1 Introduction

Over the past several years, the area of peer-to-peer (P2P) networks has attracted large amounts of research. P2P networks are decentralized application-layer (i.e. overlay) networks. They are formed dynamically to create a virtual network on top of an underlying physical network such as the Internet. In contrast to the classic client-server paradigm, P2P networks do not have a centralized server infrastructure that clients can turn to when looking for specific information. Instead, in pure P2P networks, all participating nodes (i.e. the peers) are considered equal in terms of their function in the network. The core idea is that each peer functions as a client and a server simultaneously: a peer will usually query the network for specific information, forward other peers' requests, and provide information for other participating peers. P2P networks represent a very appealing and promising technology mainly for the following reasons.

In a classic client-server network architecture, the conventional way to accommodate an ever growing number of clients is to provide additional and/or more powerful servers, whereas the resources of the clients (e.g. their computational capabilities) remain largely untapped. In peer-to-peer networks, however, each additional node is not only a new source for network traffic, but it also contributes its resources such as computing power, bandwidth, etc. to the network. Therefore, P2P networks have the potential to scale more naturally with an increasing network size.

In a client-server architecture, the server represents a single point-of-failure. When the server goes down, all clients are cut off from all information. P2P networks, on the other hand, are largely fault-tolerant. When a peer fails and becomes unavailable, other peers will automatically take over the responsibilities of that failed peer. Orthogonal issues such as replication set aside, only the information shared by the failed peer will be gone – all other information will still be available to the remaining peers in the network.

Furthermore, peer-to-peer networks are self-configuring. There is no need for a central administration. New nodes simply connect to some other existing nodes and instantaneously become part of the network, assuming their role and their share of the network load. Thus, P2P networks can be formed dynamically and spontaneously without the need for carefully creating an infrastructure or/and acquiring dedicated server hardware first. In fact, this capability for "organic growth" – i.e. the ability to start a network with minimal upfront investment and/or effort that can potentially grow into an arbitrarily large network – is a key characteristic and strength of P2P networks.

Peer-to-peer networks are also interesting from a more social point-of-view as they provide a higher degree of anonymity compared to centralized server-based

systems. Since there are no central servers, the users' behavior and preferences cannot as easily be traced. Each node in the network will usually only see a small part of the overall network traffic. Certain information can also be located virtually anywhere in the network as each node provides its own information autonomously. Theoretically speaking, this would make it quite difficult for an (outside) entity to impose any kind of censorship on the network or to even try and shut down the network entirely – although current file-sharing systems often suffer from what is referred to as "pollution" (the massive introduction of deliberately non-functional content – often offered as professional service by specialized companies – to severely decrease the practical usability of the network). At this point, it also has to mentioned, of course, that these characteristics have been abused in the past for massive copyright violations in the context of file-sharing application. This, however, does not change the intrinsic appeal and value of these characteristics as such.

Due to their distributed nature, P2P networks can and have been used for a wide variety of Internet-based applications. First-generation P2P networks have been largely used for popular file-sharing applications (e.g. [17, 26]). Those first-generation P2P networks are also referred to as *unstructured* P2P networks since they usually do not impose any structure on their topology. Instead, nodes connect to each other largely at random. Because of this lack of topological structure, requests are broadcast through the entire network to locate the desired information. It is clear to see that those broadcasts can create huge amounts of network traffic as the number of nodes and requests increases, which can potentially use up all available network bandwidth. Of course, this overhead can be reduced in practice by restricting the request broadcasts to a certain network radius. However, this can easily lead to situations where the desired objects cannot be found as they might reside outside the radius of the request. In other words, there is a trade-off between scalability and recall in unstructured P2P networks.

To overcome the scalability issues of the unstructured, first-generation P2P networks, *structured* P2P networks have been proposed more recently. Very popular representatives of structured P2P networks are the so called *Distributed Hash Tables* (DHTs) [45, 50, 56, 73]. In fact, DHTs have become largely synonymous with structured P2P networks. DHTs impose a certain structure on the topology of the overlay network in order to introduce an upper bound on the effort to locate a specific piece of information in the network. More specifically, DHTs provide the powerful primitive of key-based routing: For any given key, they can very efficiently locate among all participating peers the peer currently responsible for the key. Instead of flooding such requests, DHTs can route them within a certain number of overlay hops to the responsible peer. DHTs have been used efficiently as building blocks for a large array of decentralized, distributed network applications such as data storage systems [10, 49], distributed email systems [34], event notification systems [51, 74], and distributed name services [2, 6, 8, 43, 61] to mention but a few.

At the same time, another field that has attracted large amounts of research are *Mobile Ad Hoc Networks* (MANETs). MANETs consist of wireless mobile devices (e.g. PDAs, notebooks, sensor nodes, etc.) that dynamically form a network

among themselves. Like P2P networks, MANETs have no fixed infrastructure and centralized administration but are, instead, self-configuring as well. Nodes will usually send out their own requests, forward other nodes' requests, and respond to other nodes' requests during their participation in the network. P2P networks and MANETs also share a high degree of dynamicity as nodes can join and leave the network at any given time.

Due to their flexibility and self-organization capabilities, MANETs are well suited for scenarios where certain network services such as message routing, event notification, etc. have to be provided quickly and dynamically without any centralized infrastructure. Thus, MANETs can, for example, be used to enable Ubiquitous Computing by providing the dynamic and self-configuring networks needed in such environments. Another frequently cited example is disaster recovery. In such a scenario, there is usually no time to painstakingly set up a fixed centralized infrastructure. Instead, network services need to be established quite literally in an ad hoc fashion.

With the continuing proliferation of ever more powerful wireless mobile devices, it is, therefore, becoming more and more interesting to build the above mentioned complex distributed network applications that one is accustomed to from the Internet on top of MANETs.

1.1 Problem Statement

As observed above, MANETs and P2P networks share a good number of key characteristics, such as the lack of a central infrastructure, a highly dynamic network topology, and the need for self-organization. Hence, when designing distributed network applications for MANETs, it would be intuitive to consider the building blocks that have proven themselves appropriate in P2P systems. However, conventional DHTs are ill-suited for a simple deployment on top of MANETs for the following three reasons:

1. First of all, it is important to realize that overlay traffic as such does not exist physically. What does exist, though, is the physical traffic *incurred* by the overlay network. Furthermore, as previously mentioned, DHTs were designed as application-level overlay networks for the Internet. By abstracting away the underlying physical network, standard DHTs generally do not consider the physical topology in the construction of their overlay topology. In other words, by no means do two overlay neighbor nodes also have to be physical neighbors. This usually leads to the situation that overlay hops can incur unnecessarily long physical routes. Figure 1.1 shows an example where four overlay hops actually traverse the physical network twice. Although a number of approaches have been proposed recently (e.g. [46, 66, 60]) to alleviate this problem, standard DHTs are not primarily concerned with physical locality. While this might be tolerable on the wired Internet with its high bandwidth, it is obviously not feasible for MANETs. Here, the delivery probability of a packet quickly decreases with each additional physical hop due to factors such as low bandwidth, low computation power, packet collisions, or transmission errors.

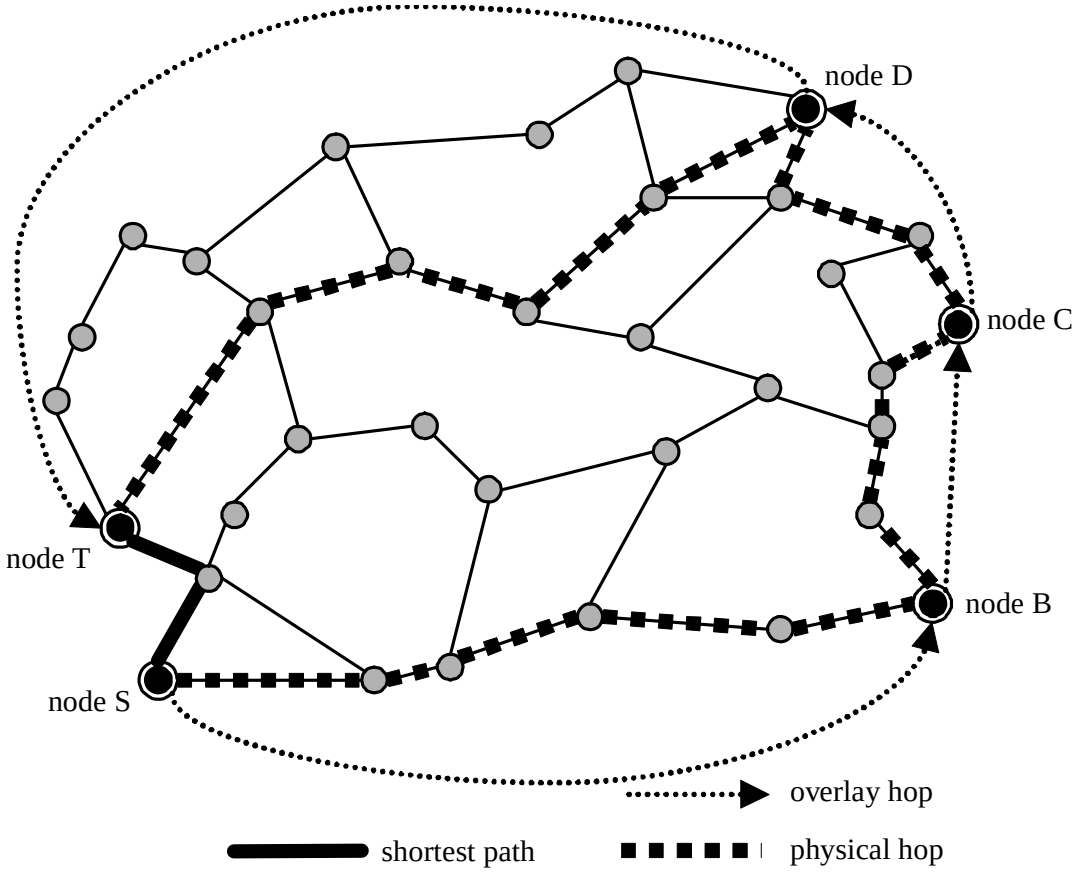

Figure 1.1 Overlay vs. physical routing.

2. As nodes move around incessantly, routes in MANETs are usually quite volatile and break quickly. For this reason, ad hoc routing protocols have to (re-)establish routes frequently. Due to the lack of a central infrastructure, the majority of ad hoc routing protocols have to at one point or another resort to flooding the network – or regions thereof. This, of course, renders the overlay routing superfluous. There is no point in maintaining an application-level DHT when the physical route to carry out an overlay hop has to be (frequently re-) established through broadcasting. In that case, one would have been better off broadcasting the key lookup itself in the first place. In fact, it is easy to imagine a situation where a key lookup requires two overlay hops, both of which have to have their physical routes discovered through broadcasting. In that case, the key lookup would cause the network to be flooded twice, which is clearly suboptimal.

3. In order to guarantee routing convergence and consistency, DHTs have to periodically maintain their routing tables. Depending on the size and structure of a DHT's routing table and the lookup traffic pattern, the maintenance traffic can constitute a significant portion of the overall traffic. Given the limited bandwidth in MANETs, conventional DHT maintenance can be prohibitively heavy-weight and overwhelm the MANET.

Therefore, in order to provide the powerful primitive of key-based routing, it clearly does not suffice to merely deploy a conventional Internet-based DHT on top of a MANET. Instead, when designing structured P2P overlays for MANETs, it is essential to explicitly take the characteristics of MANETs into consideration.

1.2 Contributions

This thesis presents the design, performance evaluations and applications of a DHT substrate explicitly designed for the use in MANETs. *Mobile Ad Hoc Pastry* (MADPastry) combines conventional ad hoc routing and P2P overlay routing at the network layer to provide DHT functionality for MANETs. The main contributions of MADPastry are as follows.

Key-based routing for MANETs. The main concept of a DHT is to provide key-based routing. DHTs route a packet based on a key to that node that is currently responsible for the key– i.e. whose own ID is closest to the packet's key among all live nodes in the network. By enabling indirect routing for MANETs, MADPastry is an ideal building block for the previously mentioned DHT-based distributed network applications that one knows from the Internet. Since MADPastry provides the DHT functionality that those distributed network application rely on, they can be ported for the deployment in MANETs in a straight-forward manner.

Locality awareness. As discussed in Section 1.1, it is vital for any DHT substrate for MANETs to carefully consider physical locality. MADPastry uses *Random Landmarking* [63, 64, 65, 66, 72] to map the physical topology to its overlay topology. MADPastry constructs clusters of physically close nodes that share a common overlay ID prefix. Therefore, physically close nodes in MADPastry are also quite likely to be close to each other in the overlay ID space.

Consideration of physical routes in the overlay routing process. One of the most costly tasks in terms of network traffic in MANETs is route discovery. To avoid this whenever possible, MADPastry might deviate from optimal overlay routing if the physical route of an overlay hop is unknown. Instead, MADPastry might choose a less optimal (in the sense of the overlay ID space) next overlay hop whose physical route is known, thereby favoring low physical traffic over optimal overlay routing.

Extensive exploitation of packet information. Another very costly task in MANETs is routing table maintenance – both ad hoc and overlay. Therefore, MADPastry augments its packet headers with both overlay and ad hoc routing information about the current node. This way, any node overhearing a packet can update its routing tables on-the-fly without having to engage in explicit routing table maintenance. This significantly limits the overhead induced maintenance.

Practicality for applications. MADPastry has been used as a building block for both a common MANET application (unicasting) as well as for a more complex distributed name service for MANETs, thereby demonstrating its versatile applicability.

The above mentioned characteristics make MADPastry a light-weight, robust and scalable DHT substrate for MANETs.

1.3 Thesis Overview

The remainder of this thesis is organized as follows.

Chapter 2 presents the background for this thesis. It describes in detail the different techniques used in both unstructured and structured peer-to-peer overlay networks as well as their respective advantages and shortcomings. Additionally, it discussed the numerous existing approaches to ad hoc routing.

Chapter 3 discusses related approaches to MADPastry including work on both unstructured and structured P2P overlays for MANETs as well as a wide variety of service discovery protocols for MANETs. Again, the respective advantages and disadvantages are thoroughly evaluated.

Chapter 4 presents MADPastry's architecture in detail. All architectural aspects such as the structure of MADPastry's routing tables, the routing strategies, the role that overlay clusters assume or how MADPastry performs routing table maintenance are described.

Chapter 5 thoroughly evaluates the performance of MADPastry through simulation experiments. Numerous different network parameters and settings such as network size, node velocity, or the impact of various request rates on the overall performance are considered. To put MADPastry's performance into perspective, MADPastry is always compared against a simple broadcast-based approach as well as a Pastry-based DHT substrate without explicit consideration of physical locality.

Chapter 6 presents a concrete application of MADPastry. It demonstrates how MADPastry can be used to build an efficient DHT-based name service for MANETs in order to discover arbitrary resources.

Chapter 7 presents another possible application of MADPastry. It discusses how MADPastry can be used to not only provide indirect, key-based routing in MADPastry but also for direct unicasting (i.e. for sending a packet from a given source to a given destination) as well.

Finally, Chapter 8 concludes this thesis and provides an outlook on future work around the convergence of peer-to-peer overlay networks and mobile ad hoc networks.

Chapter 9 and 10 provide an appendix with the list of the abbreviations used throughout this thesis as well as the reference list.