seconds. The churn causes some delay increase, because nodes are deleted and units can get lost and alternative paths have to be used. As shown Figure 2 the delay is never as large as at the start of the simulation. A proactive replication step on entering new units might improve the start-up delay and reduce the difference to the optimal algorithm. The shortest path method performs best, however, as stated before, due to omitting important conditions of network and node states, the ILP provides a more accurate theoretical bound for the evaluation. The unique and significant gain of the ILP evaluation is showing how the given self-organizing approach achieves the global optimum.

VII. Conclusion

We applied ILP-based optimization techniques to determine the optimal content delivery for evaluating unstructured dynamic networks. We introduced the main components of the ILP model and described the solution method. The performance of the approach was improved by heuristics in the preparation step and by further iterations. The model looks for the optimum in an on-line manner which makes the bound more accurate. This approach is able to determine the minimum of the average delay and the number of failed unit deliveries for reasonably sized networks (e.g., 50 nodes). The derived optimum is independent from the validated algorithm. Our main goal was to present a detailed, general method, which can be adapted to a number of different delivery problems. Furthermore, the method can be used to diagnose whether inefficient performance of a system comes from the applied delivery method or from the underlying system itself. In further work, we intend to improve the speed of the case of the limited storage with replication by applying the content placement model and the static unit replication models alternatively.

VIII. Acknowledgment

This work was supported by Lakeside Labs GmbH, Klagenfurt, Austria, Alpen-Adria Universität Klagenfurt and funding from the European Regional Development Fund and the Carinthian Economic Promotion Fund (KWF) under grant KWF 20214—17097—24774 and grant KWF 20214—18128—26673 and the Hungarian National Development Agency under grant HUMAN_MB08-1-2011-0010. We further thank Lizzie Dawes for proof-reading the paper.

References

[1] 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 (SAPMIA 2010)*. Florence, Italy: ACM, 2010, pp. 3–8.

[2] ——, “Replication for Bio-inspired Delivery in Unstructured Peer-to-Peer Networks,” in *Ninth Workshop on Intellingent Solutions for Embedded Systems (WISES 2011)*. Regensburg, Germany: IEEE, 2011, p. 6.

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

[4] G. D. Caro, “AntNet: Distributed Stigmergetic Control for Communications Networks,” *Journal of Artificial Intelligence Research*, vol. 9, pp. 317–365, 1998.

[5] 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, USA: ACM, 2002, pp. 84–95.

[6] M. Hossain and D. Elghobary, “Ant based routing algorithms for resource constrained networks,” in *Instrumentation and Measurement Technology Conference I2MTC 2010 IEEE (2010)*, 2010, pp. 192 – 197.

[7] J. Wang, L. Li, S. H. Low, and J. C. Doyle, “Can shortest-path routing and tcp maximize utility,” *Office*, 2003.

[8] A. Banerjee, W.-c. Feng, D. Ghosal, and B. Mukherjee, “Algorithms for Integrated Routing and Scheduling for Aggregating Data from Distributed Resources on a Lambda Grid,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 1, pp. 1–11, 2008.

[9] Y. Li, S. Ranka, and S. Sahni, “In-advance path reservation for file transfers in e-science applications,” *J. Supercomput.*, vol. 59, no. 3, pp. 1167–1187, mar 2012.

[10] K. Krishnan, D. Shallcross, and L. Kant, “Joint optimization of transmission schedule and routing for network capacity,” *IEEE Military Communications Conference, 2008. MILCOM 2008*, 2008.

[11] J. Moy, “OSPF Version 2. IETF RFC 2328. [Online]. Available: http://www.faqs.org/rfcs/rfc2328.html,” 1998, Apr.

[12] S. Kwon and N. Shroff, “Analysis of shortest path routing for large multi-hop wireless networks,” *Networking, IEEE/ACM Transactions on*, vol. 17, no. 3, pp. 857 –869, june 2009.

[13] A. Sobe, W. Elmenreich, and L. Böszörmenyi, “Storage balancing in self-organizing multimedia delivery systems,” *arxiv e-print 1111.0242; TR/ ITEC/ 01/ 2.13*, p. 16, 2011.

[14] A. Sobe, L. Böszörmenyi, and M. Taschwer, “Video Notation (ViNo): A Formalism for Describing and Evaluating Non-sequential Multimedia Access,” *IARIA International Journal on Advances in Software*, vol. 3, no. 12, pp. 19–30, 2010.

[15] “CPLEX, http://www.ilog.com/products/cplex/product/suite.cfm,” 2012.

[16] T. Koch, T. Achterberg, E. Andersen, O. Bastert, T. Berthold, R. E. Bixby, E. Danna, G. Gamrath, A. M. Gleixner, S. Heinz, A. Lodi, H. Mittelmann, T. Ralphs, D. Salvagnin, D. E. Steffy, and K. Wolter, “MIPLIB 2010, Mixed Integer Programming Library version 5,” *ZIB-Report 10-31*, p. 47, 2010.

[17] A. Borodin and R. El-Yaniv, “Online computation and competitive analysis,” 1998.