[DMHLSt06] *Vasilios Darlagiannis, Andreas Mauthe, Oliver Heckmann, Nicolas Liebau, and Ralf Steinmetz;* **On Routing in a Two-Tier Overlay Network based on de Bruijn Digraphs;** The 10th IEEE/IFIP Network Operations and Management Symposium NOMS 2006, Vancouver, April 2006, auf CD.

On Routing in a Two-Tier Overlay Network based on de Bruijn Digraphs

Vasilios Darlagiannis*, Andreas Mauthe†, Oliver Heckmann*, Nicolas Liebau*, Ralf Steinmetz*
*Multimedia Communications (KOM), Technische Universität Darmstadt
Merckstr. 25, D-64283 Darmstadt, Germany
Email: {Darlagiannis, Heckmann, Liebau, Steinmetz}@KOM.tu-darmstadt.de
†Computing Department, Lancaster University
Lancaster, LA1 4YR, UK
Email: andreas@comp.lancs.ac.uk

***Abstract*— The intrinsic properties of the employed graphs in designing Peer-to-Peer overlay networks are crucial for the performance of the deployed Peer-to-Peer systems. Several structured topologies have been proposed based on meshes, enhanced rings, redundant tree structures, etc. Among them, de Bruijn graphs are promising alternatives since they provide some crucial asymptotically optimal characteristics.**

In this paper, we discuss the necessary algorithms and protocol messages to develop efficiently the employed routing procedure of Omicron, which is a hybrid overlay network based on de Bruijn graphs enriched with clustering and role specialization mechanisms. Enhancements of the original de Bruijn structure are advised to cope with the intrinsic issue of uneven distribution of the routing workload. The developed system is evaluated and compared with Chord, which is used as the reference point. The superiority of de Bruijn based overlay networks with respect to scalability is quantitatively demonstrated using simulation experiments. Further, the ability of the two systems to exploit the underlying network is investigated.

I. Introduction

In the modern era of Peer-to-Peer (P2P) systems, considerable research efforts have been devoted in the construction of efficient overlay networks. The most essential characterizations of a P2P overlay network are (i) whether its topology is *tightly structured* (e.g. Chord [29]) or *loosely structured* (e.g. Freenet [3]) and (ii) whether it has a *flat* (e.g. Gnutella v0.4 [24]) or *hierarchical* structure (e.g. eDonkey [14]). In addition, *hybrid* approaches have been investigated (e.g. Omicron [8], SHARK [23]) that exploit the advantages of combining mechanisms of different nature. The design space of P2P overlay networks is discussed in more detail in [22].

Structured approaches or hybrid solutions using structured components can provide upper bounds on the complexity of the employed routing mechanism in worst-case scenarios. Structured overlay networks are mostly illustrated based on *Distributed Hash Tables* (DHTs) [4]. Examples of such systems include Chord, Pastry [27], Tapestry [30], CAN [25], Kademlia [21], etc. Each of the aforementioned systems utilizes a different routing procedure to forward queries towards their logical destination in the overlay network. The employed *routing algorithm* is customized to fit with the constructed graph that models the topology of the overlay network.

Therefore, the selection of an optimal graph structure is a critical step for the performance of the deployed P2P system. Multiple requirements raise in the design phase of the overlay networks, such as scalability, expandability, resilience, simplicity, etc. However, the fulfillment of these requirements is not a trivial task since conflicts often arise [8]. de Bruijn graphs [9] are an appealing type of graphs investigated in the scope of interconnection networks [17], [18]. Despite the intrinsic shortcomings found in de Bruijn graphs when applied in dynamic environments like P2P systems [5], they still represent promising candidates for such networks. The most important characteristic is their *asymptotically optimal density* where the graph diameter increases logarithmically with respect to the network size [19]. This optimality is achieved even when the node degree is fixed. In fact, several P2P overlay networks have been designed using de Bruijn graphs, e.g. Koorde [16], D2B [12], ODRI [19] and Omicron [8].

The goal of this paper is to investigate in depth the details of a routing mechanism designed to apply to P2P networks based on de Bruijn graphs. Particularly, we investigate the required routing algorithms and protocol messages utilized in the development of Omicron [8], a hybrid, two-tier, de Bruijn digraph (directed graph) based overlay network. Omicron is selected among the other de Bruijn based approaches because it offers additional desirable characteristics, such as load balance mechanisms and support for heterogeneous peers. Omicron divides the core vertical overlay operations into distinguishable roles and assigns the roles to peers based on their physical capabilities and user behavior. Peers form clusters that provide the desired stability to the fixed degree network. The proposed mechanisms are quantitatively evaluated using the discrete event, packet based P2P overlay network simulator described in [7]. It should be mentioned that the de Bruijn digraph related algorithms and protocols are general and can be applied to any de Bruijn digraph based P2P system.

The contribution of the paper is summarized here. (i) The required routing protocol details are investigated and the structure of the involved messages is provided. (ii) The routing algorithms for de Bruijn graphs found in the literature as well as their developed enhancements aiming to increase the

routing workload balance are quantitatively evaluated with simulations and compared with their analytical approximations. (iii) The routing performance in terms of scalability and exploitation of the underlying is provided both for the two-tier de Bruijn based network and Chord. (iv) The routing workload balance for de Bruijn based overlay networks is quantitatively evaluated. (v) The trade-off between efficient underlying network exploitation and even routing workload distribution is investigated for the two-tier architecture.

This paper is organized as follows. Section II reviews the basic characteristics of de Bruijn digraphs and the related Moore bounds. Afterwards, Section III provides an overview of Omicron. The fundamental concepts of the related protocols and algorithms are presented in Section IV. Then, the focus of Section V aims to reveal the details of the routing mechanism where the related algorithms and protocol messages are presented. The investigated mechanisms are quantitatively evaluated in Section VI that summarizes the related simulation results we gathered. Finally, Section VII concludes the contribution of this paper.

II. de Bruijn Digraphs

Directed graphs (digraphs) have been extensively used in interconnection networks for parallel and distributed systems design (cf. [15], [18], etc.). Digraphs received special attention from the research community aiming to solve the problem of the so-called (k, D) digraph problem. The goal is to maximize the number of vertices N in a digraph of maximum out-degree k and diameter D [11]. Some general bounds relating the order, the degree and the diameter of a graph are provided by the well-known Moore bound [2]. Assume a graph with node degree k and diameter D; then the maximum number of nodes (*graph order*) that may populate this graph is bounded by Equation 1:

$$N \leq 1 + k + k^2 + \ldots + k^D = \frac{k^{D+1} - 1}{k - 1}. \quad (1)$$

Interestingly, the Moore bound is not achievable for any non-trivial graph [2]. Nevertheless, in the context of P2P networks, it is more useful to reformulate Equation 1 in a way that provides a lower bound for the graph *diameter* (D_M), given the node degree and the graph order [10]:

$$D_M = \lceil log_k(N(k-1)+1) \rceil - 1 \leq D. \quad (2)$$

The *average distance* (μ_D) among the nodes of a graph may also be bounded by the following inequality provided in [28] (which is approximated in [19]):

$$D_M - \frac{k(k^{D_M} - 1)}{N(k-1)^2} + \frac{D_M}{N(k-1)} \approx$$

$$\approx D_M - \frac{1}{k-1} \leq \mu_D. \quad (3)$$

An interesting class of digraphs is the so-called lexicographic digraph class, which includes the de Bruijn and Kautz digraphs[1] [1]. de Bruijn graphs belong to a very important class of graphs; they have *asymptotically optimal graph diameter and average node distance* [19]. Thereby, they are employed in the design of our work. Figure 1 shows a directed de Bruijn$(2, 3)$ graph denoting a graph with a maximum out-degree of 2, a diameter of 3 and order 8. For graphs with fixed out-degree of 2 the maximum number of nodes[2] is always limited by 2^D. The graph contains 2^{D+1} directed edges in this case. Each node is represented by D-length (3 in this example) strings. Every character of the string can take k different values (2 in this example) from an alphabet (lexicographic digraphs). In the general case each node is represented by a string such as $u_1u_2...u_D$. The connections between the nodes follow a simple left shift operation from node $u_1(u_2...u_D)$ to node $(u_2...u_D)u_x$, where u_x can take one of the possible values of the characters $(0, k-1)$.

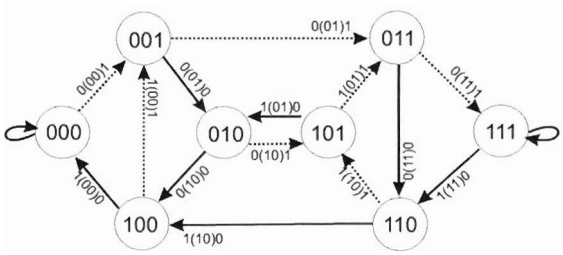


Fig. 1. Directed de Bruijn$(2, 3)$ graph.

de Bruijn graphs have been suggested to model the topology of several P2P systems, though as it will become clear in the following section, we exploited them in an innovative way. Other considerable examples of P2P overlay networks that are based on de Bruijn graphs are Koorde [16], D2B [12] and Optimal Diameter Routing Infrastructure (ODRI) [19].

III. Omicron Overview and Architecture

A. Overview

Omicron *(Organized Maintenance, Indexing, Caching and Routing for Overlay Networks)* is a P2P overlay network aiming to address issues of heterogeneous, large-scale and dynamic P2P environments. Its hybrid, DHT-based approach makes it highly adaptable to a large range of applications. Omicron deals with a number of conflicting requirements, such as scalability, efficiency, robustness, heterogeneity and load balance. Issues to consider in this context are:

Topology. The rational in Omicron's approach is to reduce the high maintenance cost by having a small and fixed node degree, thus, requiring small and fixed size routing tables (at least for the majority of peers), while still performing lookup operations at low costs. For this reason the usage of

[1]de Bruijn graphs are less dense than Kautz graphs but they are more flexible since they do not have any limitations on the sequence of the represented symbols in every node.

[2]The Moore bound determines always maximum upper bounds on the size of the graphs that are not reachable for non-trivial cases.

appropriate graph structures (such as de Bruijn graphs [9]) is suggested. However, while the small fixed node degree reduces the operational cost, it causes robustness problems.

Clustering mechanism. To address the robustness issue, clusters of peers are formed with certain requirements on their endurance. *Cluster endurance* $E_C(t)$ of cluster C is defined as the probability that at least one peer of the cluster will survive until time t. Endurable clusters increase the stability of the network despite the (un)reliability of the peers as they unexpectedly depart from the system. Clusters can be considered as an equivalent mechanism to the suspensions used in vehicles to absorb shocks from the terrain. In order to maintain cluster endurance, new peer join requests are directed to the least endurable clusters. When the endurance of a cluster gets below a certain threshold, a merging operation takes place between the critical cluster and neighbor cluster(s).

Roles. A unique feature of Omicron is the integrated specialization mechanism that assigns particular roles to peers based on their physical capabilities and user behavior. The specialization mechanism provides the means to deal with peer heterogeneity. This scheme fits the contribution of each node to its resource capabilities and aims at the maximization of the cluster efficiency by providing appropriate incentives to peers to take a certain role. As it can be observed from Omicron's name, four different core roles have been identified: *Maintainers (M), Indexers (I), Cachers (C) and Routers (R).* Maintainers are responsible to maintain the overlay network topology, while Indexers handle the relevant indexing structures. Routers forward the queries towards their logical destination and Cachers reduce the overall routing workload by providing replies to popular queries. Roles are additively assigned, meaning that peers do not remove their older roles as they get new ones.

Identification scheme. A dual identification scheme has been introduced for Omicron with a number of advantages. Clusters are assigned a *Globally Unique IDentifier* (GUID) that is used to route requests over the network. Advertised items are assigned a GUID and are located at the clusters whose GUID matches best (prefix-based). Moreover, peers are assigned their own GUID to trace their actions in the system.

Expandability. In addition, an incrementally expandable algorithm has been designed to adapt the exponentially growing de Bruijn graphs to the incrementally expandable P2P systems. Detailed descriptions of the Omicron mechanisms and algorithms are provided in [8] and [5].

B. Two-tier Network Architecture

The suggested *two-tier* network architecture is a major step towards accomplishing the fulfillment of the targeted requirements. It enables the effective usage of de Bruijn graphs by successfully addressing their shortcomings. In fact, the successful "marriage" of two different topology design techniques (in a combination of a *tightly structured macro level* and a *loosely structured micro level*) provides a hybrid architecture with several advantages.

1) **Tightly structured macro level.** Adopting the topological characteristics of de Bruijn graphs, the macro level is *highly symmetrical* enabling *simple routing* mechanisms. Composed of endurable components, it results in a relatively *stable* topology with *small diameter* and *fixed node degree.*
2) **Loosely structured micro level.** On the other hand, the micro level provides the desirable characteristics to the macro level by following a more loosely structured topology with a great degree of *freedom* in the neighbor selection. This freedom may be invested on regulating and achieving a *finer load balance*, offering an effective mechanism to handle potential *hot spots* in the network traffic. Moreover, *locality-aware* neighbor selection may be used to maximize the matching of the virtual overlay network to the underlying physical network. Finally, *redundancy* may be developed in this micro level supplying seamlessly *fault-tolerance* to the macro level.

An example of the hybrid topology is illustrated in Figure 2. The structured macro level is a de Bruijn$(2, 3)$ digraph. Two nodes (representing peer clusters) are "magnified" to expose the micro level connectivity pattern between them. Two different connection types are shown: inter-cluster connections and intra-cluster connections.

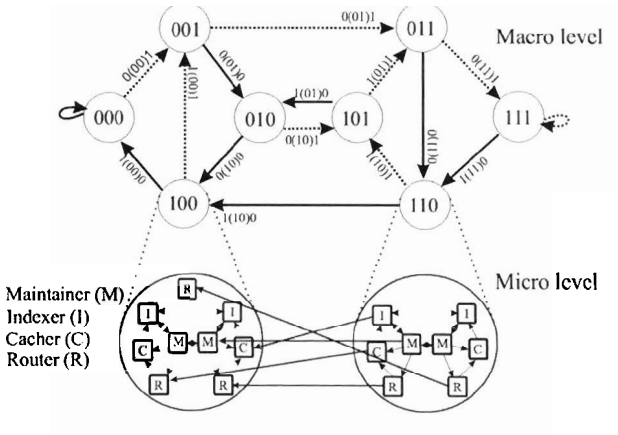


Fig. 2. Omicron Overlay Network

IV. Protocol Basics

Several of the developed protocols for the particular aforementioned roles follow the hop-by-hop paradigm to deliver the messages to their destination. The developed communication mechanism uses a local message dispatcher on every peer. The message dispatcher examines several message fields in order to direct the incoming message to the appropriate instance that can handle it. Further details on the developed message dispatcher can be found in [7].

Inter-peer communication is implemented via a number of messages that constitute the necessary communication protocols. A number of common fields appear in each defined message. Table I provides the structure of an abstract message

that defines the common fields. Every concrete message is derived from this abstract structure.

TABLE I
COMMON MESSAGE STRUCTURE.

Field	Description
Name	Name of the message.
Scope	Defines the related overlay network.
Type	Relates the message to a specific role.
Style	Defines the way the message is forwarded towards its destination.
Initiator	Encapsulates the GUID of the peer that created the message.
Sender	Encapsulates the GUID of the peer that forwarded the message.
Destination	Encapsulates the GUID of the peer that should receive the message.
TTL	Time-to-live (optional).
Hops	Number of visited peers (optional).
Visited	List of traversed peers (optional).

The field *name* uniquely identifies each individual message (e.g. `UpdateClusterMap`, `LookupRequest`, etc.). The *scope* of the message defines the related overlay network. For example, an inter-cluster message has a different scope compared to an intra-cluster message. The *type* of the message specifies the role that should handle the incoming message, e.g. a Router, a Maintainer, etc. The *style* of the message defines the forwarding communication style, i.e., recursive or iterative.

The field *initiator* encapsulates the GUID of the message originator. Supplementary information is provided in the field *sender* that encapsulates the GUID of the most recent peer that forwarded the specific message. The combination of these two fields provides the essential information to enable efficient communication schemes for query-reply based protocols. For example, a reply can be directly provided to the originator of any query without requiring to reversely traverse the same path that the query passed through. However, it should be noted that in particular cases the semantics of routing can change, e.g. involving the caching mechanism described in [6]. Thus, intermediate nodes may be visited in order to deliver the reply, too. The field *destination* is set to the next closest peer in every hop based on the local routing table. Then, the message is forwarded to this peer using the underlying network.

Finally, some optional fields can provide additional useful functionality. For example, setting the field *Time-to-Live* (TTL) to a maximum value can decrease the generated traffic (particularly useful when broadcasting mechanisms are utilized). However, it should be noted that the scope of the message is limited when the TTL is actively used, which is inappropriate for e.g. `LookupRequest` messages. The following two fields (*hops* and *visited*) provide the means to construct non-oblivious messages. Thereby, it is possible to avoid cycles in the routing procedure, which is proven to be especially useful for random walk based communication mechanisms (that require single visit of peers).

V. ROUTING

Efficient routing is the most crucial functionality expected to be provided by a network system, which is the main goal of DHT-based structured overlay networks. The importance of efficient routing is investigated in [26] and [13], where the authors raise questions on the optimal routing that can be achieved. In terms of de Bruijn graphs, which appear to be one of the most promising network topologies for P2P systems, the complexity of the routing procedure increases logarithmically with the size of the system. This is formally denoted as $O(log(N))$, where N is the number of nodes. The bound holds even when the node degree is fixed. For cases where the node degree also increases logarithmically with respect to the network size the resulting routing complexity is provided in [16]:

$$D = O(\frac{log(N)}{log(log(N))}), \quad (4)$$

where D is the diameter of the de Bruijn digraph.

The optimal routing procedure in de Bruijn graphs has been investigated by other researchers as well. In particular, de Bruijn graphs have been studied in [20], aiming to combine short routing paths with fault-tolerant routing. Also, routing schemes for de Bruijn networks mostly applicable for parallel systems have been considered in [15], aiming to evaluate oblivious and non-oblivious mechanisms. Undirected de Bruijn graphs have been investigated in [18], focusing on the computational complexity of the routing algorithms. A pattern matching based approach for optimal routing is provided in [17]. Finally, Sivarajan and Ramaswani provide a shortest path solution for de Bruijn digraphs [28]. The simplicity of this solution makes it an interesting approach. In fact, this solution has been adopted and extended to fit the needs of our work.

A. Enhanced de Bruijn Topology

As it can be noticed in Figure 1, nodes (000) and (111) are self-connected. In fact, each node of the form $(u_1u_2...u_D)$ where it holds that $(u_1 = u_2 = ... = u_D)$, is a self-connected node. Such nodes are less loaded with routing traffic since their incoming and outgoing degree is practically $D-1$. This can be also empirically observed with simulation experiments, as it is shown in Section VI. In order to reduce the lower node degree effect, an enhanced de Bruijn structured has been advised. The enhanced structure connects self-connected nodes to each other. For example, the de Bruijn graph shown in Figure 1 is modified as shown in Figure 3. The dashed connections between nodes (000) and (111) can improve the routing load balance of the network. However, the routing algorithm has to be changed accordingly.

B. Routing Algorithms

1) Basic Algorithms: In order to construct a shortest-path algorithm for de Bruijn digraphs an operation

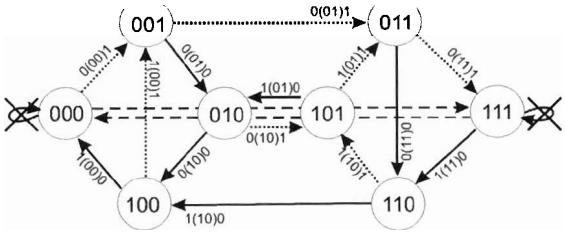


Fig. 3. Directed de Bruijn(2, 3) graph without self-connected nodes.

`shift_match(shift, K, L)` (where $0 \leq shift \leq D$) is defined in [28]. In terms of Omicron, $K = (k_1 k_2 ... k_D)$ is the GUID of the cluster that includes the particular Router peer forwarding a message. $L = (l_1 l_2 ... l_D)$ is the GUID of the target key. The notations $L[k]$ and L_k are used interchangeably to represent the kth symbol of the L GUID. The operation on the two keys returns `true` if and only if:

$$(k_{1+shift} k_{2+shift} ... k_D) = (l_1 l_2 ... k_{D-shift}), \quad (5)$$

and `false` otherwise. The operation is described in Algorithm V.1.

Algorithm V.1: SHIFT_MATCH($shift, K, L$)

for $i \leftarrow 1$ **to** $sizeof(K) - shift$
do { **if** $(K[i + shift] \neq L[i])$ **then** **return** (**false**)
return (**true**)

In addition, it is necessary to define an additional operation `merge(shift, K, L)` (where $0 \leq shift \leq D$). The operation returns the sequence of length $D + shift$ given by $k_1 k_2 ... k_D l_{D-shift+1} ... l_D$. In order to complete the routing of a particular message, L steps are required, where

$$L = D - shift. \quad (6)$$

Then, the shortest path is calculated by Algorithm V.2. However, this algorithm does not consider the enhanced de Bruijn digraph as shown in Figure 3.

Algorithm V.2: SHORTEST_PATH(K, L)

$shift = 0$
while $(shift\_match(shift, K, L) ==$ **false**
and $shift < D)$
do $shift = shift + 1$
return $(merge(shift, K, L))$

2) Enhanced Algorithms: In order to develop an algorithm for the enhanced digraph, it is necessary to define the following operations on the de Bruijn based GUIDs: (i) `similarSymbol(K)` and (ii) `invSymbol(K)`, where K is a de Bruijn based GUID. The first operation returns the k_D symbol of K if and only if $k_i = k_D, \forall\ i \in [(D/2) - 1, D]$, otherwise a symbol that does not belong to the alphabet of the de Bruijn digraph. Similarly, the operation `invSymbol(K)` returns the k_0 symbol of K if and only if $k_i = k_0, \forall\ i \in [0, (D/2) + 1]$, otherwise a symbol that does not belong to the alphabet of the de Bruijn digraph is returned ($\oslash$). Algorithm V.3 provides the `enhanced_next_hop` algorithm for enhanced de Bruijn digraphs. The variable `routingTable[s]` represents the local routing table structure, which provides the address of the next hop that matches the provided symbol s. The entries of `routingTable` are populated based on the received `ClusterMaps` of the neighbor clusters. ClusterMap is a structure that provides information about the members of each cluster.

Algorithm V.3: ENHANCED_NEXT_HOP(K, L)

$shift = 0$
while $(shift\_match(shift, K, L) ==$ **false**
and $shift < D)$
do $shift = shift + 1$
if $(shift == 0)$
then
return $(NULL)$
if $(K.similarSymbol() \neq \oslash$
and $L.invSymbol() \neq \oslash$
and $K.similarSymbol() \neq L.invSymbol()$
and $shift > D/2)$
then
return $(routingTable[K.similarSymbol()])$
else
$s = L[D - shift]$
return $(routingTable[s])$

Algorithm V.3 is called on each Router to determine the peer of the next cluster that is closer to the final destination. It considers the enhanced de Bruijn topology.

C. Messages

This section provides a description of the messages that constitute the routing protocol. Table II describes the structure of the `LookupRequest` message. It includes a *key* representing the GUID of the queried item. `LookupRequest` messages are forwarded towards the cluster whose GUID matches best with the included key.

TABLE II
LOOKUPREQUEST MESSAGE STRUCTURE.

Field	Description
Common	Derived common message structure (cf. Table I).
Key	Includes the GUID describing the queried item.

Table III describes the structure of the LookupReply message. It includes a *key* representing GUID of the queried item and the related indexing information (*IndexValue*). It is common technique to set the indexing information to merely the address of the owner of the related service or resource. However, the indexing information may include a long list of addresses for very popular items. Efficient mechanisms can be introduced in deployed systems to handle such scenarios. LookupReply messages are delivered directly to the originators of the queries.

TABLE III
LOOKUPREPLY MESSAGE STRUCTURE.

Field	Description
Common	Derived common message structure (cf. Table I).
Key	Includes the GUID describing the queried item.
IndexValue	Includes the address of the peer, which has advertised an item that matches the key.

Table IV describes the structure of the PublishRequest message. It includes a *key* representing the GUID of the queried item and the *OwnerAddress* of the advertised service or resource that matches the provided key. The PublishRequest message is delivered to the cluster that matches best with the included key and subsequently, Indexers update their local structures with the newly advertised item. Typically, advertisements have an expiration timeout. It is the responsibility of the owner to refresh the published information.

TABLE IV
PUBLISHREQUEST MESSAGE STRUCTURE.

Field	Description
Common	Derived common message structure (cf. Table I).
Key	Includes the GUID describing the published item.
OwnerAddress	Includes address of the owner of the advertised service or resource that matches the provided key.

After successfully advertising an item, the responsible Indexer provides a confirmation to the originator of the advertisement using a PublishConfirmation message. Its structure is described in Table V.

TABLE V
PUBLISHCONFIRMATION MESSAGE STRUCTURE.

Field	Description
Common	Derived common message structure (cf. Table I).
Key	Includes the GUID describing the published item.
Confirmation	Confirms the successful advertisement of the published resource.

VI. SIMULATION RESULTS

The simulation experiments have been performed using the general purpose discrete event simulator for P2P overlay networks presented in [7]. The characteristics of interest in the performed simulations include the generated traffic to locate and deliver the requested information, the distribution of the load among the participating peers and clusters, etc. The size of the evaluated P2P networks is ranging from 500 to more than 130,000 peers. During the performed experiments, the size of clusters ranges between 2 and 16 peers. The degree of the de Bruijn networks (inter-cluster degree) is mostly set to 2 and 4 connections per node. While it is possible to set it to any arbitrary integer, the interest of our research lies in fixed, low degree de Bruijn networks. The results presented in Section VI-A and Section VI-E are general and apply to any de Bruijn based overlay network. The results discussed in Section VI-B, Section VI-C and Section VI-D derive mostly from the two-tier architecture of Omicron. The critical setup parameters of the experiments are the average cluster size $\overline{K}$ and the inter-cluster degree k.

A. Scalability

Scalability is the most critical requirement for most P2P applications. In terms of core overlay network operations, the cost of LookupRequest(s) is the metric we use to evaluate the scalability of the network. It should be noted that the cost both for LookupRequest(s) and PublishRequest(s) (as they are described in Section V-C) is identical. In the rest of this chapter we refer to such messages as *queries*. The focus of the experiments is mostly on the way the communication cost increases as the network grows in size.

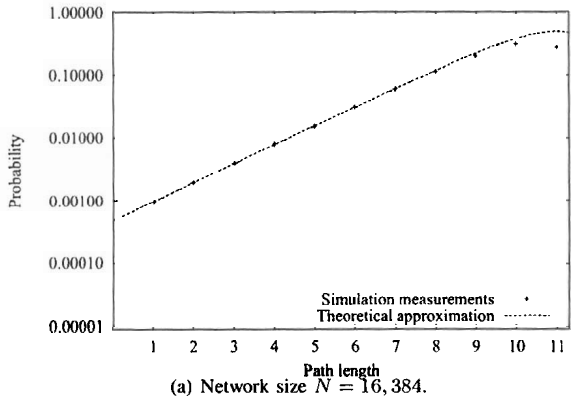
1) Impact on Shortest Path Distribution: In this section the quantity of interest is the distribution of the shortest paths between nodes. This investigation reveals the relationship between the average shortest path and the network diameter. Loguinov et al. [19] provide an approximation formula of the asymptotic distribution (PMF) of shortest paths in de Bruijn graphs:

$$p(n) \approx \frac{k^n}{N} - \frac{k^{2n-1}}{N^2}. \quad (7)$$

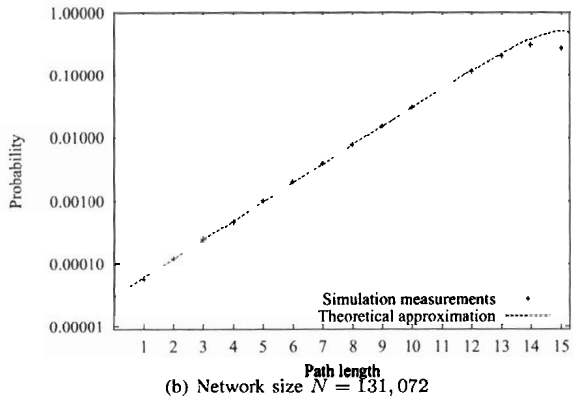
Here, N is the order of the de Bruijn network and k is the degree of each node. Unfortunately, the distribution of de Bruijn distances $d(x, y)$ between two nodes x and y is very complicated and there is no closed-form expression for its PMF [28]. Equation 7 is exact when the diameter $D \leq 3$ and very close to the real value for larger graphs.

In the related experiments many different orders of Omicron networks have been evaluated. Nevertheless, we provide only the results of two representative experiments to give a better

idea of the shortest path PMF. For these experiments, the size of the Omicron network is set to $N_1 = 16,384$ and $N_2 = 131,072$ and the average cluster size is $\overline{K} = 8$. The degree of the network is set to $k = 2$. Therefore, the order of the de Bruijn network in these two scenarios is $M_1 = 2,048$ and $M_2 = 16,384$, respectively (note that $M = N/\overline{K}$). Figure 4 provides the graphical representation of the results. The analytical approximation given by Equation 7 is drawn as a continuous function to demonstrate more comprehensively the matching of the two approaches. Figure 4(a) displays the results for the small order de Bruijn network ($M_1 = 2,048$) while Figure 4(b) displays the results for the larger de Bruijn network ($M_2 = 16,384$). Note that the vertical y-axis is logarithmically scaled. The very close fitting of the simulation results to the analytical approximation can be clearly observed. It is only the last hop that differs slightly. This deviation can be explained by the two following facts. On the one hand, the theoretical values are merely an approximation of the real values. On the other hand, the simulation environment encounters factors not present in analytical methods, e.g. the network topology may temporarily differ from the targeted structure.



(a) Network size $N = 16,384$.



(b) Network size $N = 131,072$

Fig. 4. Shortest path distribution.

It can be observed that de Bruijn graphs expand exponentially. The majority of the shortest paths between pair of nodes is very close to the diameter of the network. This explains the results of the following section on the average query length.

2) Impact on Average Query Length: The average query length can be regarded as the most relevant metric to evaluate the scalability of a P2P overlay network. Moore's law defines lower bounds for k-regular graphs as it is provided in Inequality 3. In fact, de Bruijn graphs are asymptotically optimal graphs converging to this bound for sufficiently large graph order N and node degree k. As it is shown in [19]:

$$\mu_{DB} \approx D_{DB} - \frac{1}{k-1}, \tag{8}$$

where μ_{DB} is the average asymptotic distance in de Bruijn graphs and D_{DB} is the diameter. Since the diameter of the network is already close to the Moore bound (cf. Inequality 2), the average distance cannot shrink much further.

The relevant collected measurements include experiments involving both the Omicron and the Chord networks. Initially, we consider only the scalability of the Omicron network with varying average cluster size between three values ($\overline{K}_1 = 4$, $\overline{K}_2 = 8$, $\overline{K}_3 = 16$). Figure 5 provides the graphical representation of the experiments. Both x-axis and y-axis are logarithmically scaled to provide a more comprehensive view. The average query length is constituted both of the inter-cluster routing (de Bruijn based) towards the targeted cluster and the intra-cluster step to reach an Indexer. In the particular experiments routing is performed by peers assigned only with the Router role. No Cachers are present in this experiment, therefore, Routers access directly an Indexer of the cluster to get the queried information. The close matching of the analytical approximation and the simulation results can be evidently observed from this figure. Moreover, it can be stated that as the size of clusters grow exponentially the average routing length decreases linearly. Therefore, as the cluster maintenance cost grows exponentially, it can be safely assumed that there is an optimal maximum value for the size of the clusters beyond which, the cost grows more quickly than the utility.

In addition to the absolute performance of Omicron, it is essential to compare it with a widely known system like Chord (which can be considered as a reference point). Figure 6 shows the related simulation results. It should be noted that both X and Y axes are logarithmically scaled to provide a more comprehensive view. As it can be observed from this figure, Chord outperforms the Omicron network configuration with average cluster size $\overline{K} = 8$ and node degree $k_1 = 2$. However, it should be noted that Chord's design requires a logarithmically increasing node degree. By similarly increasing the node degree of Omicron, the average query length is getting significantly smaller than Chord. To analytically express this, if the diameter in Equation 8 is replaced by the formula of

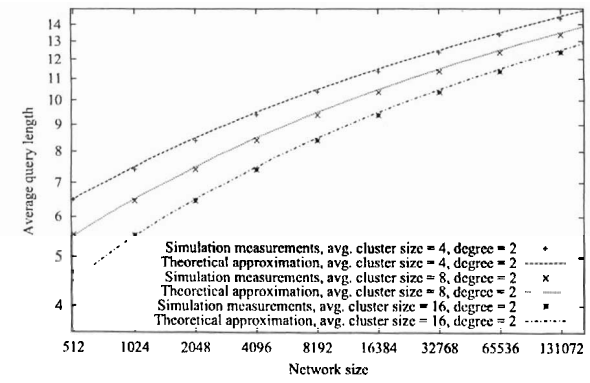


Fig. 5. Routing scalability.

Equation 4 we get:

$$\mu_{DB} \approx \frac{log(N)}{log(log(N))} - \frac{1}{log(N) - 1}. \quad (9)$$

Chord's average length approximation is give by $\mu_{CH} = log(N)/2$. Therefore, for similar node degree, Omicron achieves smaller average query length. In fact, for the network order range described in Figure 6, by setting Omicron's node degree to $k_2 = 4$, Omicron clearly outperforms Chord. Again here, it can be noted the good matching of the analytical approximation and the simulation results.

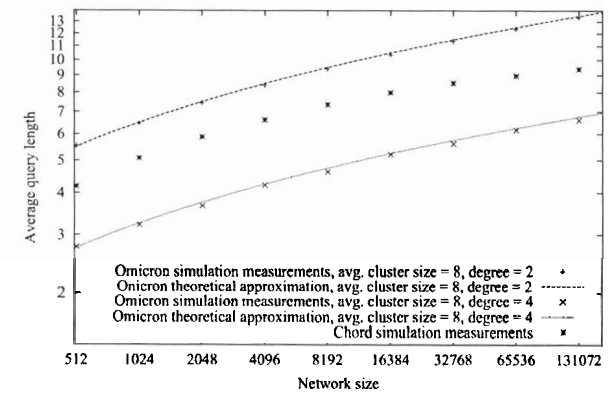


Fig. 6. Routing scalability: Omicron versus Chord.

B. Vicinity exploration

The focus of the previous section is on the average shortest paths between nodes. The length of the path is expressed in number of hops required to reach the final destination. Each step contributes equally with weight 1. While such a metric expresses well the complexity of the designed network, it lacks the ability to capture the accomplished efficiency on mapping the P2P overlay network to the underlying network infrastructure.

In fact, many P2P overlay networks do not consider at all the efficient mapping in their design. For instance, Chord's algorithm determines uniquely the neighbor peers (both successors and fingers) of each individual peer. Therefore, peers cannot select neighbors that are in the closest vicinity.

In contrast, Omicron has this freedom since any neighbor peer with the needed role (i.e. Router, Indexer, etc.) in a cluster is similarly adequate[3]. It is the purpose of this section to explore this ability quantitatively and observe the improvement in end-to-end distance.

In order to define the underlying network distance between two peers, a simple, two-dimensional torus-based model with Euclidian distances is utilized by the simulator. The cost of a connection is analogous to their distance, which we shall call *"virtual delay"*. For the performed experiments, the Euclidian space contains a 2×2 area. Therefore, the maximum virtual delay between two peers is $\sqrt{2}$ delay units (please, consider the properties of toruses). This model has been been integrated to the discrete event simulator designed for P2P overlay networks [7]. Moreover, it should be noted that for the calculated distance in these experiments, the random effects introduced within the simulator to estimate the transmission delay have been explicitly removed.

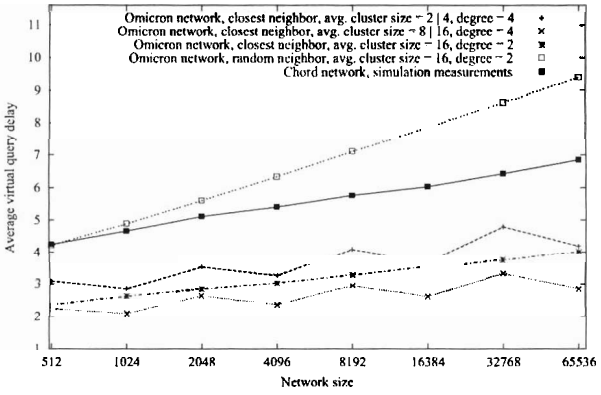


Fig. 7. Vicinity exploration: Omicron versus Chord.

The performed experiments include four different Omicron configuration settings and a typical Chord network. The average cluster size varies between 2 to 16 peers and the inter-cluster node degree varies between 2 and 4. Two different policies are utilized for Omicron: (i) The first one is a *random neighbor selection* based policy, unaware of the distance between the peers. Such policy has similar effects to the policy followed by Chord. (ii) The second policy selects the closest one among the peers of the neighbor cluster. However, in the performed experiments peers may select only their neighbor Routers. Their connection to the local Indexer (belonging to the same cluster) is locality unaware. The last selection has been introduced to reflect scenarios where a small number of

[3]We ignore issues such as trust or reliability at this selection. However, they are important aspects for consideration.

reliable peers assigned with the Indexer role are available. This "greedy" algorithm is expected to perform more efficiently than the random selection policy.

As it can be observed from Figure 7, the Omicron network based on the random selection policy and the Chord network perform inferiorly compared to the closest neighbor approach. In fact, their relative performance is analogous to the hop based distance metric (studied in the previous section) since the large number of uniformly distributed peers is the determining factor. In contrast, the closest neighbor based experiments exploit peer vicinity much more efficiently. In fact, the performance is even better when peers (Routers) can optimally select the targeted Indexer, too. Further, the non-monotone curves observed in cases where the inter-cluster node degree is 4 is an interesting point. The explanation for this effect is based on the fact that some network sizes cannot be defined as integral powers of the node degree (4). Therefore, the average cluster size varies between two values (i.e., 2 and 4 or 8 and 16).

C. Query Workload Distribution

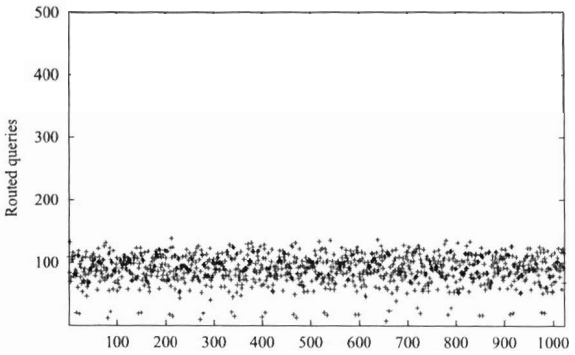
Omicron has been specifically designed to deal efficiently with the heterogeneity of the peers. In order to do this, several mechanisms have been introduced. However, it is desirable to accomplish a relatively even distribution of the workload among peers assigned with similar roles. This section focuses specifically to the distribution of the routing workload, aiming to find potential bottlenecks that can degrade the performance of the system.

In Section VI-B, we compared the accomplished efficiency in mapping the overlay connections to the underlying network cost. It has been observed that vicinity aware neighbor selection can considerable improve the observed end-to-end delay. However, it is interesting to examine how the described greedy algorithm affects the routing workload distribution among the Routers.

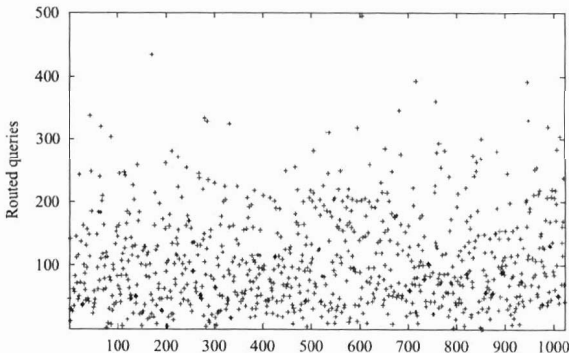
On the one hand, Figure 8(a) provides the query workload distribution for a typical Omicron configuration, where neighbors are randomly selected. The order of the network is $N = 1,024$ peers, where the inter-cluster node degree is $k = 2$ and the average cluster size is $\overline{K} = 16$. A relatively even workload distribution can be observed. On the other hand, Figure 8(b) shows the accomplished workload distribution when peers select their neighbor based merely on proximity criteria. Apart from the connection policy, the two networks are identical.

Obviously, the greedy algorithm has a negative effect on the load balance. For certain pairs of peers the assigned relative workload ratio is as high as 50 times (regarding the specific network configuration described above). Exploring more deeply the details of the simulation experiments the critical factor has been identified, which results to this undesirable effect.

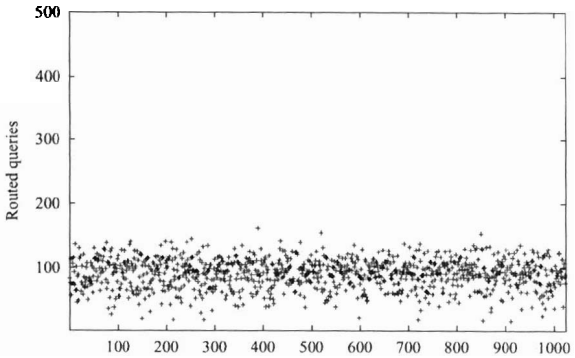
For single-tier de Bruijn networks (where a node represents a single peer), it is guaranteed that both the incoming and the outgoing node degrees are equal and fixed (with the exception of particular nodes, as it is discussed in Section V. However, for cluster based de Bruijn networks there is no such guarantee. In fact, only the outgoing degree is equal and fixed ($k = 2$ for the investigated scenario). In contrast, the random neighbor based network is constructed by enforcing also the equal and fixed incoming node degree, resulting in the observed load distribution.



(a) Random neighbor.



(b) Closest neighbor.



(c) Restricted closest neighbor.

Fig. 8. Query routing load distribution.

Therefore, it is rational to examine whether by applying this restriction to the vicinity aware network *(restricted closest neighbor)*, improved load distribution results could be achieved. However, such a restriction results to non optimally selected neighbors. Figure 8(c) shows the results of such a scenario. Peers accept new neighbors as long as their incoming degree is less than or equal to their outgoing degree. If a new connection request arrives while they have reached their maximum allowed incoming degree, the connection request is refused and the originator peer requests a connection from a different peer of the same cluster. It should be noted that such a policy is not optimal. A better strategy can be exploited, e.g., by disconnecting the furthest among the currently connected peers. Nevertheless, even this simple policy can provide the desirable results as we can see in Figure 8(c). Practically, the query routing load distribution is indistinguishable between randomly selected neighbors and the restricted closest neighbor selection policy.

D. Workload Distribution and Network Vicinity Exploitation Trade-off

The experiments performed in the previous section evaluating the routing workload distribution revealed that the restricted closest neighbor policy has a desirable distribution. However, it is not expected to exploit the network vicinity as efficiently as the greedy algorithm. Therefore, it is essential to evaluate the ability of the new algorithm to exploit network vicinity. The network size varies between 512 to 65,536 peers. Peers submit queries to randomly selected destinations. The queries are forwarded to the targeted cluster via Routers belonging to neighbor clusters; then, a local Indexer (of the targeted cluster) replies.

Figure 9 provides the collected results where the x-axis scales logarithmically in order to provide a more comprehensive view. The three different policies are graphically displayed with curves. As it can be observed, the restricted closest neighbor policy performance is close to the greedy closest neighbor policy. It is expected that a more advanced policy that disconnects the furthest among the connected neighbors can provide even better results. Moreover, if the random neighbor Indexer policy is replaced by a vicinity based one, both vicinity aware policies perform better.

E. Enhanced de Bruijn Routing Algorithms

As it has been mentioned in Section V-A, particular de Bruijn nodes that are self-connected receive a significantly less routing workload compared to the rest of the nodes. The number of self-connected nodes in a de Bruijn graph are k. The provided solution develops an enhanced de Bruijn structure where self connected links are removed and replace by others that connect these nodes (c.f. Figure 3). The focus of this section is to evaluate the improvement accomplished with the new structure.

The related experiments involve the original and the enhanced de Bruijn digraphs. For these experiments, one peer per cluster submits queries targeted to each existing cluster of the network (resulting in M^2-M queries, where M is the number of clusters). The collected measurements include the number of messages forwarded by each cluster (aggregating all the forwarded messages of Routers belonging to each particular cluster). The node degree is $k = 2$ and the results shown in Figure 10 involve 512 constructed clusters. The y-axis scales logarithmically in order to provide a more comprehensive view.

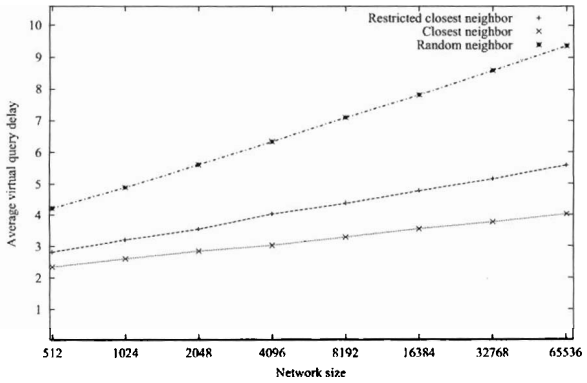


Fig. 9. Vicinity exploration versus load-balance.

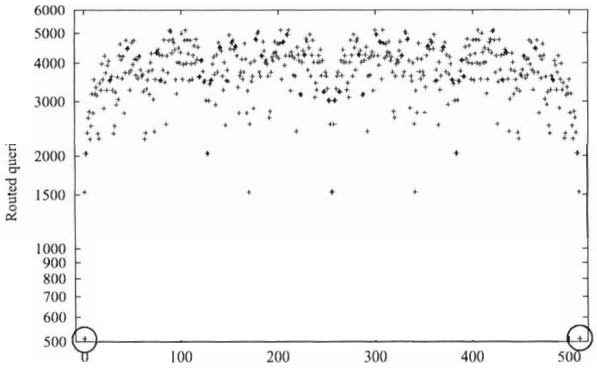
As it can be observed from Figure 10(a), self connected clusters having GUID (000000000) and (111111111) receive 511 ($M-1$) queries. This means that only the queries that are targeted to these clusters arrive there and no further queries are forwarded via them to other clusters (the initial submission of the query is not accounted in these measurements). Certainly, this is not a desirable behavior.

In contrast, by examining Figure 10(b) we note that the workload of the modified clusters (by replacing their self-connected links) is considerable higher (twice the load observed in the other scenario). It should be noted that Algorithm V.3 (used in this experiment) aims mostly at providing optimal shortest paths. By replacing this algorithm it is possible to receive even better routing workload balance at the cost of longer paths. Such an algorithm is described in [28].

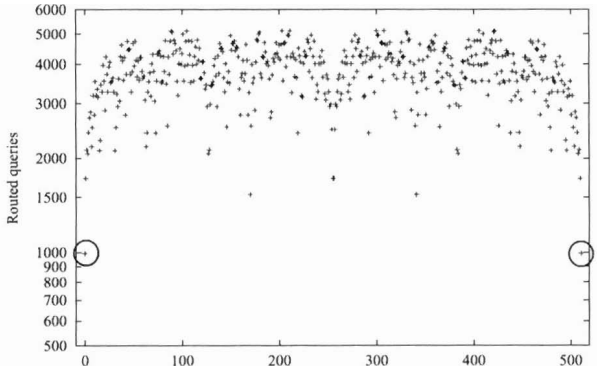
Aiming to explore the effect of the enhanced de Bruijn topology with a wider network size range, we provide a comparison of the routing workload distribution between the two topologies. Figure 11 shows the achieved load on the self-connected clusters where both X and Y axes scale logarithmically in order to provide a more comprehensive view. As it can be observed the enhanced topology distributes almost 50% additional load to the self-connected clusters compared to the original structure for all ranges of the explored network sizes. It should be noted that the x-axis is the size of the de Bruijn network and not the size of the complete Omicron network.

VII. Conclusions

This paper discusses the routing mechanisms employed in a two-tier P2P overlay network based on de Bruijn digraphs.



(a) With self-connected nodes.



(b) Without self-connected nodes.

Fig. 10. Routing query distribution on de Bruijn graphs.

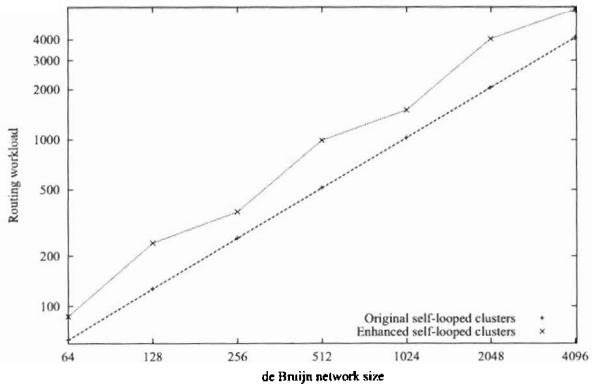


Fig. 11. Routing workload on self-connected nodes.

The required algorithms and protocol messages have been discussed in order to shed the necessary light in these core system components.

The specialization based approach introduced with the role model and the two-tier network architecture provide an advantageous mechanism to exploit heterogeneous populations of peers. The common characteristics of the utilized messages have been identified in order to develop them based on a base structure and furthermore, to seamlessly handle them with common mechanisms, i.e., message dispatcher.

Special attention has been paid to design an efficient routing mechanism. The original structure of de Bruijn graphs has been enhanced by removing the self-connected links of particular nodes. Thereby, the optimal routing algorithms found in literature have been adapted to take advantage of the new structure and to provide a better balance of routing traffic.

Additionally, the quantitative results of the experiments performed on Omicron and Chord provide an evaluation of their performance in several key scenarios. The results reveal the superiority of de Bruijn based overlay networks. The *scalability* of the routing process follows accurately the analytical approximations provided in the literature, both for the average and the worst case scenarios. Also, the evaluated PMF on the average length of the queries fits nicely with the theoretical approximation, proving the *expanding* properties of de Bruijn graphs. Similar results are provided for the distribution of the shortest paths between peers. The intrinsic *load balance*-related limitations of de Bruijn digraphs have been evaluated together with the suggested enhanced digraph topology. Moreover, the ability of Omicron and Chord to exploit the underlying *network vicinity* is evaluated with simulations, revealing the advantages of the two-tier architecture. The *trade-off* between efficient underlying network exploitation and even routing workload distribution is investigated too.

ACKNOWLEDGMENT

We are grateful for the international collaboration and the fruitful discussions we had within the MMAPPS project, which helped us finalize many design aspects of this work. Moreover, this work has been partly supported by the European Union under the E-Next Project FP6-506869.

REFERENCES

[1] F. Bernabei, V. D. Simone, L. Gratta, and M. Listanti. Shuffle vs. Kautz/De Bruijn Logical Topologies for Multihop Networks: a Throughput Comparison. In *Proceedings of the International Broadband Communications*, pages 271–282, 1996.

[2] W. Bridges and S. Toueg. On the impossibility of directed Moore graphs. *Journal of Combinatorial Theory Series B*, 29:339–341, 1980.

[3] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *ICSI Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[4] C. Cramer and T. Fuhrmann. On the fundamental communication abstraction supplied by P2P overlay networks. *European Transactions on Telecommunications*, December 2004.

[5] V. Darlagiannis. *Overlay Network Mechanisms for Peer-to-Peer Systems*. PhD thesis, Department of Computer Science, Technische Universität Darmstadt, Germany, June 2005.

[6] V. Darlagiannis, N. Liebau, O. Heckmann, A. Mauthe, and R. Steinmetz. Caching Indices for Efficient Lookup in Structured Overlay Networks. In *Proceedings of the Fourth International Workshop on Agents and Peer-to-Peer Computing*, July 2005.

[7] V. Darlagiannis, A. Mauthe, N. Liebau, and R. Steinmetz. An Adaptable, Role-based Simulator for P2P Networks. In *Proceedings of the International Conference on Modeling, Simulation and Visualization Methods*, pages 52–59, June 2004.

[8] V. Darlagiannis, A. Mauthe, and R. Steinmetz. Overlay Design Mechanisms for Heterogeneous, Large Scale, Dynamic P2P Systems. *Journal of Networks and System Management*, 12(3):371–395, 2004.

[9] N. G. de Bruijn. A combinatorial problem. In *Proceedings of the Koninklijke Academie van Wetenschappen*, pages 758–764, 1946.

[10] M. Fiol and A. Llado. The Partial Line Digraph Technique in the Design of Large Interconnection Networks. *IEEE Transactions on Computers*, 41(7):848–857, 1992.

[11] M. Fiol, L. A. Yebra, and I. A. de Miquel. Line Digraph Iterations and the (d,k) Digraph Problem. *IEEE Transactions on Computers*, 33(5):400–403, 1984.

[12] P. Fraigniaud and P. Gauron. An Overview of the Content-Addressable Network D2B. In *Annual ACM Symposium on Principles of Distributed Computing*, July 2003.

[13] C. Gavoille. Routing in Distributed Networks: Overview and Open Problems. *ACM SIGACT News - Special Interest Group on Automata and Computability Theory)*, 32:36–52, 2001.

[14] O. Heckmann, N. Liebau, V. Darlagiannis, A. Bock, A. Mauthe, and R. Steinmetz. *From Integrated Publication and Information Systems to Information and Knowledge Environments: Essays Dedicated to Erich J. Neuhold on the Occasion of His 65th Birthday*, volume 3379 of *Lecture Notes in Computer Science*, chapter A Peer-to-Peer Content Distribution Network, pages 69–78. Springer-Verlag GmbH, January 2005.

[15] F. Hsu and D. Wei. Efficient Routing and Sorting Schemes for de Bruijn Networks. *IEEE Transactions on Parallel and Distributed Systems*, 8(11):1157–1170, 1997.

[16] F. Kaashoek and D. R. Karger. Koorde: A Simple Degree-optimal Hash Table. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS03)*, February 2003.

[17] Z. Liu. Optimal routing in the de Bruijn networks. In *Proceedings of the 10th International Conference on Distributed Computing Systems*, pages 537–544, May 1990.

[18] Z. Liu and T.-Y. Sung. Routing and Transmitting Problems in de Bruijn Networks. *IEEE Transactions on Computers*, 45(9):1056–1062, 1996.

[19] D. Loguinov, A. Kumar, V. Rai, and S. Ganesh. Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience. In *Proceedings of ACM SIGCOMM'03*, pages 395–406, August 2003.

[20] J.-W. Mao and C.-B. Yang. Shortest path routing and fault-tolerant routing on de Bruijn networks. *Networks*, 35(3):207–215, 2000.

[21] P. Maymounkov and D. Maziéres. Kademlia: A Peer-to-peer Information System Based on the XOR metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02)*, 2002.

[22] J. Mischke and B. Stiller. Design Space for Distributed Search $(DS)^2$ - A System Designers' Guide. Technical report, ETH Zurich, Switzerland, TIK Report Nr. 151, September 2002.

[23] J. Mischke and B. Stiller. Rich and Scalable Peer-to-Peer Search with SHARK. In *5th International Workshop on Active Middleware Services (AMS 2003)*, June 2003.

[24] M. Portmann, P. Sookavatana, S. Ardon, and A. Seneviratne. The cost of peer discovery and searching in the Gnutella peer-to-peer file sharing protocol. In *Proceedings of the International Conference on Networks*, pages 263–268, 2001.

[25] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable Content Addressable Network. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172. ACM Press, 2001.

[26] S. Ratnasamy, S. Shenker, and I. Stoica. Routing Algorithms for DHTs: Some Open Questions. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02)*, 2002.

[27] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, 2001.

[28] K. Sivarajan and R. Ramaswami. Lightwave networks based on de Bruijn graphs. *IEEE/ACM Transactions on Networking (TON)*, 2(1):70–79, 1994.

[29] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: A scalable Peer-to-Peer Lookup Service for Internet Applications. *IEEE Transactions on Networking*, 11(1):17–32, February 2003.

[30] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz. Tapestry: A Resilient Global-scale Overlay for Service Deployment. *IEEE Journal on Selected Areas in Communications*, 22(1):41–53, 2004.