The Design of PAMANET
The Paderborn Mobile Ad-Hoc Network
(Extended Abstract)

Peter Bleckmann Stefan Böttcher Eduardas Cesnavicius
André Francisco Tim Daniel Hollerung Birger Kühnel Jing Liu
Sebastian Obermeier Simon Oberthür Felix Peter Franz Rammig
Christian Schindelhauer Gunnar Schomaker Thorsten Steenweg
Qamar Abas Tarar Marcel Tiemeyer Adelhard Türling Arne Vater

University of Paderborn*
Institute of Computer Science
Contact: Christian Schindelhauer
`schindel@uni-paderborn.de`

July 4, 2004

Abstract

Wireless connectivity is state of the art for local area networks. Currently, most W-LAN networks rely on a centralized design with access points routing all inner and outbound traffic. These access points are intrinsic communication bottlenecks. Mobile Ad Hoc Networks (MANET) overcome this problem, because every participant works as well as a simple node and as a router. Current MANETs are restricted in scalability, because they rely on flooding mechanisms or complete routing tables. Other approaches, providing better scalability use clustering, yet network performance deteriorates in case of high node mobility.

We describe the design of a PAMANET, the Paderborn Mobile Ad Hoc Network, a MANET overcoming these problems providing scalability and reliability in a mobile scenario. When implemented, PAMANET works with standard W-LAN IEEE 802.11 radio devices, provides IPv6 communication interfaces and works on personal computers under a standard Linux distribution.

First, we present current routing protocols and classify them with respect to scalability and stability in dynamically evolving MANETs. Then, we discuss related research in the area of distributed hash tables and consistent hashing, used for relieving hot spots in the Web, storage area networks and peer-to-peer networks, which inspires the design of PAMANET.

PAMANET consists of three main components: First, the embedding of the routing layer into IEEE 802.11 and IPv6 by using techniques used at the ad hoc support library (aslib) by Gupta et al. Second, the routing layer which combines a landmark routing, hierarchical clustering, consistent hashing for providing location dependent addresses and lookup-service for the location of nodes. Third, a peer-to-peer data storage system based on egoistic distributed caches enabling hop and traffic efficient data access on replicated data partitions.

*This work was partially supported by the Special Research Initiative (DFG-Sonderforschungsbereich) 614 - Self-optimizing Concepts and Structures in Mechanical Engineering and the Special Research Initiative (DFG-Sonderforschungsbereich) 376 Massive Parallelism, partial project C6, Mobile Ad Hoc Networks of the University of Paderborn, and was published on its behalf and funded by the Deutsche Forschungsgemeinschaft.

The routing layer incorporates a variety of new approaches. Link distances reflect the failure probability of links, which is estimated by the reciprocal age of the link. Then, we combine a landmarking system on this metric with the hierarchical layer graph yielding small landmark addresses and small routing tables. To balance the load of the distributed lookup-service for landmark addresses, a hierarchical weighted consistent hashing scheme is used. This ensures that each node receives an equal part of all landmark addresses. Using these mechanisms (regularly and on demand) PAMANET adjusts IPv6 routing tables such that short stable routes are preferred.

For the distribution of control data like landmark information PAMANET uses a message box system interface to provide fast one-hop communication. On top of this system, PAMANET provides a peer-to-peer data storage and lookup system that realizes time, traffic, and load efficient access using egoistic caches and data segmentation strategies.

1 Introduction

A Mobile Ad Hoc Network is an autonomous, self-configuring wireless network, where every participant behaves like a node and is also responsible for routing tasks. For an nearly entire overview, we refer to the excellent book of Charles Perkins [Per01]. We give now a brief overview of a few relevant MANET protocols.

1.1 Mobile Ad-hoc Network Protocols

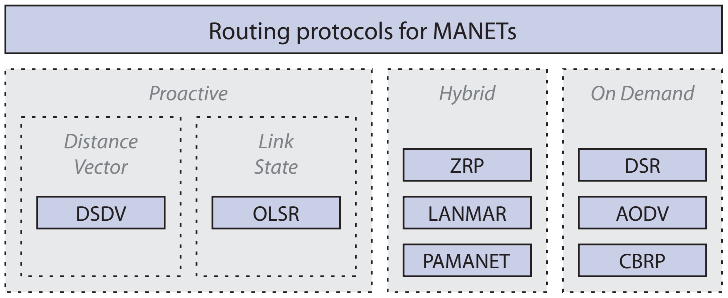

Figure 1: Categorization of MANET protocols

Commonly mobile ad hoc network protocols are classified into three categories: *proactive*, *reactive* and *hybrid protocols*. Proactive protocols permanently communicate for maintaining valid routes before they are needed, whereas reactive protocols explore possible routes on demand. Hybrid protocols are using a combination both.

Examples for proactive protocols are the Destination-Sequenced Distance-Vector Routing (DSDV) [PB94] and the Optimized Link State Routing Protocol (OLSR). In distance vector routing each node maintains a routing table. This table stores the distance to each destination, to obtain the best rout and the neighbor to achieve this. The algorithm is also known as the distributed Bellman-Ford routing [Tan96].

In DSDV, new routes and updates to existing routing information are always initiated by the destination node. Thus, routes are constructed from destination to source using periodic update messages. To distinguish old and new routing information, sequence numbers are included. Thus DSDV maintains long routing tables, growing linearly with the number of network nodes. According to this, the number of control messages grows too, because the connectivity information of each node is periodically broadcasted through the whole network. Eventually DSDV does not provide large networks, but performs very well in small scaled environments [BMJ$^+$98].

Using routing protocols relying on link-state routing mechanism, implies for each node to maintain a complete view of the network, i.e. every node knows all connections. The Optimized Link State Routing

(OLSR) protocol [CJL$^+$03] is an optimization of the classical approach. The enhancement is based on Multipoint Relays (MPRs). Every node selects a set of its neighbors as its MPRs. Such an MPR is responsible for forwarding broadcast information sent by his selectors and in addition MPRs are the only nodes in the network that generate link state information. In case of flooding link state information, OLSR requires only partial link state information. This reduces the amount of data that has to be sent and still provides enough information to provide shortest routes. OLSR uses periodic broadcast of *hello* packets to sense the neighborhood of a node and to verify the symmetry of radio links. Like DSDV, OLSR is afflicted with a similar overhead for large networks and in addition, its energy consumption is high.

On demand routing protocols explore and discover routes only when they are needed. This strategy avoids periodically updates of routing information. If a desired route is discovered and established, it is maintained by route maintenance procedures. Route maintenance is performed until either the route is no longer needed or the destination node becomes inaccessible.

Dynamic Source Routing (DSR) [JM96] exemplifies an on demand routing protocol for mobile ad-hoc networks. It relies on two mechanisms: Route discovery and route maintenance. Each node maintains a route cache which contains already discovered routes. Routes are added to the route cache when they have been found on demand. Route discovery is accomplished with route request and route reply packets. The route maintenance process deletes routes from all node caches of all nodes, beginning at the initiator node discovering a route breakage, back to the source node. After this another route discovery process is started. Like DSDV, DSR performs well in small networks with sparse movement of nodes, but the routing overhead increases if mobility increases and finally the performance decreases too.

Further reactive protocols are the Ad-hoc On-Demand Distance-Vector Protocol (AODV)[PR99] and the Cluster Based Routing Protocol (CBRP)[JL99]. As above for these protocols the network performance degrades for large or highly dynamic networks .

An example for an hybrid routing protocol is the *Zone Routing Protocol* (ZRP) [Haa97]. ZRP provides routing in a hierarchical network architecture. It defines a routing zone for each node. Each routing zone has a predefined radius. The whole network topology within this zone is known by the corresponding node (proactive part). Updates concerning a specific node are propagated only within its routing zone. Routing in this network can be accomplished if all neighbored routing zones sufficiently overlap. Routes to nodes outside a node's routing zone are explored on demand, by sending queries to all nodes in the periphery of the corresponding node's zone. This process is repeated until the destination node is known by a node that received the query. This protocol reduces communication traffic compared to the other protocols. Nevertheless, full scalability is also not provided.

Landmark ad hoc Routing Protocol (LANMAR) [XHG03] is well suited for an ad hoc network that exhibits group mobility. Therefor logical sub-nets are identified in which the members have a common interests and are likely to move as a "group". Each logical group has one node serving as a "landmark". When the destination is within the scope of the source, a local routing algorithm is used. For nodes out of scope, the packets will be routed towards the landmark of the destination. LANMAR reduces the control overhead by truncation of local routing tables and the contraction of routing information to remote node groups. Routing table sizes are reduced as well, since only routing information within local scope is in details. For the nodes out of scope, routing table only provides information for routing to the landmark of the destination nodes.

An open question in LANMAR is how to find appropriate groups and the lookup service for the landmark addresses. We will follow this track and use LANMAR as a building block for PAMANET.

PAMANET is similar to an hybrid network like ZRP and LANMAR. The main difference is the multi-hierarchical routing approach. Each node in the network acts as a landmark and thus has a unique landmark address. Depending on the network size there may be many different levels of landmarks. Like ZRP, each node knows the network topology in a predefined radius (measured in number of hops), but the size of radius depends on the hierarchy level the node belongs to. In addition the node knows the next hops to all higher landmarks. Routes to landmarks outside the zone of known nodes are discovered on demand by comparing landmark addresses of sending node and source node followed by forwarding the data packet to the neighboring node which is next hop to the smallest common landmark node compared to source and destination node.

1.2 Hierarchical Layer Graph

Recent work in communication networks shows that hierarchical partitioning comprises a huge potential for efficient networking. E.g. for static networks Harald Räcke [Räc02] has proved that a tree decomposition leads to an oblivious routing scheme such that its congestion approximates optimal congestion by a poly-logarithmical factor.

If the transmission power of participating nodes is adjustable, then congestion, dilation and energy can be approximated by a basic network topology called Hierarchical Layer Graph (HL-graph) [MSVG04]. This network design provides small number of interferences and small number of nodes. Such HL-graph is constructed starting from the bottom, where layers of networks of comparable edge lengths are constructed by reducing the number of participating nodes. For this, if nodes in a layer i get closer than distance β^i, a node is degraded to level $i-1$. On the other hand, a node of level $i-1$ is upgraded to level i if it does not violate this distance rule to existing level i nodes. Nodes in level i are connected to each other within distance $\alpha \cdot \beta^i$. The construction of such a graph from scratch can be performed in $O(\log^2 n)$ steps.

In [SLRV03a] it is shown that in a worst case mobility scenario such a Hierarchical Layer Graph also provides good properties like scalability, small number of interferences, small congestion and a small degree. We used a modified version of the Hierarchical Grid Graph in a higher-dimensional space, therefor we added extra coordinates to the nodes. These coordinates, in addition to the position coordinates, are used as speed vector.

The IEEE 802.11 wireless communication protocol does not allow to adjust transmission range and the relative coordinates and relative velocities between network nodes remains unknown. Nevertheless, PAMANET uses this worst case mobility approach. For this, the node distance in the four or six-dimensional space of [SLRV03a] is approximated by the link distance, which is calculated using the age of a communication link. Based on this metric we build the Hierarchical Layer Graph replacing single hops by multi-hop routes. Thus, PAMANET provides scalability as well as reliability in a mobile environment.

1.3 Consistent Hashing

Consistent Hashing, introduced by Karger et al. [KLL+97] and extended in [KLLS99], is a distributed data structure originally designed for relieving hot spots in the Internet. Hot spots in the Internet occur if a large number of clients simultaneously acquires data from a single web server. Thus the information host is swamped and unreachable. Consistent hashing provides a solution for this problem, by using a hash function. This causes for the addition or deletion of servers only minimal reassignment of data placement.

In PAMANET, we applied an adapted version of this technique to find for each node the corresponding landmark address, which is essential for route determination. The main idea is that the unique address (IP- oder MAC-Address) of a node is used as key to obtain information holder in the network where the destination node has deposited its landmark information. Then, consistent hashing functions lead the lookup-query through the Hierarchical Layer Graph by determining the correct sub-cluster. PAMANET uses a new variant of weighted consistent hashing that fairly balances this network information among the responsible nodes.

1.4 P2P-Networks

PAMANET and second generation peer-to-peer-networks share several concepts. E.g. all mobile hosts are equal and the communication load and memory usage is balanced. Both networks apply consistent hashing, referring to other efficient peer-to-peer-systems like CHORD or CAN [SMK+01, RFH+01]. Furthermore, PAMANET incorporates distributed database system, use multiple copies of entries for robustness as being proposed for peer-to-peer-networks like CHORD, CAN, Tapestry [HKRZ02], and some others as well.

However, there are some crucial differences to peer-to-peer-systems. First the underlying communication network is not used as a black box. On the contrary, the consistent hashing and data lookup are basic services that enable routing in the ad-hoc network. For this the mobile ad-hoc-network builds up an onion like structure. Every layer has a full-featured routing and a database, such that the complete failure (or disconnect) of the majority of nodes does not inflict still possible communication lines. Furthermore, this ad-hoc-network uses routes which are only a constant factor longer than the shortest stable route. Most peer-to-peer-networks do not guarantee such strict bounds.

1.5 Storage Area Networks

The main technical change in storage area networks (SAN) is to replace the 'old' SCSI-bus by a network, to obtain server independent and highly available storage. The SAN concept describes how to attach storage and use or distribute data via an accessible network. State of the art SANs, supporting metropolitan area networks (MAN), are commonly connected by fibre channel networks. These networks offer a high bandwidth and low access latencies. The key is to find appropriate placement strategies. These strategies realize striping, mirroring or combinations of both. Some of them, like RAID IV, also provide redundant data for guaranteeing fault tolerance [ET03] For further concepts for storage area networks we refer to [MCC03] and [CL00].

Motivated by Karger et.al [KLL$^+$97], Brinkmann et.al. [BSS02] presented a compact and adaptive placement scheme called SHARE, for non-uniform distribution requirements. This work proves that consistent hashing is applicable for heterogeneous demands in SAN environments. One of the major achievement of this work is the choice of a weighted consistent hashing function that reflects the size of the connected storage devices. PAMANET uses a variation of this concept, which will be completely described in the full version of this paper. The main difference is that PAMANET works totaly distributed in a highly dynamic setting, where only sparse information is available.

1.6 XML Database using Distributed Caches

XML-Caching techniques for databases have recently been investigated in different contributions ranging from physical approaches based on XML nodes to logical approaches based on fragments that are described by arbitrary logical XPath expressions. We have followed the approach of [ASV01] by using tree patterns, which can be regarded as a compromise between physical and logical structuring of cache fragments. Tree patterns are used for expressing different logical subclasses of the data space defined by the document's document type definition (DTD). In contrast to [ASV01], where missing data is represented by logical tree expressions, we use this representation for the cached data.

Like in [AKJP$^+$02] we aim at efficient data caching. We do not follow their strategy of testing whether cached data can contribute to a new query with the help of intersection testers, as (according to [GKP03]) such tests can be NP-hard and resource consuming. Instead our framework uses a set of pre-calculated cacheable XML fragments, making such tests unnecessary. Similarly as in [YS02] we have a peer-to-peer like network consisting of caching peers. In contrast to this approach, we still use an information provider in terms of a data origin peer. Our approach follows [BT04] which focuses on efficient computation on the requesting peers and thereby consumes only minimal resources. Beyond [BT04] we use caching with the benefits of consistent hashing as introduced in [KLL$^+$97].

2 System Overview

The PAMANET is designed as an IPv6 enabled, extremely scalable, mobile ad hoc network, for highly dynamic network demands. The idea is to replace the routing, based on locality information given by an

IP address, with a multi-hop routing and a special location service to find nodes using the IP-address as search key.

Therefor we develope a strategy using **Hierarchical Clustering** and **Landmarking** which defines a virtual positioning system within this network. Every node is a landmark for other nodes. Most of them only for their local neighborhood, others for a larger area, and at the top only one of them for the whole network. Due to its position, each node informs its neighborhood within the hierarchy that is defined via the **Neighborhood Management**. This sub-structure builds up the **Position Based Routing** for PAMANET that is based on this landmark address system.

Due to the fact that we want to provide standard TCP and UDP connections in the transport layer, the routing layer provides a standard IPv6 interface to the upper layer as shown in Figure 2. If the next hop to a given destination is set within the forwarding table, **PAMANET routing** uses the default **IPv6 stack**. Otherwise we have to identify this next hop. Therefor we use **Weighted Consistent Hashing** for storing information hierarchically within clusters, deduced from the landmark hierarchy.

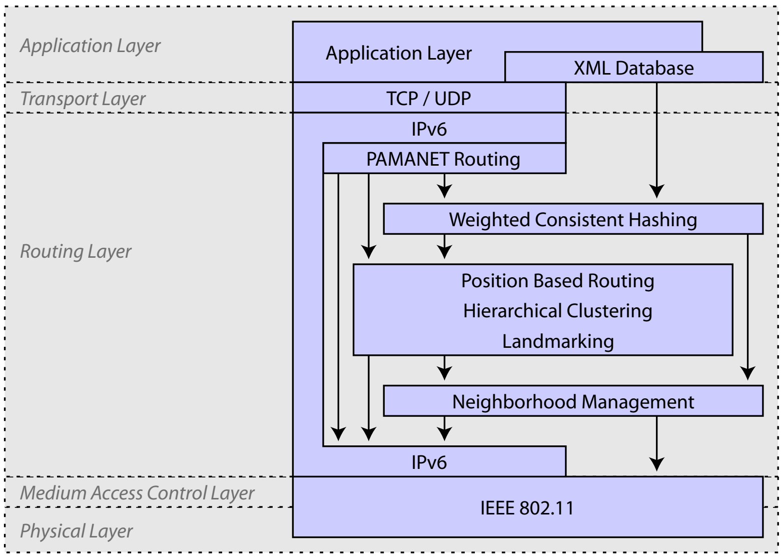

Figure 2: System Overview of the PAMANET

Each node entering the network publishes its landmark addresses by using a weighted hash function with its unique address as key. This function specifies where a node has to store the relation between its landmark addresses and its IP address.

If a node does not know the next hop to a chosen destination, it can use the same weighted hash function as above, again with the destinations IP address as the key, to achieve the landmark addresses of the destination. Based on the obtained addresses position based routing can be used to reach the destination.

For the interface to the physical layer we provide a standard IPv6 interface. It passes IPv6 packets supplemented with control information of the routing layer to the **IEEE 802.11** standard communication layer. There messages are transmitted in ad hoc mode.

As shown in Figure 2 an XML database using distributed caches is provided on top of the transport layer of every client in our PAMANET. It uses key based routing as well as the IPv6-stack for communication purposes. Certain clients are lookup-peers which store location information about the XML database.

3 Components

We now give more detailed descriptions of the components of PAMANET. The complete descriptions will appear in the full paper as well as all proofs of theorems and lemmas omitted in this design description.

3.1 Embedding and Interface to IPv6

Standard network stacks of modern operating systems like Linux or Windows are built for proactive routing. Packets are forwarded to the next hop according the routing table in the kernel, update by routing algorithms. When using on-demand and hybrid protocols, packets must be queued, while the route is discovered. This happens if a route has recently changed and the new routing information is not yet available. Such a support for queuing packets is missing in such operating systems.

Gupta, Kawadia and Zhang proposed promising system services for on-demand and hybrid routing protocols [KZG02]. These are implemented for Linux in the ad-hoc support library (aslib) [KZG], which will be used for our prototype of PAMANET.

The basic idea of the aslib is based on the standard Linux IP stack. A default route is added to the routing table, which forward packets for which no route is yet discovered through a tunnel device to the ad-hoc support daemon (ASD) in user space. The ASD triggers the route discovery and queues the packets, until the route is discovered and added to the routing table. After route discovery the packet is re-injected back into the IP stack through a raw socket. To deal with the dynamics of topology changes in ad-hoc networks the ASD is also responsible for deleting entries from the routing table.

The main advantage of this approach is that the standard IP stack for packet delivery can be used for packets for which a routing entry exists. The benefit is fast delivery in kernel space if no route must be discovered. When no route to the destination is present the packet is queued until the user space routing daemon has added the route.

Most components of the PAMANET prototype are implemented in the ASD in user space to ease program development, to simplify debugging and to reduce changes to the Linux kernel.

3.2 Neighborhood Management

This component collects all necessary neighborhood information to maintain the PAMANET-routing, clustering and landmarking system. It consists of neighborhood discovery, a message box system delivering control data of PAMANET, and a graph metric that describes the link failure probability.

Neighborhood Discovery

The ad-hoc routing algorithms require a list of neighbor nodes for each node. This list consists of the MAC and respective IP addresses of all direct neighbors of a given node.

We approach this task by adding a new protocol that is responsible for finding the neighbors. The operation principle of the proposed protocol is similar to the ARP (Address Resolve Protocol). The differences are the following: (1) The ARP operates on demand. It is only triggered when the MAC address of a certain IP must be discovered. The protocol for the neighborhood list must be called periodically to detect new nodes and delete entries that have expired. (2) The ARP consists of sending a broadcast to all nodes, requesting the information of who has a certain IP. If the node exists, it is the only one who will reply the message. For the proposed protocol, instead of only one, all nodes that receive the request packet must update or refresh their lists.

To implement this new functionality, a new communication protocol is added to the kernel. We do not want to change the ARP to avoide additional efforts and to preserve its original functions.

The neighborhood list protocol proposed here does not require any replies. Every node will broadcast its data and all the others process the transmitted packet and extract the information from it. This solution is acceptable, since low level MAC mechanisms acknowledge the transmission and therefore it can be ensured that a certain node is able to send and receive signals to/from the other.

The list is implemented by registering a new protocol in the Linux kernel on top of Ethernet beside the IP protocol.

Message Box System

To build a route in a mobile MANET, on demand or pro-active, components in the routing layer of PAMANET protocol need to communicate with routing layer components of direct neighbors. There are three types of messages: two from the landmarking component and one from the consistent hashing component. None of them requires acknowledgment, and all tolerate the loss of (some) messages. Furthermore, the receiving order is also not important. One of the landmarking message types needs broadcast functionality.

Taking this into account and considering that messages could be larger than the maximum transmission unit (MTU), IPv6 protocol was chosen as a basis, i.e. all messages (with some internal header) will be packed directly into IP packets. One reason to choose this IP layer is, that there is no need to map IP and MAC addresses statically to each other. IPv6 messages can be used for the communication of the components involved in route discovery. This procedure can be used because the messages will be sent, in IPv6 packets, only to direct neighbors. For these neighbors a route is added when they appear in the radio transmission range of a node. If the landmarking or the consistent hashing component sends multi-hop messages, the same component will route the messages.

For the implementation under Linux, a new protocol on top of IPv6 is registered in the kernel. Because our own protocol is used, the whole TCP and UDP spectrum is untouched, none of the ports is allocated. Linux socket-API besides UDP and TCP sockets provides a mechanism to access raw IP packets. This feature is vitally important, because it avoids introducing changes into the kernel like implementing a new IP based protocol would.

The communication of this new protocol is done by a message box system. Its purpose is a simple and clean interface design for one-hop communication that can be easily used by all of our network components.

Graph Metric based on Link Failure Probabilities

Our key interest is to prefer stable edges for message paths and stable sub-components for clustering.

To keep the loss of packets at a low level in such a fast changing network as the PAMANET, we rate existing links and prefer more stable ones. As the only quality indicator we have the duration of the link, mostly because other indicators like the signal-to-noise-ration, the available bandwidth, position information etc. is not available or can only be made available with additional hardware or link testing. We rate links by a distance metric $w(e)$, where small values indicate good links and large values bad ones.

Since the duration of the links increases over its lifetime the distance (aka. weight) $w(e)$ varies and is described as a function over time t:

Definition 1 (Link Distance). *The function $w(e,t)$ defines the current link distance of a link e at time t and is defined as*

$$w(e,t) = \min\left\{1, \max\left\{p_f, \frac{c_w}{t - T(e)}\right\}\right\}$$

where t is the current system time, $T(e)$ is the time the link has been established, and p_f and c_w are parameter explained below.

In the implementation of PAMANET we have normalized this term by multiplying the distances by $\frac{2^{c'}}{p_f}$ for a small integer c'. Then the outcome is rounded such that link distances are described by integers.

Upcoming links receive heavy weights, keeping the associated nodes in a long distance, such that the network is unwilling to use such (virtually) remote nodes as intermediate nodes on routes. Within time

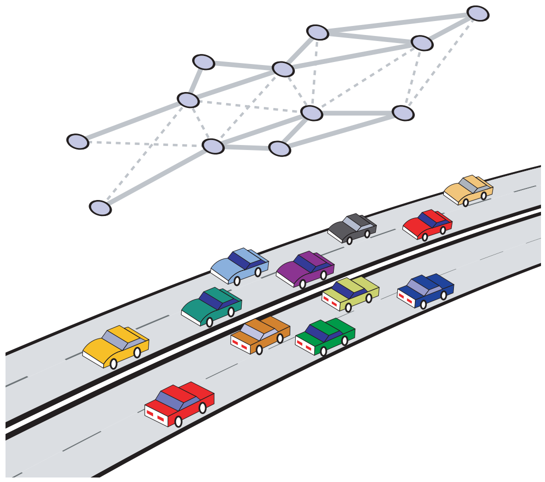
Figure 3: Scenario "Highway"

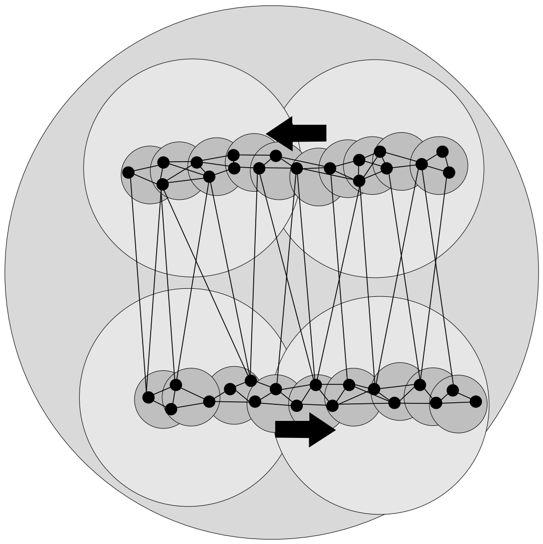
Figure 4: Clustering in the highway-scenario.

this distance reduces. More and more messages will be routed over this link. Eventually, this link is weighted with p_f which means it behaves like a wired connection.

The parameter p_f denotes a failing probability that holds for every link resulting from hardware or software errors. The parameter c_w will be chosen appropriately for the PAMANET-protocol based on empirical data.

This model leads to similar results as provided by empirical analysis [BNC00].

Assumption 1. *For some fixed $\tau > 0$ the edge distance $w(e, t)$ describes the conditional probability, that the link is available in the time interval $[t, t + \tau]$ under the assumptions it started in $T(e)$ and was available up to t.*

Under this assumption the length of a path according to this edge distances gives some information on the stability of routing. For a path $P = (u_1, \dots, u_m)$ let $w(P, t) := \sum_{i=1}^{m-1} w((u_i, u_{i+1}), t)$ the distance of the path.

Lemma 1. *Under assumption 1 and further assuming that the link failure probabilities are independent, if $w(P, t) \leq \frac{1}{2}$ then $w(P, t)$ is a constant factor approximation of the probability that P fails in the time period $[t, t + \tau]$.*

The proof will appear in the full paper. Since PAMANET optimizes message paths for the path weight $w(P, t)$ more stable routes weight are preferred.

Link distances and Mobility

One might argue that the change of link distance may cause unnecessary changes in the cluster-hierarchy. But note, that if a node is already connected to the network, then a fresh link does not change the cluster structure. Over time clusters may come closer and a smooth transition process takes place, e.g. if two large sets connect for the first time.

Beyond the smooth transition of the cluster structure the link distance concept helps to cope with mobile scenarios. We discuss the following scenario for its usefulness in PAMANET.

"The Highway Scenario"

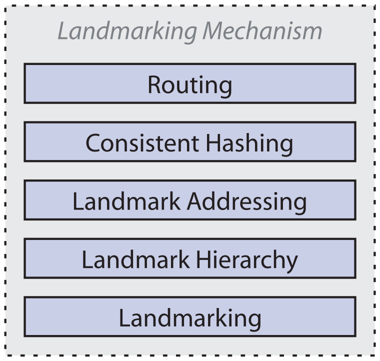

Figure 5: The separation of the landmarking mechanism.

Considering a highway with dense traffic, where cars in both directions are nodes of our PaMaNet. Same oriented cars move with small relative speed to each other, such that the corresponding links persist for a quite long time. Although the distance between the lanes with opposite driving direction is small, it does not make much sense to use such links very often, because each of them exists for only short time. Thus the average link availability t_S between two neighbored cars depends on their diverging speed.

The inverse relationship is now valid for the average length of the corresponding link distances. Based on this observation it is possible to embed a snapshot of this network into the Euclidean plane (allowing some small stretch factor of edges and considering only the shortest paths between nodes). Then the scenario of Fig. 3 reduces to a graph as depicted in Fig. 4. So, we have short edges in each lane, while the distance according to edge weight between cars of opposite driving direction is rather high.

3.3 Landmarking and Clustering

In its core PaMaNet's routing is based on the landmarking mechanism which was introduced by Tsuchiya [Tsu88] and a clustering algorithm. They provide the basis for position based routing and for consistent hashing. The landmarking and clustering component consists of the following three sub-components: landmarking, landmark hierarchy and landmark addressing.

Landmarking

On the top level of landmarking of PaMaNet only one node resides, which differs from the approach of Tsuchiya and follows the approach of the Hierarchical Layer Graph [SLRV03b]. This top level landmark ensures that all nodes of a level receive the landmark information of all sibling nodes, as we show below.

Two important measures for the landmarking mechanism are the publication and domination radius which are defined as follows:

Definition 2 (domination radius). *The domination radius of a node is the distance (in hops) where no other node with the same level may exist. It is defined by $r_{\beta,\ell} := \beta^{\ell-1}$ with $\beta \in \mathbb{R}, \beta > 1$ and ℓ is the level of the node.*

Definition 3 (publication radius). *The publication radius of a node is the distance (in hops) in which every node knows a route to that node. It is defined by $r_{\alpha,\beta,\ell} := \alpha \cdot r_{\beta,\ell}$ with $\alpha > \beta$.*

Distances relate to the graph metric induced by link distances (normalized by a factor of $2^c/p_f$ and rounded to receive integers).

For our PAMANET we use $\alpha = 6$. In fact, $\alpha = 2$ would be sufficient for all nodes to reach their parent nodes and thus would be sufficient for landmarking. But routing needs the information provided by the landmarking message within at least $\alpha = 4$ to prevent nodes of becoming a bottleneck when packets have to be routed from one side of the network to the other. Consistent hashing needs even $\alpha = 6$ which ensures that every node has knowledge about its sibling nodes.

Lemma 2. *In the publication radius of a landmark, messages can be routed on the shortest path (according to link distances) to the landmark generating node.*

Proof. This follows from the use of broadcasting messages in the publication sphere and storing the backward path information. □

Using this simple landmarking scheme we can formulate a key feature of this protocol. Link distances describe an Euclidean metric, if there is a mapping $f : V \to \mathbb{R}^d$ such that the shortest path from node u to v equals $||f(u), f(v)||_2$. Note that in [SLRV03b] it was shown that in the mobile scenario link distances can be embedded into 6-dimensional Euclidean space.

The following Lemma is analogous to Lemma 5 in [MSVG04]. We denote the set of nodes within the domination radius of a landmark as domination sphere, and analogously we use the term publication sphere for the set of nodes within the publication radius.

Lemma 3. *Assume that the link distances describe an Euclidean metric of d dimensions with minimum distance 1. For every node subset V' of V with domination radius $r_{\beta,\ell} = \beta^{\ell-1}$ and publication radius $\alpha r_{\beta,\ell}$ we observe the following, if node nodes of V' lie in domination spheres of other nodes of V'.*

1. *The number of nodes of V in each publication sphere is bounded by $(2\alpha\beta^{\ell-1} + 1)^d$, while the number of nodes in each domination sphere is bounded by $(2\beta^{\ell-1} + 1)^d$.*
2. *Every node of V is in at most S_d domination spheres, where $S_1 = 3$, $S_2 = 7$, $S_3 = 21$, and $S_d \leq 3^d$.*
3. *Every node of V is in at most $(2\alpha + \frac{1}{\beta^{\ell-1}})^d$ publication spheres.*

Note that since α is bounded by 6 each node only lies inside a constant number of publication and domination spheres (for each layer).

Landmarking is responsible for publishing nodes within the network. This is achieved by periodically "broadcasting" landmarking messages within the publication radius $r_{\alpha,\beta,\ell}$, except for the highest level node whose broadcasts are unlimited. This is necessary for the landmark addressing algorithm (see below). A landmarking message contains information about a node's level $\ell(N)$, a certificate $c(N)$, the distance $d(N)$ to that node, a time-to-live field TTL and the last node from which the message was received $\text{lhop}(N)$. Additionally, it contains information about cluster heads and sub nodes which are necessary for consistent hashing. On every hop the distance field will be increased and the time-to-live field will be decreased by the link's weight. The TTL ensures that only nodes within publication radius $r_{\alpha,\beta,\ell}$ will receive the message. To distinguish old and new information we are using certificates. If these certificates are not renewed within a certain time frame (depending on the distance in hops), all information about the corresponding node is deleted.

To build up the landmark hierarchy, we are using the information that a node gathers from all received hello messages . A hello message contains information about the distances to all of a node's cluster heads (one level up and higher), the levels of these cluster heads, their certificates, the node's own certificate, its level and all its landmark addresses.

Our approach to build and maintain the hierarchy works as follows: On reception of hello messages, a node processes all received and stored messages having a valid certificate. For each node in a hello message the minimum of all distances to this node, the highest stated level and its certificate will be stored. Then the node will reset its own level to zero and check whether all nodes in its set of nodes have a lower or the same level as itself and whether they are all in a distance of at least the domination radius

$r_{\beta,\ell}$. Now the level will be increased by one as long as the condition is fulfilled. At last the node tests whether it has any sibling nodes on the current level. If so, the decision which node establishes a new hierarchy level, is simply done by choosing the node having the lowest ID. In the end, the node compares the new computed level with its old level. If they differ the node publishes this change by sending out a new hello message, otherwise the node remains silent.

This way landmarks are assigned as cluster heads of sub-clusters. Note that the main objective of cluster heads is to provide landmark (routing) information for other nodes. Besides this, no further duties or priveleges arise for cluster head nodes.

Suppose there are two large disjoint networks A and B. Let one node from each network establish a new link between each other. Assuming that link's weight is larger than the distribution radii of A and B, no landmark message from one network would spread into the second one. To enable communication between A and B, it is necessary, that both top-level nodes get to know each other. Thus, landmarking of those nodes has no delimiting publication radius. Knowing each other, the top-level nodes adjusts to bridge-over the long (weighted) distance and communication between two arbitrary nodes in A and B is possible.

Landmark addressing is necessary for the routing mechanism (cp. [Tsu88]). We build our landmark addresses in the following way: At first we initialize the node's set of addresses with an empty set. For each landmark address of a node that we have received by a hello message we are taking a suffix from the current address, append our MAC address $(level + 1)$ times and add this new generated landmark address to our set of addresses. In the end we compare the set of new created addresses with our old set of landmark addresses. If they differ the new created set will represent our addresses.

Example: Suppose we are an level 1 node in a 4-level network and we have the MAC address X. Now we receive a landmark address of node Y that is $A.B.C.Y$. So we take the suffix $A.B.$ and append our own MAC address twice which leads to our new landmark address $A.B.X.X$.

Now the following Lemma ensures moderately sized address lengths for a network of n nodes.

Theorem 1. *If the link distances describe an Euclidean metric of d dimensions, then the length of a Landmark address is at most* $(\log n - \log p_f)3^d$.

For landmark routing we provide routing tables. The following theorem shows that the size of a routing table is logarithmic.

Theorem 2. *If the link distances describe an Euclidean metric of d dimensions, then the number of addresses to be stored in a node is at most* $(\log n - \log p_f)3^d$.

Note that these theorems describe worst case bounds. We expect that the actual address length and size has smaller constant factors.

Clustering

The clusters are implicitly given by the landmark hierarchy. A node u belongs to a cluster of level ℓ and representing v, if v is in a landmark address of the node at level ℓ. According to the hierarchy every node knows its ancestors and all child nodes. This leads to overlapping clusters on every level of the hierarchy.

We will exemplify the features of this clustering with the mobile scenario introduced above.

"The Highway Scenario"

Cars heading to the same direction will be collected in low-level clusters. Only on a macroscopic level cars in opposite direction will be found in a cluster. For the landmarking, this means that only a few number of cluster heads spread landmark information into the opposite direction. When a message needs to be sent to the opposite lane, then only once the link between the lanes will be used, since it is the most expensive link and routing optimizes the weighted distance of the path. This significantly increases network reliability and decreases the dynamics in the change of landmark addresses and routing tables, see Fig 4.

Landmark Address Routing

The choice of routes follows the approach of [Tsu88]. A message is routed in the direction of the nearest known landmark on the way to the destination node. To discover the next hop to such a landmark the routing mechanism has two alternatives. At first, the routing mechanism will set the forwarding rules in the routing table according to the information gathered by the landmarking messages. This is, if a node receives a message from node X through its neighbor node N it sets a forwarding rule in that way that node N is the next hop to X. If a lookup in this table fails the routing daemon will be called. It finds out the landmark addresses of the target node by calling the consistent hashing with that node's IP address. After this, the routing daemon searches the forwarding rules in the routing table for all nodes that appear in the landmark address(es) starting with the lowest level landmarks. If a match is found in the routing table the next hop to the destination is found.

The overall goal of this routing is to send packets along the most stable route. For each link we have assumed an independent failure probability which is described by the link distance. Recall that the sum of the link probabilities along a message path is a good approximation of the failure probability of the message path. Further note that if all links have the same age then the message path length is proportional to the hop-distance

Our goal is therefore to find message routes which are nearly as good as the shortest path (according to the link metric). Such a shortest path can only computed with offline with all link information (or using large routing tables of linear size.) We now show that our on-line algorithm approximates the off-line solution by a constant factor c. Such an approximation algorithm is called an c-competitive algorithm following the notation of [BEY98].

Theorem 3. *If a message is sent using a correct landmark address using the position based routing, then the chosen path is* 4*-competitive, i.e. the message path distance according to the dynamic link distance metric is at most a constant factor larger than the optimal choice.*

3.4 Weighted Consistent Hashing

For providing efficient landmark address lookup and as a data structure for the peer-to-peer data storage we implement weighted consisting hashing as follows.

Weighted Distributed Hash Table

As we have already mentioned consistent hashing is a key method in PAMANET for the distributed maintenance of data. We use it mainly for resolving landmark addresses from given IPv6-addresses. Besides this, we provide an interface for the application layer to make this distributed data structure available for the implementation of an efficient peer-to-peer database.

The weighted distributed hash-table is hierarchically organized using the cluster described above. Given as input the data, the keyword, and the cluster, described by the clusterhead and the level, a node is uniquely determined as follows. First all sub-clusters are mapped into a space using the IPv6-address of the cluster-head as key. Then the keyword is mapped into the same space. Then, the closest cluster-head to the mapped keyword describes which sub-cluster needs to be chosen. This realizes the consistent hashing as described in [KLL$^+$97].

Due to the hierarchical structure within the PAMANET and the underlying link distances, nodes (representing clusters) are not equal, as it is in [KLL$^+$97]. Some nodes are cluster of a large number of sub-nodes, while others are only responsible for a few sub-nodes. Therefore, we use a weighed consistent hash function that equally balances the load according to the relative size of the clusters. Our association of hash values to hash locations works with a concept that is similarly to [BSS02], yet improves on the balancing properties and can be evaluated more efficiently by this distributed network. It will be described in a subsequent publication.

We need a pseudo-random uniform deterministic function mapping the keyword to the . Following the standard approach we make use of the common cryptographic MD5 function [Riv92]. MD5 computes to any given bytestream, i. e. key, a hash value with a size of 16 bytes, which we use as hash range.

Hash locations, i. e. nodes, have to have a unique, common known identifier which is given by the MAC-address. Hence, we use MAC-addresses as input for MD5 to compute the position of the hash locations within the hash range rather than IPv6-addresses.

We provide data integrity if hash locations disappear due to mobility or other reasons in the following ways.

First, several copies of the data are stored within a cluster. If one node leaves the cluster, there are others, that still have the data.

Second, if a node leaves a cluster but not the network, this nodes receives a new landmark addresses and thus recognizes this changed situation. If it has data, it is no longer responsible for, it sends them to the now responsible node and deletes it from its local memory.

Third, for both described procedures we cannot guarantee the preservation of the data, it is inevitable to periodically re-hash the data to the network. This is done by the node with the original data.

Having a network with many thousand nodes spread over a large area, it is profitable to have more copies of the hashed data close-by than far away. Therefore, hash data is copied λ-times for each cluster containing the emitting node and stored at several nodes using λ generic keywords.

In addition, we provide the possibility of limiting the distribution of hash data to a specified range. The range gives the maximum level of a cluster, where the data may be stored in.

Hashing new data to the network is done in two steps. First, the emitting node creates as much copies as needed for the whole network. Each copy for the same level needs to have a different hash value, therefore the key is concatenated with an integer number between 1 and λ before processing the MD5 hash. This integer is the *hashoffset* of the copy, i. e. the copy number within a level. Second, each copy is sent to its destination node.

The route a data packet takes through the network is analogical to a landmarking route. Each node on the route will calculate the destination and, if unknown, send it to the node known as the closest to the destination. If the destination is reached, the data is saved by the node. Data is always stored within the smallest clusters, i. e. on level 0. Since each node of arbitrary level is also a level 0 node, all nodes have the same expected amount of storage.

The advantage of this procedure is, that even if a data packet is already on its way while the target changes, this can be recognized before the original target is reached. Then, the packet is automatically redirected, where the new destination is likely to be near to the old one and the packet can be delivered quickly.

Note that the following information has to be sent with the data to enable recalculation of the hash locations in other nodes:

1. Key and hashoffset (copy number for a level) to compute the hash value.
2. The locality value to limit the spread range.
3. The cluster head corresponding to the locality value, because for a node within more than one cluster, it is essential to know the cluster the data came from and thus it has to be redirected to.

If a node stored some hash data and moves to a different cluster, it may be not longer responsible to keep the data. The node simply calls the algorithm for sending data for each stored data. This will redistribute it automatically.

Key Based Routing

Using the consistent hashing component, we provide key based routing as an interface to high-level applications, enabling them to make use of a balanced distribution between all nodes of the PAMANET. For example, this is used by our database to compute the lookup clients (see 3.6).

If an application gives a keyword and some data to key based routing, it forwards this data to the destination node(s), which are determined by consistent hashing. At the destination node, data and keyword are delivered to a corresponding receiving application.

Since each lookup peer can check locally whether its responsibilities has changed. We feed the lookup node with sufficient data for such a local check by delivering *hashoffset*, the number of copies λ and the *cluster head* besides the keyword. A special interface is given to the application for using this information. If a reassignment of data at a lookup nodes occurs (due to network changes) the lookup node is informed by this procedure. Then, the application layer of the lookup node can trigger the update information before a regular update from the data source is possible. For arriving information, then in case of a different responsibility, the data is rerouted to its new destination, anyhow. See also 3.4.

This implies that if a node moves within the network, it informs the high level application, that recalculation by key based routing may be necessary. The execution of data checking has to be done by the application, though. In either way, this leaves the control of data to the corresponding application.

3.5 PAMANET Routing

PAMANET-routing combines the above described components. If an up-to-date landmark address of the destination is known then messages are routed directly with the 4-competitive position based routing scheme.

If not, key-based routing is used to look up the landmark address of a given IPv6-address. For the lookup of a d distant node, key-based routing needs at most a query of length $4d$ according to the graph metric based on link distances. If some λ copies are available at each level then on the average the lookup is accelerated by this factor λ, since the consistent hashing protocols give all necessary information to find thee nearest holder of the information.

If all edges used in the network have approximately the same age, then the number of messages necessary for the lookup grows linearly with the hop-distance between source and destination.

For the proactive part, each node needs to store at most $O(\log n)$ landmark entries of size $O(\log n)$ (which multiplies with the size C of the unique identifiers and optionally with the number λ of additional copies). So, every node uses at most $O(C\lambda \log^2 n)$ memory with high probability.

For the proactive communication protocol, every link regularly sends also $O(\log^2 n)$ landmark signals. Most of the traffic, however is induced by consistent hashing. This can lead to serious problems if two networks of same size are connected over only one link. We think that such worst-case instances rarely occur and even if, there is no work around for this network, since also the message routing suffers from these communication bottlenecks.

If key-based routing is used to look up the address, a route discovery packet will be created, containing all information about the landmark addresses of the original destination. This packet will be send towards the destination before any other packets to prevent other nodes from using the key-based routing again. This lowers the overall traffic in our network.

```
pamanetrouting(destination, packet) {
    if (! forwardingTable.contains(destination.ip)) {
        destination.lms = keyBasedRouting(destination.ip)
        updateForwardingTable(destination)
        sendRouteDiscoveryPacket(destination)
    }
    sendPacket(packet)
}
```

Listing 1: PAMANET-Routing

3.6 A Peer-to-Peer XML Database with Distributed Egoistic Caching

Caching is the method of choice to reduce communication traffic in networks. As a first assumption we consider documents that remain unchanged during their lifetime. Of course documents can appear and disappear over time, and even cached copies can prolong this lifetime. However, if a document change is necessary, then a new document (using a version system) needs to be introduced and the old document is deleted. Furthermore, information is provided at its originating peer, called the *data origin peer*. This peer is responsible for the appearance (and the disappearance) of the document. This implies that if parts of the network disconnect, then peers in the disconnected parts fail to have access to this information.

We use the notion of egoistic caching, that (except referential information for finding caches) peers cache only information they have queried before. The peers store this information for the egoistic purpose of avoiding the latency another query for the same data. These peers are called *cache peers*.

For further traffic reduction the information content of an XML document is partitioned into so-called segments. Based on [BT04] these segments divide the contents into logical meaningful sub-parts, while the union of all sub-segment can resemble the original document. As a data structure to determine which segments are needed to answer a query, each cache peer additionally stores the *ColoredSchemaGraph*. So, database queries in PAMANET address only segments of documents, which reduces computational time and memory usage of the peers as well as network traffic significantly.

For this, each query is split according to these data segments and forwarded to some close cache peer or data origin peer holding the requested partial information. Then, the answers are combined at the request peer for answering the original query. After this operation, the partial information will be cached on this query peer for internal use as well as for providing a new cache peer for close neighbors. This extends the approach of [BT04], where proximity information of peers was not available.

Logical subset testers supporting caching strategies for mobile environments have been proposed before, cf. [NS03] and [BS03]. These testers, however, do not provide egoistic behavior, implying that peers along the communication route provide high computational resources for third-party evaluation. This is not the case in PAMANET.

In the following, we describe how the database localisation takes advantage of the weighted consistent hashing layer of PAMANET.

Database Localization Strategies

Each data origin peer and cache peer is responsible for its availability to close neighbors. For this, we assign so-called *lookup peers* which serve as a pointer to the closest peer holding the queried information of a document's segment. This is done be the *refresh*-operation, which is based on the key based routing. Every cache or data origin peers routes referential information (i.e. the IPv6-address) to each landmark cluster it is assigned to. The search key for key based routing is hereby described by the document name and the segment name. Then key based routing delivers this message to a peer that will serve as a lookup peer for rerouting queries to the cache peer. We describe the refresh-operation in more detail below.

Segmentation

To provide an effective caching mechanism and faster query evaluation, PAMANET uses the finite segmentation concept introduced in [BT04] and extends it to the use of our mobile ad-hoc network.

This caching concept is based on a segmentation of the underlying XML data of a database into disjoint patterns. Let D be a *Document Type Definition* [BPSM$^+$04] and X an XML document of the database. The algorithm given in [BT04] uses the ColoredSchemaGraph to create functions $d_1, \ldots, d_n$ (in terms of XPath expressions) which extract certain parts of X. If we apply these functions $d_1, \ldots, d_n$ to the data X, we get the *data segments*: $d_1(X), \ldots, d_n(X)$. These $d_1, \ldots, d_n$ are representatives of the data segments and are called *data segment functions*. These data segment functions describe a partition: $\forall i, j \in \{1, \ldots, n\}, i \neq j : d_i(X) \cap d_j(X) = \emptyset$ and $\bigcup_{i=1,\ldots,n} d_i(X) = X$. Here, we define the intersection of data segments $d_i(X)$ based on predefined data nodes and define the union of data segments $d_i(X)$ based on XML join operations for predefined ID nodes.

A peer evaluates a query Q using thw ColoredSchemaGraph. It identifies the data segment functions $\{q_1, \ldots, q_k\} \subseteq \{d_1, \ldots, d_n\}$ whose corresponding data segments $q_1(X), \ldots, q_k(X)$ are necessary to answer the query locally. More formally we have:

$$Q(X) = Q\left(\bigcup_{i \in \{1,\ldots,k\}} q_i(X)\right)$$

This avoids querying and delivering the whole XML-document X across the network. For the answer of Q only specific queries on a partial set of data segments is $q_i(X)$ needed.

Note that data lookup peers provide information about some close peers holding the specific segments of X, such that for one lookup the query is performed by a parallel search optimizing the route for each segment.

The Query-and-Cache Operation

Peers may (and probably will) serve as data origin peers, cache peers, and lookup peers at the same time.

Cached segments and original XML-documents are stored in a local database which resides in the application layer. This describes the main amount of information stored on each peer, describing the egoistic part of memory usage. Lookup information for segments are also stored in a local database. This altruistically stored amount of information is relatively small compared to the size of the referenced segment, since it only contains which PAMANET-peer (IPv6-address) stores a segment (addressed by ID) in its local database.

For the atomic query for a segment, a peer sequentially increases the search range until it finds a lookup peer, referring to a cache peer or the data origin peer. The search range is implemented using the levels of the landmark hierarchies. In each of the landmark clusters, key based routing forwards queries to the same peer which serves as lookup peer. These lookup peers of each level answer in parallel whether the queried segment is stored within their landmark clusters. If all lookup nodes fail to point to some peer, then the level is increased and this search iterated.

Because of the overlapping of the landmark clusters in each level, it is ensured that a cache peer in distance d can be determined in distance $O(d)$ (according to link metric). Then, the querying peer will contact the so-called *r*esponse peer, which is either the data origin peer or a cache peer holding the wanted segment.

After receiving new segments each peer will serve as a cache peer for these segments. If such a cache peer runs out of cache memory, segments are replaced by an least-recently-used-replacement strategy. This initiates a refresh-operation to uphold the data structure.

Since data sources are distributed and the network is dynamic, available data sources change over time. For this the same distributed query operation (albeit without caching) is used with a special key that routes requests to lookup-peers maintaining the knowledge about all available data sources in the PAMANET.

Coping with Dynamics

Network changes and node dynamics cause permanent changes in the infrastructure e.g. caused by diverging of peers distances between query and response peer vary. A network change may lead to the following situations. Responsibility of lookup nodes change. Cache peers and data origin peers may become unavailable or eviction on the cache peers may take place. Then, lookup nodes carry out-dated references.

Therefore, we periodically perform refresh operations triggered by the data origin or cache peers. This operation is analogously to the insert operation, while an adaptive longest in system strategy is used to invalidate reference information in lookup nodes.

4 Outlook

Consistent Hashing and Network Bottlenecks The underlying layers of landmarking schemes need only small proactive communication. The largest proactive traffic is induced by consistent hashing, which provides the look-up service for finding moving nodes. There are several strategies to reduce this traffic. One could by the that if only low-level landmark information is changed that this information is not transported to larger distances. For routing a message, then local lookups need to be performed to fill up the outdated local landmark information.

Another approach is to bound the maximum traffic of consistent hashing of each level to a certain percentage of the communication bandwidth of link. If more is acquired than a random deletion process will reduce this flow. To compensate this cut-back on information flow mechanisms to deal with old landmark information must be applied.

PAMANET goes Internet Note that the PAMANET routing and lookup service does not rely on specifically features of wireless communication. Therefore it makes sense to discuss an application of PAMANET to standard networks. Then, we would face a self-configuring Internet without designated routers and built-in look-up service. However, this approach does not reflect the heterogeneity of communication networks and especially the problem of different bandwidths and latencies between node. It is an open question whether the link distance metric can be adjusted such that the communication bandwidth also becomes an optimization objective.

Peer-to-peer networks and distributed database PAMANET provides a peer-to-peer-network and not the full functionality of a database. Further research will show how PAMANET can be extended to a full featured distributed database.

State of the art As of this paper's writing PAMANET is not fully implemented. Experimental studies will be conducted with the implementation of the prototype in the fall of 2004. To prove scalability of PAMANET, the main issue of this project, a large scale communication network at least 100 mobile ad-hoc nodes will be established starting at winter 2004/2005. These experiments will answer the question how large the network need to be such that the asymptotic proved in this paper outperform mobile ad-hoc-networks studied so far.

References

[AKJP$^+$02] Shurug Al-Khalifa, H. V. Jagadish, Jignesh M. Patel, Yuqing Wu, Nick Koudas, and Divesh Srivastava. Structural joins: A primitive for efficient XML query pattern matching. In *Proceedings of the 18th International Conference on Data Engineering*, 2002.

[ASV01] Serge Abiteboul, Luc Segoufin, and Victor Vianu. Representing and querying XML with incomplete information. In *Proceedings of the Twentieth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 2001.

[BEY98] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, 1998.

[BMJ$^+$98] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols, October 1998. In Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98).

[BNC00] Jeff Boleng, William Navidi, and Tracy Camp. Metrics to enable adaptive protocols for mobile ad hoc networks, 2000.

[BPSM+04] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, Franois Yergeau, and John Cowan. Extensible Markup Language (XML) 1.1. W3C Recommendation, `http://www.w3.org/TR/xml11/` (last visit 29.06.2004), 2004.

[BS03] Stefan Böttcher and Rita Steinmetz. *Testing Containment of XPath Expressions in order to Reduce the Data Transfer to Mobile Clients*. ADBIS. 2003.

[BSS02] A. Brinkmann, K. Salzwedel, and C. Scheideler. Compact, adaptive placement schemes for non-uniform distribution requirements. In *Proceedings of the 14th Annual ACM Symposium on Parallel ALgorithms and Architectures (SPAA-02)*, pages 53–62, New York, August 10–13 2002. ACM Press.

[BT04] Stefan Böttcher and Adelhard Türling. Finite segmentation for XML caching. In *IFIP TC8 Working Conference on Mobile Information Systems (MOBIS). Oslo, Norway*, 2004.

[CJL+03] T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A.Qayyum, and L.Viennot. Optimized link state routing protocol, October 2003. IETF Internet Draft version 7.

[CL00] T. Cortes and J. Labarta. A case for heterogeneous disk arrays, 2000. 1st IEEE International Conference on Cluster Computing (Cluster2000).

[ET03] Rainer Erkens and Ulf Troppens. *Speichernetze, Grundlagen und Einsatz von Fibre Channel SAN, NAS, iSCSI und InfiniBand*. dpunkt.verlag GmbH, 2003.

[GKP03] Georg Gottlob, Christoph Koch, and Reinhard Pichlera. The complexity of xpath query evaluation. In *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 2003.

[Haa97] Z. J. Haas. A new routing protocol for the reconfigurable wireless networks, October 1997. IEEE ICUPC'97, San Diego, CA.

[HKRZ02] K. Hildrum, J. D. Kubiatowicz, S. Rao, and B. Y. Zhao. Distribted object location in a dynamic network. In *Proceedings of the 14th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA-02)*, pages 41–52, New York, August 2002. ACM Press.

[JL99] Y. T. Mingliang Jiang and Jinyang Li. Cluster based routing protocol, July 1999.

[JM96] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks, 1996. Mobile Computing, T. Imielinski and H. Korth, Eds., Kluwer, pp. 183 - 97.

[KLL+97] David Karger, Eric Lehman, Tom Leighton, Mathhew Levine, Daniel Lewin, and Rina Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *ACM Symposium on Theory of Computing*, pages 654 – 663, may 1997.

[KLLS99] David Karger, Tom Leighton, Daniel Lewin, and Alex Sherman. Web caching with consistent hashing. In *Proc. of 8th International World-Wide Web Conference*, Toronto, Canada, 1999.

[KZG] V. Kawadia, Y. Zhang, and B. Gupta. http://aslib.sourceforge.net/.

[KZG02] V. Kawadia, Y. Zhang, and B. Gupta. System services for implementing ad hoc routing protocols, 2002.

[MCC03] Alessandro Di Marco, Giovanni Chiola, and Giuseppe Ciaccio. Using a gigabit ethernet cluster as a distributed disk array with multiple fault tolerance, 2003. 28th Annual IEEE International Conference on Local Computer Networks.

[MSVG04] Friedhelm Meyer auf der Heide, Christian Schindelhauer, Klaus Volbert, and Matthias Gruenewald. Congestion, dilation, and energy in radio networks. *MST: Mathematical Systems Theory*, 37, 2004.

[NS03] Frank Neven and Thomas Schwentick. XPath Containment in the Presence of Disjunction, DTDs, and Variables. In Diego Calvanese, Maurizio Lenzerini, and Rajeev Motwani, editors, *9th International Conference on Database Theory (ICDT)*, pages 315–329, Siena, Italy, 2003. Springer.

[PB94] C. E. Perknis and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers, October 1994. Comp. Commun. Rev., pp 234-44.

[Per01] C. E. Perkins, editor. *Ad Hoc Networking*. Addision-Wesley, 2001.

[PR99] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing, February 1999. Proc. 2nd IEEE Wksp. Mobile Comp. Sys. and Apps., pp 90-100.

[Räc02] Harald Räcke. Minimizing congestion in general networks. In *Proceedings of the 43rd Symposium on Foundations of Computer Science (FOCS-02)*, pages 43–52, Los Alamitos, November 16–19 2002. IEEE COMPUTER SOCIETY.

[RFH$^+$01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Computer Communication Review*, volume 31, pages 161–172. Dept. of Elec. Eng. and Comp. Sci., University of California, Berkeley, 2001.

[Riv92] Ronald L. Rivest. The md5 message-digest algorithm, April 1992. RFC-1321, MIT LCS and RSA Data Security, Inc.

[SLRV03a] Christian Schindelhauer, Tamás Lukovszki, Stefan Rührup, and Klaus Volbert. Worst case mobility in ad hoc networks. In *Proceedings of the 15th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA-03)*, pages 230–239, New York, June 7–9 2003. ACM Press.

[SLRV03b] Christian Schindelhauer, Tamás Lukovszki, Stefan Rührup, and Klaus Volbert. Worst case mobility in ad hoc networks. In *Proceedings of the 15th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA-03)*, pages 230–239, New York, June 7–9 2003. ACM Press.

[SMK$^+$01] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. In Roch Guerin, editor, *Proceedings of the ACM SIGCOMM 2001 Conference (SIGCOMM-01)*, volume 31, 4 of *Computer Communication Review*, pages 149–160, New York, August 2001. ACM Press.

[Tan96] Andrew S. Tanenbaum. *Computer Networks 3rd edition*. Prentice Hall, 1996.

[Tsu88] Paul F. Tsuchiya. The landmark hierarchy: a new hierarchy for routing in very large networks. *ACM SIGCOMM Computer Communication Review*, 18(4):35–42, August 1988.

[XHG03] Kaixin Xu, Xiaoyan Hong, and Mario Gerla. Landmark routing in ad hoc networks with mobile backbones, 2003.

[YS02] P. Yolum and M. Singh. Flexible caching in peer-to-peer information systems. In *AOIS '02, Agent-Oriented Information Systems, Proceedings of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems*, 2002.