Network Architectures
and Services
NET 2009-03-1

PhD Dissertation

Title

CSP, Cooperative Service Provisioning using Peer-to-Peer Principles

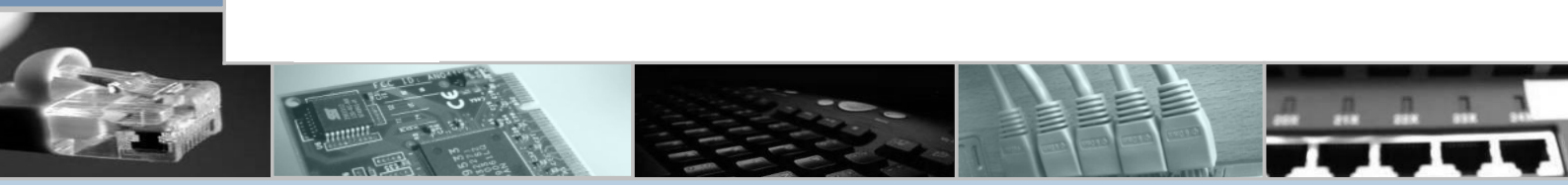
Author

Michael Kleis

Organisation

Chair for Network Architectures and Services
Department of Computer Science, Technische Universität München

Technische Universität München



Cataloging-in-Publication Data

Michael Kleis
CSP, Cooperative Service Provisioning using Peer-to-Peer Principles
PhD Dissertation
March 2009
Chair for Network Architectures and Services
Technische Universität München

ISBN: 3-937201-05-X

ISSN: 1862-7803 (print)
ISSN: 1862-7811 (electronic)
Network Architectures and Services NET 2009-03-1
Series Editor: Georg Carle, Technische Universität München, Germany

Institut für Informatik
der Technischen Universität München

CSP, Cooperative Service Provisioning using Peer-to-Peer Principles

Michael Kleis

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr. J. Schlichter

Prüfer der Dissertation:
1. Univ.-Prof. Dr. G. Carle
2. Prof. E.W. Biersack,
EURECOM, Sophia Antipolis / Frankreich

Die Dissertation wurde am 24.10.2008 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 09.03.2009 angenommen.

Abstract

There are two main challenges for flexible service provisioning on top of today's Internet. First the end-to-end networking paradigm which is based on a service and data unaware transport, and secondly a lack of high level communication abstractions.

With this thesis we approach both challenges. To address the first point we propose a methodology to integrate the Network Provider into the process of service provisioning. The introduced cooperative service provisioning (CSP) enables an interaction between Service Provider, Network Provider and Client. CSP can be applied in classical Provider as well as novel Peer-to-Peer scenarios.

To address the second point, we adopt a Service Overlay approach. A new method based on the extension of Distributed Hash Tables (DHTs) is proposed to realise a distributed control plane for a decentral and reactive creation of Service Overlays. To archive this, a methodology to reduce Service Overlay related routing problems to search problems was developed. With regard to the required Quality of Service (QoS) measurements, the thesis focuses on scalable and time efficient estimation techniques. As a result, two new delay prediction schemes have been developed which improve estimation accuracy when compared to the state of the art.

Furthermore, a novel approach to bandwidth estimation is investigated. It combines methods from landmark based delay estimation with a transformation step and has favourable estimation accuracy.

A CSP prototype has been realised based on the Peer-to-Peer middleware software OMNIX [KKKD03]. For the implemented Content-Addressable Network (CAN) DHT a new algorithm to repair the CAN structure in case of multiple not gracefully leaving nodes has been integrated.

Zusammenfassung

Zwei Kernprobleme bei Internet basierten Dienstleistungen sind ein von Daten und Dienstanforderungen unabhängiger Netzwerktransport sowie fehlende Kommunikationsabstraktionen.

Um einen daten- und dienstgerechten Transport zu ermöglichen, wird in dieser Arbeit ein Ansatz zur Integration von Netzwerk-Providern in den Dienstleistungsvorgang beschrieben. Das resultierende kooperative Dienstleistungsprinzip (CSP) ist anwendbar auf klassische Provider sowie neuartige Peer-to-Peer Dienstleistungsszenarien.

Zur Bereitstellung von Kommunikationsabstraktionen bei CSP wird in der Arbeit ein neuer Ansatz zur Verkettung von verteilten Dienstkomponenten vorgeschlagen. Grundlage bildet dabei eine verteilte Hash-Tabelle, die zu einer Plattform für den verteilten und reaktiven Aufbau von dienstspezifischen Overlay-Netzwerken erweitert wird. Als Voraussetzung für diesen Ansatz werden die im Overlay-Netzwerk-Fall auftretenden Routing-Probleme auf Such-Probleme reduziert. Für die notwendige Erfassung von Netzwerkeigenschaften wie Verzögerung und Durchsatz werden Schätzverfahren untersucht. Als Ergebnis stellt die Arbeit zwei neue Verfahren zur Abschätzung von Netzwerkverzögerung, sowie ein neues Verfahren zur Abschätzung von Netzwerkdurchsatz vor. Eine prototypische Realisierung des beschriebenen CSP Systems wird in Zusammenhang mit einem Multimedia Transcoding Szenario evaluiert.

Acknowledgements

The ideas behind the approach presented in this thesis have been developed together with my supervisor Prof. Dr. Georg Carle. First of all I would like to thank him for his steady encouragement and support. I would further like to express my gratitude to Prof. Dr. Ernst W. Biersack for being the second supervisor and for his valuable comments and suggestions. The main part of research and development work has been performed inside the project "Situated and Autonomous Service Control (SASCO)" founded by France Telecom R&D Orange Labs. I am grateful to Dr. Mikaël Salaün for initiating and leading the project. Several meetings and discussion with the project partners from the University of Tübingen and France Telecom R&D Orange Labs helped to improve the presented results. I would also like to thank my competence-center leader at Fraunhofer FOKUS Dr. Tanja Zseby. Without her support this thesis would not have been possible. Further acknowledgements go to Dr. Dorgham Sisalem and Dr. Mikhail Smirnov for many helpful discussions as well as to Ahmed Elmokashfi and Wang Fei for writing their master- respectively diploma theses in the context of this dissertation. I also want to thank my colleagues at Fraunhofer FOKUS for their friendship and good times. I further would like to express my deep appreciation and gratitude to Angelika Pauli for her valuable support and patience.

Berlin, April 26, 2009

Michael Kleis

Contents

List of Figures

List of Tables

List of Abbreviations

AS	Autonomous System
BBS	Big Bang Simulation
BF	Breadth First
CAN	Content-Addressable Network
CBMP	Constraint-based Media Processing Problem
CBRP	Constraint-based Routing Problem
CC/PP	Composite Capabilities/Preferences Profile
CDF	Cumulative Distribution Function
CNLS	Closest Neighbour Loss Significance
CSP	Cooperative Service Provisioning
DAG	Directed Acyclic Graph
DF	Depth First
DHT	Distributed Hash Tables
DRE	Directional Relative Error
GCP	Geometric Cluster Placement
GNP	Global Network Positioning
GPS	Global Positioning System
GT-ITM	Georgia Tech Internetwork Topology
IETF	Internet Engineering Task Force
IN	Intelligent Network
IP	Internet Protocol
LCBMP	Least Cost Constraint-based Media
MC	Media Client
MDS	Multi-dimensional Scaling
MP	Media Processor

MPEG	Moving Pictures Expert Group
MS	Media Server
MTSA	Multimedia Transport Service Agreement
NP	Network Provider
NS	Name Server
OSPF	Open Shortest Path First
P2P	Peer-to-Peer
PIC	Practical Internet Coordinates
PM	Processing Module
QN	Query Node
QoS	Quality of Service
RQI	Range Query Initiator
RRL	Relative Rank Loss
RTT	Round Trip Time
SBN	Service Bootstrap Node
SDP	Service Description Protocol
SG	Service Graph
SID	Service Identifier
SLA	Service-level Agreements
UDP	User Datagram Protocol
VCC	VideoLAN Client Controller
VID	Virtual Identifier
VLC	VideoLAN Client
WSDL	Web Service Description Language
XML	Extensible Markup Language

Chapter 1

Introduction

A central challenge of Internet service provisioning is to incorporate an increasing number of wireless and wired network technologies, a variety of heterogeneous end user terminals and the requirements of QoS-sensitive and realtime multimedia services. A scenario expected in near future is a group of mobile users equipped with heterogeneous terminals, connected via heterogeneous wireless access technologies while using the same Quality of Service (QoS) sensitive multimedia service [amb08], [dai08]. To optimise the service for its customers, a Service Provider needs to adapt the multimedia service to the corresponding end user terminals as well as to the transport and error characteristics of the used network technologies. This includes: QoS-aware media transport, error correction and off-line or on-demand transcoding/downscaling of audio/video data.

1.1 The Challenge of Network and Terminal Aware Service Provisioning

Addressing such a functionality at *Network Layer* is a challenging task. Current Internet provides in general unicast, best-effort point-to-point communication while more general communication abstractions are missing. In fact, Internet Engineering Task Force (IETF) standards and implementations exist for e.g. multicast [WVK+01], anycast [PMM93] or end-to-end QoS [Wro97],[BBC+98] but for political and/or economical reasons these features are not available at Internet scale.

As a consequence professional, Internet based, Multimedia Delivery is addressed at *Application Layer*. However, the classical Client/Server-based media delivery principle is expected to be too static for the described scenario. The main shortcomings are congestion-related as well as scalability and single-point-of-failure problems. Further, a high complexity of the server application can be expected in case media processing functionalities have to be integrated into the server. To overcome these limitations, the Content Distribution Network (CDN) principle has been developed. Mainly driven by industry, CDNs are realised in the form of proprietary Overlay Networks

optimised for request routing, media caching and replication as well as the distribution of multimedia data to a large audience. Companies as e.g. Akamai [aka08] maintain CDN systems with more than 20.000 servers distributed in 71 countries. The resulting management and configuration complexity of such systems is considered as the main disadvantage of the CDN approach. To establish a platform supporting multimedia services for mobile users in 3G and 4G networks, the IP Multimedia Subsystem (IMS) was developed inside the third generation partnership project (3GPP). The IMS can be described as a SIP-based signalling overlay including Authorisation, Authentication and Accounting (AAA) functions and support for multimedia calls and services. To enable the flexible introduction of new IMS based services a SIP-Application server is included into the standard. The resulting IMS architecture is complex while lacking self-organisation and self-repair features. In addition the configuration and management overhead of an IMS is expected to be high.

1.2 A Cooperative Service Provisioning Principle

The problem addressed in this thesis is the exploration of a service provisioning platform for QoS aware data transport and processing with low management and configuration complexity. Because of their stringent QoS requirements, multimedia services are used to evaluate the system. The two central obstacles identified for realising such a platform on top of today's Internet are:

1. An end-to-end networking paradigm [SRC84] based on a service and data unaware transport.

2. The lack of *high level* communication abstractions [SAZ+02].

To address the first point, we will investigate a method to integrate the Network Provider into the process of service provisioning. As the expected result, service provisioning can be addressed by using a *cooperative service*

provisioning principle relying on interaction between Service Provider, Network Provider and the Client.

To address the second point we adopt an *Overlay-based Service Composition* approach, allowing to combine and reuse existing service components for building new (composed) services.

1.2.1 Integrate the Network Provider into Service Provisioning

In the past, two traditional business relationships with regard to service provisioning have dominated:

1. A direct business relation between clients and Service Providers in networks based on the end-to-end principle as in the Internet case.

2. A business relation between the client and an operator or Network Provider as in networks based on the Intelligent Network (IN) principle [MPZ96].

For INs, new services have to be introduced either by the Network Provider itself or by a third party provider using defined interfaces (e.g. Parlay/X) offered for this purpose. Since a Network Provider has full control over its network infrastructure, a QoS-aware transport of the data can be realised from the service source to the service sink. However, the main drawbacks of intelligent network architectures are their complexity, the administrative overhead as well as the difficulty of adding new services.

In this thesis we propose to combine both approaches while targeting at integrating the Network Provider into the process of service provisioning. After such an integration, the actual service provisioning can be addressed cooperatively by interaction between the Service Provider, the Network Provider and the Client. As an expected benefit, all entities involved in the transport of data related to a service are also aware of service specific information. As a further effect, a Network Provider can be part of the value chain by providing *value added transport* to third party Service Providers as well as its clients. To summarise, the proposed approach will allow:

- The Service Provider to concentrate on Service/Content and to abstract from transport or end user terminal-related issues

- The Network Provider to offer value added transport service

- The Client to access services that are optimised for his/her enduser terminal as well as the used access network technology

The resulting entity model we also use during this thesis is shown in figure 1.1. It contains the three entities Service Provider, Network Provider and Client. As shown in figure 1.1 there are three interfaces for inter entity

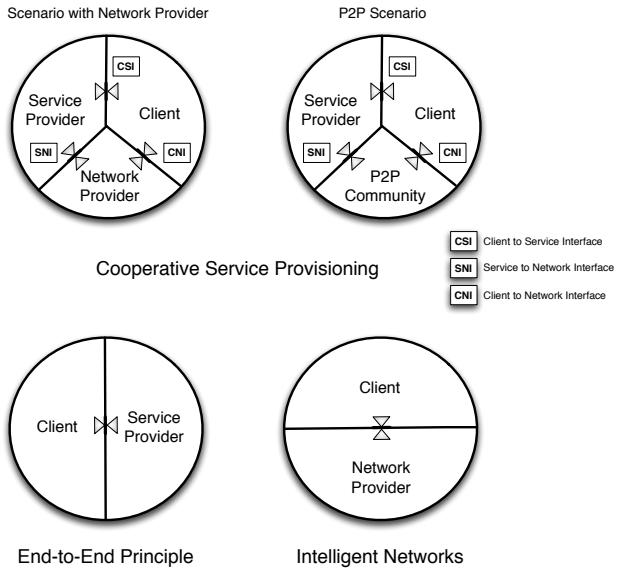


Figure 1.1: CSP Entity Model

communication which are a *Client to Network Interface* (CNI) for Network Provider to Client communication and vice versa as well as a *Client to Service Interface* (CSI) and a *Service to Network Interface* (SNI) for Service provider to Client and Service Provider to Network Provider communication respectively.

1.2.2 Overlay-based Service Composition

To realise communication abstractions without the requirement of significant changes at Network Layer, Overlay Network principles have gained attention. Since 2000, one can observe a rising interest in overlay-related research. Since then many classical network problems as QoS, resilience, multicast or security have been addressed using the overlay approach. In addition, one branch of overlay-related research started to study the possibility of using the overlay concept for the flexible, on-demand composition of services. This thesis contributes to this branch of overlay research by investigating the question how principles from the area of Peer-to-Peer (P2P) networks can be used to establish an on-demand service composition platform.

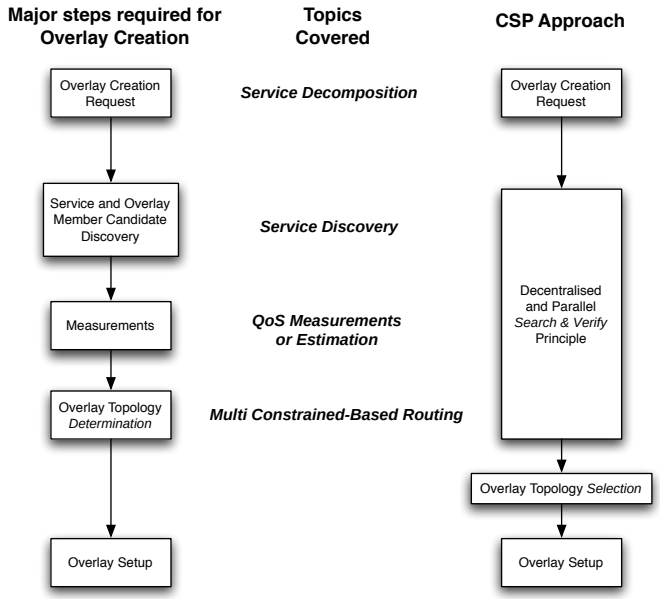


Figure 1.2: Main Steps in Service Overlay Creation

In contrast to other work done so far in this area we address classical P2P as well as a provider based scenarios with a strong emphasis on a cooperation aspect. As a central concept it is studied how the principle of Distributed Hash Tables (DHTs) can be extended to realise a distributed control plane for the setup of Service Overlays. As the anticipated main benefit, the resulting system can be realised in a fully distributed fashion and inherits the

self-organisation and self-repair capabilities of DHTs. As illustrated in figure 1.2, a further consequence of the DHT-based approach is the fact that we can address service discovery, QoS-measurements and routing aspects at the DHT layer. The prerequisite of our DHT based approach is a methodology to reduce constraint based routing problems to related search problems. The development of such a methodology is one subject addressed in this thesis. In addition we discuss and evaluate different search principles. As a further topic, we address the question how QoS constraints can be verified during overlay creation in a scalable and time efficient manner by using QoS estimation techniques.

1.3 Research Questions

Summarising above considerations, the research questions addressed in this thesis are the following:

How can a Network Provider be integrated into the process of service provisioning?

This question points to service registration, service level agreement, service decomposition and service request processing schemes in conformance with client, service- and network- provider requirements. In chapter four we provide a top down description of the Cooperative Service Provisioning proposed in this thesis.

How can principles from the area of Peer-to-Peer (P2P) be applied to establish an on-demand service composition platform?

The process of Service Overlay construction includes service decomposition, service discovery as well as routing and QoS specific issues. All this topics cover architectural as well as algorithmic aspects related to a control plane. The proposed architectural concept are introduced and described in chapter four. To address the algorithmic aspects we focused on the following two overlay related research questions:

How to reduce service composition to a search problem?

The prerequisite of the DHT based approach used in this thesis is a

methodology to reduce constraint based routing problems to related search problems. This question addresses the necessary requirements, prerequisites and steps of this process. In chapter five we describe the proposed approach.

How to select overlay members with respect to QoS constraints of services?

This questions addresses measurement and predicting of QoS related information. In chapter six the pros and cons of reactive and proactive QoS verification are discussed. In the same chapter two new delay prediction schemes are introduced. It is further described how the accuracy of landmark based distance estimation can be increased by adequate landmark selection and clustering. In the same chapter we show how landmark based techniques can be applied to bottleneck bandwidth estimation as well.

1.4 Methodology

The methodology used to address the research questions described above has been based on the classical *hypothesis testing* method. In Figure 1.3 the steps of the performed research work are mapped to the corresponding chapters of the thesis. To evaluate the performance and to study the general characteristics of the developed CSP concept, a simulation based approach has been selected. The main reason for this decision was the fact that simulations allow to analyse specific properties of an algorithm without interference. It is further possible to study the behaviour of different parts of the developed system in large scale. For the required network simulation a packet-based, discrete event simulator has been developed similar to the one described in [CJK+03]. We have used two network data sets throughout this thesis: The first one is the publicly available P2PSim [p2p08] king data set which represents Internet delay measurements, and the second one is based on a synthetic topology generated by the GT-ITM [ZCB96b] topology generator. The P2PSim king data set has been investigated comprehensively to ensure its suitability for representing the real internet. For the performed experiments we used a GT-ITM generated topology whenever it was required to

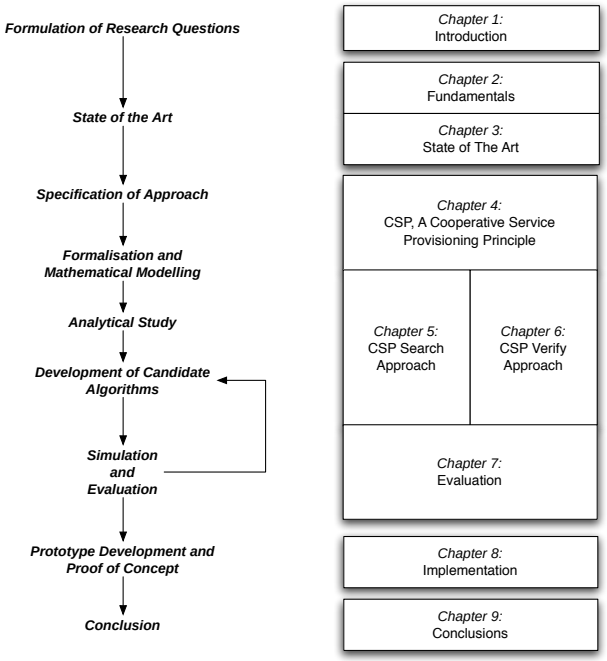


Figure 1.3: Methodology and Document Structure

consider delay and bottleneck bandwidth as metrics. Delay and bottleneck bandwidth values have been calculated based on the GT-ITM generated link weights simulating shortest and widest path routing respectively.

1.5 Document Structure

After this introduction, the remainder of the thesis is organised as follows: In **chapter two** we collect and present fundamental concepts used as a background in this thesis. That is, the chapter does not describe any own results but contains the fundamental models, principles and algorithms used throughout the document. In **chapter three** follows a survey of the related state of the art. We provide a classification and comparison of relevant work including a feature/property table. The chapter is completed with a collection of selected mathematical foundations of the network embedding approach. **Chapter four** will introduce CSP as a method for overlay-based service provisioning using P2P principles and provide a description of the

main tasks to be addressed with CSP for Service Overlay creation. We describe CSP at Service Level, by introducing the notion of Service Graphs and show the different steps required from Service Request till Service Delivery. After this, we focus on CSP at network level and provide a graph theoretic formalisation of the addressed Constraint-Based Media Processing Problem (CBMP). The chapter concludes with a complexity analysis of the proposed CSP approach. Now reaching **chapter five** we focus on the question how to reduce the CBMP Problem to a search problem. We show how fundamental search principles can be realised utilising an extended DHT. The required DHT extensions are introduced together with a CSP specific Range Query principle for Content Addressable Networks (CANs). A CAN has been selected because this DHT allows to realise the required service specific indexing in an elegant way. The main focus of **chapter six** is on the impact of estimation techniques to the CSP verify operations. Starting with delay estimation, we introduce Geometric Cluster Placement (GCP) developed during the thesis and compare it with other state of the art network distance estimation schemes. Based on this study we derive Netforecast, a scheme combining (1) clustering, (2) a triangluation heuristic first introduced by Hotz [Hot04] and (3) GCP to optimise the results with regard to long-range and short-range distance estimation. In a next step, we discuss if and how the developed schemes can also be used for bottleneck bandwidth estimation. **Chapter seven** is dedicated to the evaluation of CSP. We first perform simulations to evaluate the properties of different search principles. In the remainder of the chapter we analyse the impact of QoS estimation to accuracy of the proposed CSP approach. In **chapter eight** we describe a CSP prototype implementation. The core task required for the implementation has been interfacing with and extension of a P2P middleware called OMNIX [KKKD03] used for the realisation of CSP DHT functions. We also describe a new algorithm for the repair function of a Content-Addressable Network in case of multiple not gracefully leaving nodes. With **chapter nine** the thesis is competed with conclusions and a collection of possible future work items.

Chapter 2

Fundamentals

In this chapter we provide the fundamentals required to follow the argumentation in this thesis. This includes an introduction to overlay networking and P2P concepts as Distributed Hash Tables. In addition we describe the data used for the experiments performed throughout the thesis.

2.1 Introduction to the Overlay Network Principle

A common view is that the success of the Internet, as a network based on the end-to-end principle [SRC84] was and is driven by the creativity of its users. In fact, one benefit of the end-to-end approach is that the only precondition to act as a service provider is to have a computer and internet connectivity combined with a static IP or a globally unique and resolvable name. However, since service-related signalling is performed on an end-to-end basis, the actual transport of service data is accomplished by a network not aware of service demands. Thus *non-elastic* realtime multimedia data is treated in the same way as *elastic* world wide web traffic.

For a service-specific transport, it is required to have information about the service a transmitted packet belongs to. In case of multimedia services, an example for service related information is the type of the MPEG frame an actual IP packet belongs to. Having such knowledge it is possible to decide in a critical situation if a single packet can be dropped (e.g. if it belongs to a B-Frame) or has to be protected (e.g. if it belongs to an I-Frame).

To realise a service-specific transport without the need of significant changes at the Network Layer, overlay network principles have gained attention [SA05]. Overlay Networks are build in general by end-systems, and can be considered as networks defined over another set of networks. In fact, the principle of overlays is very common in networking and the most famous Overlay Network is the IP-based internet itself. More concrete, each IP network can be considered as an overlay for which the set of layer two components defines the corresponding underlay network. Examples for other well known Overlay Networks are the Multicast Backbone (MBONE) or P2P

Networks as Gnutella [gnu08], Kazaa [kaz08] or EDonkey [emu08]. Recent and past research in the area of Overlay Networks mainly considers virtual networks composed of end-systems interconnected using virtual network links as realised by IP-to-IP tunnels [Tou01]. The union of all virtual links and participating end-systems is denoted as the Overlay Topology. In general a single link in the Overlay Topology corresponds to a path in the underlay network (c.f. figure 2.1). To utilise an Overlay Network for communication, most

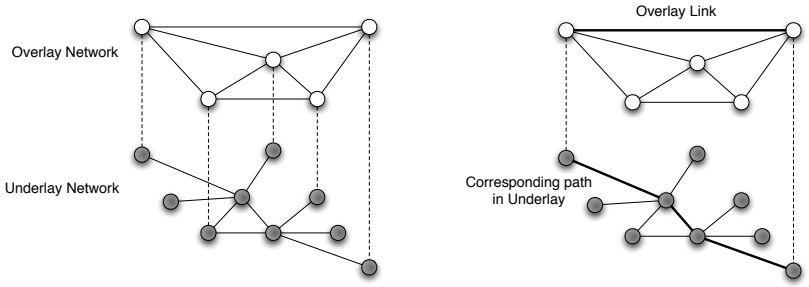


Figure 2.1: Overlay Principle

overlay approaches implement their own overlay-related addressing schemes and routing principles. Based on this characterisation we can provide a first definition for an Overlay Network which is:

Definition 1 (Overlay Network)**.** *An Overlay Network is defined to be a virtual network of nodes and logical links, build on top of an existing underlay network.*

Due to the fact that overlay topologies are defined by end systems one of the central research questions in the overlay context is: *How do end systems with limited topological information cooperate to construct good overlay structures? [CRZ00], [RHKS02], [HWJ05].* This question is also of central importance for CSP and will be addressed in chapter 6.

2.2 Service Provisioning Using Overlay Networks

The Web and GRID communities intensively discussed and researched service-oriented computing and architecture concepts. The resulting standardised frameworks and protocols as e.g. the Web Service Description Language (WSDL) [wsd08] already allow a flexible modelling of web applications and services. Furthermore it is possible to combine and reuse existing service components to build new *composed services* in the Web and GRID domain [NB05]. Using similar ideas, the Overlay community is working towards a service composition principle [GN02a], [GNCW03], [JN04], [XN02], [Kel04], [amb08], [GNY04]. One of the most important differences to the work in the Web and GRID area is that the overlay community puts a much stronger focus on incorporating QoS constraints, as e.g. the requirements of realtime multimedia, into the service composition process. To illustrate this, in figure 2.2 a multimedia-based service composed of two processing steps is used as an example. This service has to be realised by interconnecting a multimedia

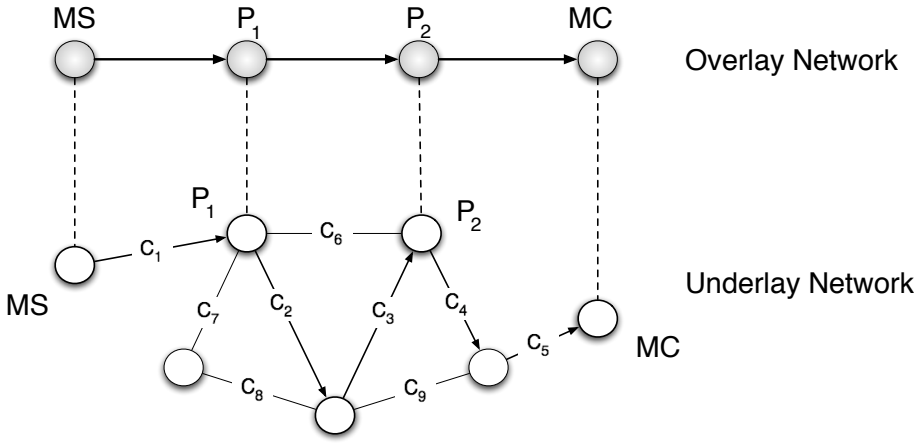


Figure 2.2: Service Composition Principle

media server (MS), two media processors (P_1, P_2) and a multimedia media client (MC) using an Overlay principle. Since multimedia services are QoS-sensitive it is required to ensure that service related constraints for bandwidth

and delay are not violated by processing as well as transport related tasks. In this context the link weights c_i for $i = 1, \ldots, 9$ in the underlay denote vectors for the delay and bandwidth properties of the corresponding link. In case of considering more than one metric overlay based Service Composition has a strong relation to (multi-) constraint-based routing problems [RS00], [OS00], [KK01] we will discuss in more detail in chapter 4.

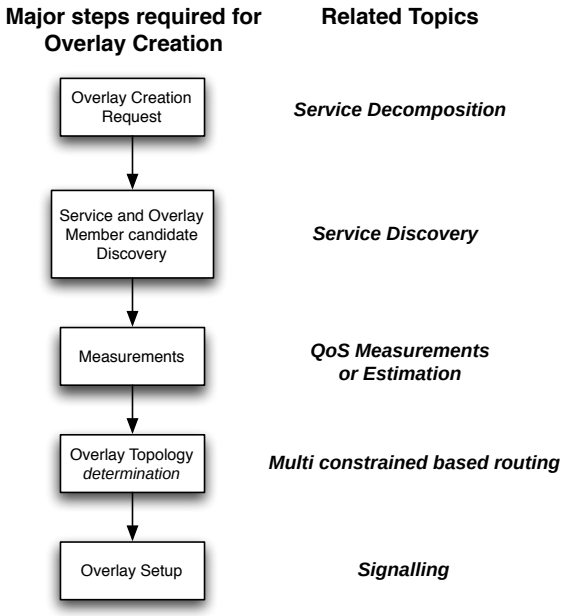


Figure 2.3: Major Steps in Overlay Creation

In figure 2.3 we have summarised the major steps that have to be addressed for overlay-based Service Composition. Each system addressing an on demand creation of Service Overlays has to cover these steps either explicitly or implicitly. In this thesis we propose to combine the service discovery, QoS-verification and Overlay Topology determination steps using a *Search-and-Verify* principle. Since the proposed approach is centred around the concept of Distributed Hash Tables, we will introduce the required fundamentals about DHTs in the next section.

2.3 Distributed Hash Tables

There are two main categories of P2P networks, namely *structured* and *unstructured* ones. While in a structured P2P network the topological properties of the overlay in combination with an addressing scheme are used to establish a platform with provable communication characteristics, unstructured P2P networks rely on statistical properties of their Overlay Topologies. Unstructured P2P networks organise peers in overlay networks with scale-free or random graph based topologies. Such overlay topologies are known to have a diameter growing logarithmic with regard to the number of nodes (also called "Small World" property). Consequently search queries can be distributed among peers using a simple message flooding algorithm with a relatively low time to live (TTL). In case of Gnutella [Rip01] a common setting for the TTL is 7 hops.

The most popular models for structured P2P networks implement a Distributed Hash Table. In a DHT the hash buckets are represented by one or more P2P-nodes (to store the data). The hash keys are either provided with, or derived from the data. After a node has joined a DHT-based P2P network, the hash of a key corresponding to a special data item is used to identify the next hop towards the P2P node where the data item is stored. The benefit of DHT-based systems is that they can provide guarantees for the success of a search query (if the search item is in the system), as well as bounds for the scope of the search (e.g. $O(\log N)$, where N is the number of nodes). This last fact allows DHT based systems to efficiently locate rare items. As basic functions, a DHT in general provides:

DHT Function	Description
Join	Function which can be used to become a member of the DHT
Publish	Function used to store a pointer to data
Search	Function to locate before published pointers
Fetch	Function to download data from an arbitrary member of the DHT
Leave	Function used to leave the DHT overlay remaining its structure intact
Repair	Function used to repair the DHT overlay structure in case of node failure

Table 2.1: Basic DHT Functions

2.3.1 Content-Addressable Networks

A Content-Addressable Network (CAN) [RFH+01] is a realisation of the DHT principle. In its most basic form a CAN can be described as a geometric P2P overlay structure defined in a d-dimensional co-ordinate space e.g. a d-dimensional torus defined by the cube $[0,1]^d \subset \mathbb{R}^d$. A CAN supports the basic hash table operations described above for key value pairs (K, V). During the CAN join procedure, each node obtains an overlay address in the form of a d-dimensional co-ordinate. In a two dimensional CAN as e.g. shown in figure 2.4, the CAN address is of the form $(x, y) \in \mathbb{R}^2$. Based on the defined overlay addresses, the CAN space is dynamically partitioned between all the nodes in a way that each node has its own distinct territory. This territory represents the chunk of the DHT a node is responsible for (c.f. figure 2.4). In

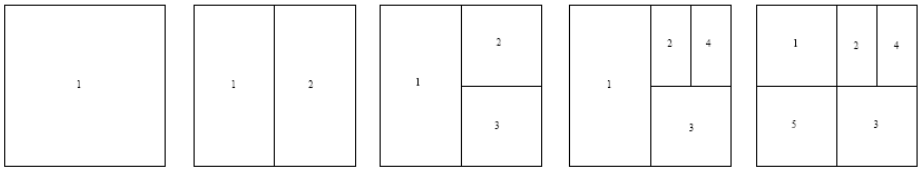


Figure 2.4: Two dimensional Content Addressable Network

addition, each node stores information about the nodes responsible for the zones in its direct neighbourhood. In the two-dimensional case, two CAN nodes are neighbours if their corresponding zones have a common border in the form of a line but not in form of a point. In the situation of figure 2.4 for example the neighbours of node 2 are the nodes 1,3,4 but not node 5. CAN requests like publish or search are routed by intermediate nodes using a *greedy* or *compass routing* [KSU99] principle. Using figure 2.5 as an example, the search function in a two-dimensional CAN is realised as follows: When a node receives a query for value V it first calculates $hash_{CAN}(V) = (x, y)$, where the function $hash_{CAN}$ is a globally known hash function. After this calculation, it sends a *search* V message to the CAN overlay address (x, y) using e.g. a greedy routing principle. This *search* V message arrives finally at the CAN node owning the territory where (x, y) belongs to. This node

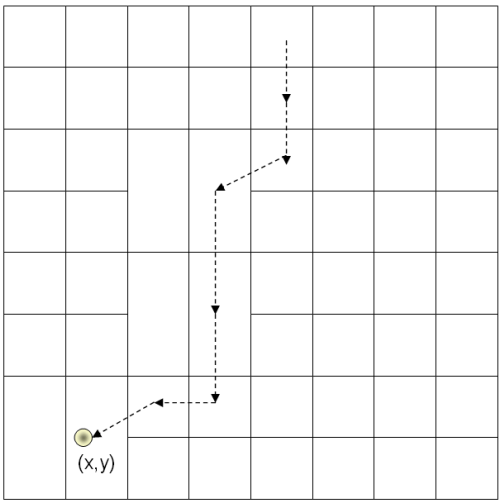


Figure 2.5: Routing in a Content Addressable Network

checks if it stores information about the underlay address of a node where V can be fetched from. If yes, the corresponding transport address is returned, if not an error message is send.

2.4 QoS Measurement and Estimation

The verification of a set of QoS constraints can be considered as one of the core tasks in the described Service Overlay scenarios. In this section we will provide fundamentals for the on demand verification of QoS constraints. Besides classical active or passive QoS measurements, the estimation of QoS parameters as e.g. delay has evolved in the past years to a research area in its own right. One of the main anticipated reasons for this is the scalability of estimation approaches as e.g. *landmark-based distance estimation*. For the CSP approach we will study the impact of QoS estimation techniques to complexity and accuracy of overlay creation. The impact of QoS estimation to the CSP verify operation will be described in chapter 6 while in the following chapter we outline a state of the art.

2.4.1 Active and Passive Measurements

Active measurements inject test traffic into the network in order to measure network characteristics. Traffic patterns and execution times can be adapted to specific measurement objectives. Active measurements allow the performance of controllable experiments and are well suited for monitoring the network situation, fault detection etc. Nevertheless, they are not appropriate for traffic-related applications like accounting and Service Level Agreement (SLA) validation. Usage-based accounting unquestionably has to be based on the existing traffic in the network. SLA validation is possible with active measurements to a certain degree, but there are several disadvantages. First of all test traffic always induces additional load on network links and routers. This can cause further congestion problems in times when network load is high. The additional traffic interferes with the customer traffic and therefore can influence the transmission quality and measurement results. A second problem is the generation of appropriate test traffic. In order to obtain representative measurement results, the artificial traffic must emulate the expected customer traffic, or at least work with patterns that allow a suitable quality statement about the customer traffic. In most cases the generation of appropriate test traffic is not trivial. A third problem is the treatment of test traffic. For appropriate measurements it has to be ensured that the injected test traffic is treated as real user traffic. This is especially important if independent third parties or customers themselves perform the measurements. If test traffic is not recognizable as such, providers may suspect an attack because the additional traffic cannot be matched to a customer, and discard the packets. If test packets are clearly marked, providers may give them a preferred treatment in order to manipulate the statistics. Passive measurements are based on the observation of already existing traffic only. They provide real information on how current traffic is served in the observed network. This is exactly the information that is needed for accounting and SLA validation. Since no test traffic is generated, passive measurements only can be applied when the traffic of interest is already present in the network. This is the case for accounting and SLA validation, because both are only

needed during the service usage, i.e. when the traffic of interest is present. Furthermore, measurement results for SLA validation are especially needed in case of quality reduction e.g., due to congestion. In such situations it is unwise to further increase the network load by sending test traffic. Therefore passive measurements are considered here as the method of choice not only for accounting but also for SLA validation. In short one can conclude that active measurements are well suited to measure the network state in a controllable way whereas passive measurements are more appropriate to provide information about the existing traffic in the network

2.5 Data Used for Simulations

To evaluate the performance of the QoS estimation approaches described above and to study the general characteristics of the developed CSP concept, we have selected a simulation based approach. We used two data sets throughout this thesis: the first one is the publicly available P2PSim [p2p08] king data set which represents Internet delay measurements, and the second one is based on a synthetic topology generated by the GT-ITM [ZCB96b] topology generator. In the following we will describe this two data sets in more detail.

2.5.1 P2PSim King data Set

The P2PSim king data set is a full matrix of inter-distance round trip time (RTT) measurements between 1740 Internet name servers. The data set is collected using the *king method* [GSG02] for estimating RTT between any two hosts in the internet by estimating the distance between their name servers. For example if we want to estimate the RTT between Host-A and Host-B in figure 2.6 below, we need first to identify the name server NS-A responsible for the A domain and the name server NS-B responsible for the B domain. Now we measure the RTT between our measurement point and the NS-A as shown in the diagram. In the second step we ask NS-A to recursively resolve a domain name administrated by NS-B which represents measurement step

two in the diagram. Following this request, NS-A forwards the query to NS-B and forwards back the result to the measurement point represented as step three in the diagram. Furthermore we just need to subtract the distance between the measurement point and NS-A from the time taken to complete the recursive DNS query. This will provide an estimation for the distance between NS-A and NS-B. In addition, if one assumes that the network delay between Host-A and NS-A as well as Host-B and NS-B is negligible it is also possible to use this result as an approximation for the distance between Host-A and Host-B. The P2PSim king data set is investigated comprehensively to

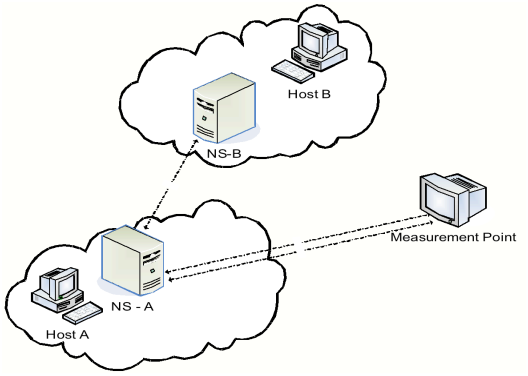


Figure 2.6: King Method

ensure its suitability for representing the real internet with regard to the following perspectives:

- **Network diversity:** The IP addresses of the name servers used for building the P2PSim king data set are publicly available. By using this list of IP addresses and Whois service we map each name server to the corresponding autonomous system (AS) number to check if the members in the name server set are evenly distributed over a broad ASs range. We find the set contains nodes belonging to 1316 ASs, where 1101 ASs are contributing only with one node in the data set.

- **Geographical diversity:** By using the IP addresses list mentioned above and Caidas Netgeo service [Net08], we map each name server to its geographical coordinates (i.e. longitude and latitude). Furthermore using Google Earth software we plot these locations in the world map which shows that most of the name servers are located in North America and Europe and a small percentage in the other parts of the world as shown in figure 2.7(a), and figure 2.7(b) below.

- **Triangle Inequality violations:** We checked all the possible triples in the data set for the triangle inequality violation. That is, if $rtt(a, b)$ denotes the RTT measurement between two nodes a, b we test if for all triple a, b, c from the P2PSim king set one has

$$rtt(a, c) \leq rtt(a, b) + rtt(b, c).$$

  We find that just 4.1% of the triples violating the triangle inequality. As we will see later, the validity of the triangle inequality has an impact to the accuracy of landmark based QoS estimation.

- **Missing measurements:** We found 0.84 % missing measurement pairs in the data set.

As a conclusion of this investigation we find that the P2PSim king data set is suitable to be used in our scenario due to its diversity as well as its appropriateness to be considered as a fully meshed data set.

2.5.2 GT-ITM Topology

As a mean to simulate the schemes over a network satisfying all necessary requirements for a metric space embedding, we generated a synthetic transit-stub topology using the GT-ITM [ZCB96b] topology generator. The generated topology has 10 transit domains, 7 stub domains and 1740 nodes. After the generation of the topology we used the GT-ITM generated link weights to simulate shortest path routing. To have access to bandwidth related data, we calculated an end-to-end bottleneck bandwidth matrix for the same topol-

(a) *Europe*

(b) *North America*

Figure 2.7: King Name Server Distribution

ogy based on widest path routing also with regard to GT-ITM generated link weights.

Chapter 3

State of the Art

After introducing fundamental concepts in the last chapter, we now collect the relevant state-of-the-art with regard to overlay-based Service Composition. Since we will have a strong focus on QoS estimation during this thesis we also collect a state-of-the-art in delay estimation techniques. We will provide a classification of the different schemes available, compare their working principles as well as measurement complexity.

3.1 Overlays and Service Composition

In recent years one can observe a rising interest in overlay-related research. Since 2000, many classical network problems like QoS [SSBK04], Resilience [ABKM01], Multicast [LNS02] or Security [SSAP06] have been addressed using the overlay approach. In addition, one branch of overlay-related research started to study the possibility of using an overlay concept for the flexible, on-demand composition of services. One of the first projects in this direction was the Ninja Project [GWvB+01]. The architecture developed by the project includes the notion of (logical and physical) service paths which have counterparts in many of the following proposals towards a overlay-based Service Composition as e.g. [LGN07], [XN02], [JN04], [LN05], [Kel04], [GNY04], [KBE+07]. Further projects addressing service composition include SAHARA [RAC+02], SWORD [PF02] and the Ambient Networking Project [MSG+07]. The SAHARA project addressed trust and performance related aspects of service composition in case the component services are hosted by different providers. SWORD focused on the generation of service composition plans based on the requirements of the composed service with a strong focus on web service composition, while in Ambient Networks service composition was used for an on-demand processing and routing of media flows. Except [MSG+07], all of these projects had a focus on application and terminology-related topics and did not address in depth the network related issues as e.g. the multi-constraint routing problems related to Service Composition. Inside the overlay community the MONET group of Klara Nahrstedt at the University of Illinois Urbana-Campaign had a strong focus on network and QoS related questions [XN02], [GN02a], [GNCW03],

[GNY04], [GN06], [LGN07], [JN04], [LN05]. Further contributions to this field can be found in [WLL04], [YST+05], [KBE+07]. In addition, from the perspective of active or programmable networks, routing problems closely related to the ones in overlay-based Service Composition have been addressed for example in [CTW01] or [Kel04]. Load balancing and stability issues for Service Composition have been discussed e.g. in [RK03]. In general, the overlay centric work towards a network and QoS aware composition of services can be classified into two main categories:

1. Centralised systems or systems which require global knowledge
2. Decentralised systems

In the following we will describe each category in more detail.

3.1.1 Centralised Systems or Systems Which Require Global Knowledge

As part of this category we consider approaches relying on a central point where service and QoS-related data is aggregated or schemes applying link state routing to address the Service Composition problem. Examples falling into this category are [XN02], [GN02b], [GNCW03], [LGN07], [JN04], [LN05], [CTW01],[Kel04].

In general, all schemes in this category require or assume permanent QoS measurements between all potential overlay nodes which results in a measurement overhead of $O(n^2)$ for n nodes. In addition it is necessary to either broadcast the measurement results and a description of offered services to the whole group, or to deliver them periodically to a central entity responsible for overlay setup.

This extensive measurement and dissemination overhead is required since in an overlay context the situation is different to the case of network layer link state protocols as e.g. OSPF [Moy98]. The reason for this is the fact that the nodes involved in overlays are usually end systems. For a given end system every other end system in the network is a potential overlay neighbour. Thus there is no notion of quasi-static network topology as in the

case of layer three networking. Instead a Service Overlay topology is built up on demand and a link in the overlay in general corresponds to a path in the underlying network. As a consequence schemes in this category usually come with a large measurement overhead. In addition, a service composition problem involving multiple QoS metrics (as e.g. one additive and one concave metric) can already be considered as NP-Complete [GJ79b]. Therefore most schemes consider only one network-specific metric, or in case of two metrics use heuristics to reduce the complexity of the routing problems.

To summarise, the main drawback of this kind of schemes is the overhead introduced by the required periodic updates of QoS and service related information. As a consequence the size of addressable scenarios is limited with regard to number of nodes. On the other hand, since all required QoS information is collected proactively, the resulting service path calculation can be addressed directly after a request. Thus the schemes in this category in general have a short request response time.

3.1.2 Decentralised Systems

Decentralised approaches as [WLL04], [GNY04], [GN06], [GN02a] and [KBE+07] address the service composition problem in a more reactive manner. SpiderNet [GNY04] and CSP [KBE+07] combine a DHT-based search for service components with on-demand QoS measurements. The actual measurements are used to verify that the QoS constraints of the requested composed service are not violated. In contrast to sFlow [WLL04] both schemes also provide means to incorporate the computing resources required for processing into the service composition process. This is important for resource consuming multimedia processing tasks as e.g. for transcoding. A general problem of the search and probe/verify approach of SpiderNet and CSP is that individual branches of the search for a service path are probed independently. Therefore, two probes may involve the same nodes and in case active measurements are used this can result in wrong probing results. In addition, the delay introduced by measurements and the DHT search can have a negative impact on the response time. To address this, CSP as the

only scheme in this group incorporates QoS estimation for delay and bandwidth. A further positive impact using QoS estimation is that compared to SpiderNet, CSP is able to avoid the usual indirection step required by all DHT based decentralised approaches to Service Composition. More details about this will be provided in chapter 6.

3.2 QoS Estimation

As already mentioned in the last section, we consider in case of a decentralised service overlay construction QoS estimation as an alternative to on-demand measurements. To some extend active measurements represent an accurate way of verifying QoS constraints between different node pairs in a network. However, using active measurements to proactively collect QoS related information for a network with n nodes comes with a measurement complexity of $O(n^2)$. In case of P2P scenarios where n is in a range of *millions*, the required measurement traffic will reach a critical scale.

Motivated by this fact, the problem of finding a scalable methodology for predicting QoS related information as e.g. round trip time in P2P scenarios, has evolved to become a research topic in its own right. One of the first suggested solutions for network distance estimation with low measurement overhead was IDMaps [FJP+99]. IDMaps involves the deployment of a set of hosts, called tracers, to play the role of a knowledge base assisting end hosts in estimating network distances between themselves. A different approach based on the idea of introducing network coordinates was first suggested by T.S. Eugene Ng. and Hui Zhang in [NZ02]. Their Global Network Positioning (GNP) can be considered as a pioneering approach in the area of network embedding. GNP associates network nodes with points in a low dimensional Euclidean space. A network node in GNP measures the distances to a finite set of nodes called *landmarks* and uses these measurements to further compute corresponding coordinates in the targeted Euclidean space. After finding the coordinates in the targeted Euclidean space, the network node can predict the network distances to other nodes in the network, based on their coordinates and without any additional measurements. This is possible

since the calculation of the nodes coordinates aims at minimizing the discrepancy between the measured and the predicted network distances. Even though the GNP approach is efficient in reducing the active measurement overhead, it lacks accuracy because of the policy-based routing used today in the Internet [LGP+05]. Many alternative solutions have followed GNP with the goal to improve the accuracy of the actual network embedding, such as [CCRK04], [KZ04], [LHC03], [MS04], [PCW+03], [TC03]. In parallel, a new family of distance estimation techniques emerged using principles like spring embedding [DCKM04] [ST03], or by constructing a distributed network distance database based on active measurements as [WSS05] and [XSLB05]. In this thesis we will introduce Geometric Cluster Placement (GCP)[KZ04] and Netforecast [EKP07]. Based on the underlying principles used by the schemes mentioned we can do a classification into three main classes, which are:

1. *Multidimensional scaling based distance estimation*

2. *Creation and maintenance of a distributed network distance database*

3. *Landmarks based distance estimation*

In the following, we will describe each class including selected reference candidates in more detail.

3.2.1 Multidimensional Scaling-Based Distance Estimation

Vivaldi [DCKM04] is based on a technique from the Multidimensional Scaling (MDS)[CC01] domain also called spring embedding. In order to assign network nodes with points in a low dimensional Euclidean space, Vivaldi models network nodes as masses connected by springs (links), and then relaxes the spring length (i.e. the potential energy) in an iterative manner in order to reach the minimum energy state for the system. In this state the system has its minimal possible overall discrepancy between the predicted and measured

distances. Furthermore the Big Bang Simulation technique (BBS) [ST03] followed the same approach as used in Vivaldi while considering also the kinetic energy of the masses and the friction effect in the relaxation process.

3.2.2 Creation and Maintenance of a Distributed Network Distance Database

This class of schemes uses proactive measurements for building a knowledge base about the underlying network, instead of embedding network nodes into a metric space as in the other classes. Schemes falling to this category have been designed to efficiently answer queries of the form: Who is the closest neighbor to a certain node in the network? Since these schemes are based on direct measurements they have better accuracy. As a drawback the required measurements inject more traffic into the network than in the case of the network embedding technique. For example Meridian [WSS05] is based on a framework consisting of a loosely coupled overlay structure around multi-resolution rings, where queries are forwarded through direct measurements. Each Meridian node keeps track of a finite number of nodes and organizes them into a set of concentric rings of exponentially increasing radii. To disseminate system information between its participants, Meridian employs a form of gossip protocol.

3.2.3 Landmark-Based Distance Estimation

Landmark-based distance estimation schemes associate each node in the network with a point in a metric space, aiming to predict the communication delay between any two nodes in the network by just measuring their corresponding distance in the metric space. The concept is built around the notion of landmarks, which are usually represented by a set of selected nodes used by other nodes as measurement references for their relative position in the network. The cardinality of the landmark set in general has to be greater than or equal to the dimensionality of the target metric space.

Usually in the first phase of such schemes, an initial landmark set is se-

lected, and the landmark coordinates in the metric target space are calculated by minimizing the discrepancy between the measured network distances and the corresponding distances in the metric space. After this step any new joining node can calculate its coordinates by measuring its distances to the landmarks, and minimizing the discrepancy between the measured and calculated distances. The most common landmark selection approaches are:

1. Use a fixed set of landmarks for all nodes joining in phase two mentioned above.

2. On-demand selection of a landmark set out of the nodes having already computed their coordinates.

3.2.3.1 Global Network Positioning

By using a fixed landmark set, Global Network Positioning [NZ02] was a pioneering work in the area of landmark-based distance estimation. Based on a two-part approach each node in the network may calculate its geometrical coordinates. The first part in the GNP approach consists of a set of landmarks collaborating to calculate their coordinates. Furthermore they play the role of a frame of reference by publishing their positioning information to be used by the other nodes in the network. The second part in the GNP approach deals with the other hosts in the network.

3.2.3.2 Practical Internet Coordinates

Practical Internet Coordinates (PIC) [CCRK04] uses the same objective function as suggested by the GNP authors. Unlike the GNP centralised approach with a fixed set of landmarks, PIC is completely decentralised. The set of the landmarks is arbitrarily chosen by each joining host. Furthermore PIC authors proposed three approaches for the landmarks selection

1. Random: Pick landmarks randomly with uniform probability from the existing nodes.

2. Closest: Pick landmarks from the nodes which are in the network proximity of the requesting node.

3. Hybrid: Pick some part of landmarks as in the *random* approach and pick the rest as in the *closest* approach

With regard to the results presented in [CCRK04] the *hybrid* approach has shown better results than the other two. In order to apply the closest or the hybrid approaches mentioned above in a real world scenario, the network node needs more information about the underlying network topology for finding the real neighboring nodes to it. Therefore, PIC presupposes the existence of a P2P substrate with special conditions, in which every member has information about its neighbors (i.e. more information about the nodes in the network). Furthermore PIC suggests two algorithms to assist in finding the closest nodes to a certain node.

3.2.4 Comparison

Table 6.1 shows a comparison of the most relevant schemes used for network delay estimation. To compare these schemes we use the following metrics:

- *Measurement overhead:* this shows the order of measurements needed to maintain the system without including query overhead.
- *Prerequisites:* this is about any special requirement for implementing and/or using the specific scheme.
- *Churn recovery:* this gives information whether churn recovery is supported or not.
- *Infrastructure dependability:* this gives information whether the scheme requires support from the underlying network infrastructure or not.

The following abbreviations are used in Table 6.1: L corresponds to the number of landmarks used by a scheme and H to the total number of participating hosts. In the case of Meridian, N denotes the number of nodes, m the number of rings per node, GI is the overhead per gossip interval, and RI is the overhead per ring management interval. For Netvigator, R denotes the number of milestones (routers) in use. No information is provided about the

measurement overhead for Vivaldi since it is not using active measurements, but only piggyback latency information to the application traffic. Based on

Scheme	Measurement overhead	Prerequisites	Churn recovery
GNP	$O(L^2 + L \cdot H)$	A set of landmarks	No
PIC	$O(L^2 + L \cdot H)$	P2P substrate	No
ICS	$O(L^2 + L \cdot H)$	A set of landmarks	No
Lighthouses	$O(L^2 + L \cdot H)$	Frame of reference	No
Vivaldi	-	Inter-nodes traffic	Yes
Meridian	$O(N \cdot m^2)(GI) + O(log^2 N)(RI)$	-	Yes
Netvigator	$O(L \cdot H + R \cdot H)$	Routers support for traceroute	No

Table 3.1: Comparison of Delay Prediction Schemes

above mentioned criteria, we selected Vivaldi [DCKM04] and PIC [CCRK04] as appropriate schemes for further investigations with regard to the CSP verify process described in chapter 6. The reasons have been as follows:

- None of the two schemes requires explicit support from the underlying network infrastructure.
- There is no need for fixed landmarks, thus the three candidates can work in a fully decentralized manner.
- The candidates exhibit low overhead when compared with other approaches like Meridian.
- The candidates are suitable to be used directly or with less customization in a P2P context, e.g., DHT [RFH+01], compared for instance with Meridian, which is orthogonal to DHT-like structures.
- The selected schemes also show less vulnerability to landmarks replacement than in the case of schemes depending on matrix factorization.

3.2.5 Limitations of Landmark-Based Distance Estimation

There are several obstacles and limitations affecting the accuracy and the implementation of the distance prediction and distance estimation schemes.

Some of these shortcomings are due to the inherent characteristics of the internet and some of them depend on the addressed scheme. As explained before many schemes identify network nodes with points in a low dimensional metric space, and a valid question is "Does the internet inherently satisfy the requirements of a metric space?" as defined below:

Definition 2 (Metric Space)**.** *Let X be a nonempty set, the pair (X, d), where d is a mapping $d : X \times X \mapsto R_0^+$ is called a Metric Space if and only if:*

$$d(x, y) = 0 \quad \Leftrightarrow \quad x = y \tag{3.1}$$

$$d(x, y) = d(y, x) \quad \forall \quad x, y \in X \tag{3.2}$$

$$d(x, y) \leq d(x, z) + d(z, y) \quad \forall \quad x, y, z \in X. \tag{3.3}$$

While the first requirement holds, the second requirement is symmetry which is not holding in the case of the internet because of asymmetric routing. However, this can be solved if the schemes consider RTT instead of one-way delay between two nodes as a measure. The last requirement is Triangle Inequality which is shown to be violated in the internet [ZLPG05] due to policy based inter domain routing. In fact, if exclusively shortest path routing with regard to delay is used in a network, the triangle inequality always holds [CLR90]. As a result of the previous discussion we can state that the internet inherently is not a metric space. However, considering e.g. the publicly available P2PSim King [p2p06] data set, which is based on internet measurements between name servers we found that this data set has a comparable low number (i.e., 4.1%) of triangle inequality violations [Elm07]. In addition, for provider controlled environments it is possible to tune network parameters such as routing metrics with the aim to optimise the accuracy of used prediction techniques. As the gain of changing routing related parameters, the provider can expect higher prediction accuracy. The key question to address for CSP is if the decrease of accuracy caused by QoS estimation is interfering in a critical manner with the service overlay creation process which is evaluated in chapter 7.

3.2.6 Network Embedding and the Curse of Dimensionality

A further question which needs to be addressed for several network distance estimation schemes is how to select the dimension of the geometric target space [PCW+03]. In case of a concrete data set as the P2PSim King [p2p06] data it is possible to performe a principal component analysis to identify the number of eigenvalues with significant magnitudes. The number of these eigenvalues corresponds to a suitable dimension of the target space, as e.g. 8 in case of the King data. To be able to provide more general statements, we have collected some selected results from mathematics.

In the following, we model a computer network as a weighted undirected graph $G = G(V, E)$ with positive edge weights, where V denotes the set of vertices in G and E the set of edges between the vertices. The distance between two vertices $a, b \in V$ is measured using an additive *shortest-path* metric: $d_s(a, b) :=$ "length of the shortest path between a and b". The function $F : G \mapsto R^k$ represents a mapping function (e.g. GNP or PIC), that maps a node x in the network to an element $F(x) \in (R^k, d')$ of the geometric target space. The notation (R^k, d') is used to specify that d' is the geometric distance measure of the target space R^k. To achieve a maximum of freedom regarding this geometric distance function, it is introduced as l^p, defined as:

$$\begin{aligned} l^p(x, y) &:= \left(\sum_{i=1}^{k} |x_i - y_i|^p \right)^{1/p}, 1 \leq p < \infty \\ l^\infty(x, y) &= \max_{i=1,\ldots,k} |x_i - y_i| \end{aligned}$$

In the euclidean case we have $d' = l^2$.

The two simplifications made by this model, namely: (1) that G is *undirected* and (2) that d_s is defined as the *shortest-path* metric allow us to identify the resulting pair (G, d_s) with a *Metric Space* as defined above.

A special group of such mapping functions studied in the area of mathematics are the so-called *isometries*. An *isometry* is characterised by the fact

that it preserves distances. This means that if a mapping from (G, d_s) to (R^k, d') in the form of an *isometry* is found, one has

$$d_s(x, y) = d'(F(x), F(y)), \forall x, y \in (G, d_s).$$

In fact such a mapping is always possible. If for example the number of elements of X is n we can construct such a mapping to (R^n, l^∞) with measurement complexity $O(n^2)$ in the following way: Let $\{x_1, x_2, \ldots, x_k\}$ be the elements of X. The mapping function F is defined as

$$F(x) = (d(x, x_1), d(x, x_2), \ldots, d(x, x_k)).$$

The distance preserving property of F can be seen by

$$l^\infty(F(x), F(y)) = \|F(x) - F(y)\|_\infty = \max_{i=1,\ldots,k} |d(x, x_i) - d(y, x_i)| \leq d(x, y)$$

where equality on the the right side is reached if $x_i = x$ or $x_i = y$.

If one poses constrains to the measurement complexity, the dimension k as well as the metric of the target space, it can not be expected to find always an isometric mapping. For example there is no distance preserving mapping of the simplex in figure 3.1 to (R^2, l^1). More flexibility regarding

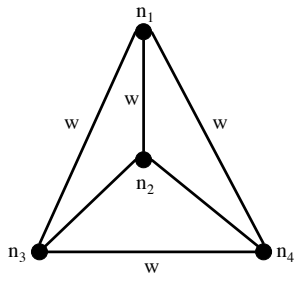


Figure 3.1: Simplex

the metric and dimension of the target space is gained by accepting a degree

of distortion [LLR95] of the inter-object distances caused by the mapping.

Definition 3 (Distortion)**.** *Let $c_1, c_2 \geq 0$ be two real numbers. The distortion of a mapping $F : (G, d_s) \mapsto (R^k, d')$ is defined as the product $c_1 \cdot c_2$ of the lowest values c_1, c_2 for which the inequality*

$$\frac{1}{c_1} \cdot d_s(x, y) \leq d'(F(x), F(y)) \leq c_2 \cdot d_s(x, y)$$

holds for all vertices x, y in G.

The constants c_1 and c_2 denote the *contraction* respectively the *extension* caused by the mapping.

There are two fundamental results from mathematics that provide theoretical bounds for the distortion of a mapping. The theorem of *Bourgain* [Bou85] provides a relation between the number of elements of a metric space, the distortion of the mapping and the dimension of the target space. We cite the version of this theorem provided in [LLR95].

Theorem 1 (Bourgain)**.** *Every n-point metric space (X, d) can be embedded in an $O(\log n)$-dimensional Euclidean space with an $O(\log n)$ distortion.*

The statement of Theorem 1 provides an answer to a general problem in the area of landmark-based distance estimation schemes that is for example formulated in [PCW+03] as: "Could we assume that a vector space with 5 to 7 dimensions can model any network performance metric?". Accepting the simplifications necessary for the identification of a computer network with a metric space, the theorem states that there is a functional relation between the dimension of the geometric target space, the error of the mapping function and the number of nodes to be placed. While being a qualitative result, in the case of a network with n nodes to be mapped, the resulting dimension of the target space has to be selected in an order of $\log(n)$ expecting a worst case distortion of the inter-node distances caused by the mapping also in the order $\log(n)$. At this point, the question rises to which extent a further dimension reduction of the target space has an impact on the distortion of the mapping function. For the purpose of such a dimension reduction (i.e. a mapping from

a high dimensional geometric space to a lower dimensional geometric space) the following theorem, from *Johnson-Lindenstrauss*[1], provides an answer. We cite the version of the theorem provided in [DG99].

Theorem 2. (Johnson-Lindenstrauss) *For any* $(0 < \epsilon < 1)$ *and any integer* n*, let* k *be a positive integer that*

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n.$$

Then for any set V *of* n *points in* R^d *with* $d \geq k$*, there is a mapping* $F : R^d \mapsto R^k$ *such that* $\forall u, v \in V$*,*

$$(1 - \epsilon)\|u - v\|^2 \leq \|F(u) - F(v)\|^2 \leq (1 + \epsilon)\|u - v\|^2.$$

Further this map can be found in randomized polynomial time.

In the case $\epsilon \to 1$ we get from theorem 2 as a lower bound for the dimension of the target space, $k \geq 24 \cdot \log n$. A further result presented in [DG99] states that any weighted graph can be embedded into a $k \leq C \log n$ dimensional geometric space with a distortion of $O(n^{2/k}(\log n)^{3/2}/\sqrt{k})$.

With regard to the presented mathematical results, it is important to note that they do not provide sharp bounds for the required dimension. Results of future work in this area may provide lower bounds especially for the case of network embeddings. As we will see later for real world applications, 8 to 11 dimensional target spaces have shown to be sufficient.

[1]The Theorem is also known as the *Johnson-Lindenstrauss Lemma*.

Chapter 4

A Cooperative Service Provisioning Principle

In this chapter we introduce the cooperative service provision approach proposed in this thesis. Following the classification provided in section 3.1 CSP can be characterised as a reactive and decentralised system for overlay based service composition. To illustrate its basic concepts we first introduce required terminology, followed by a top down description of CSP and the service overlay creation process. After this introduction we provide a more graph theoretic view to the service and network specific aspects of service overlay construction. To do this we will start with a formal description of CSP at service level by introducing the concept of service graphs. In addition we provide a description of the required prerequisites that have to be fulfilled before it is possible to begin the overlay construction process. After this, we start to focus on network level aspects by evolving a graph theoretic formalisation of the addressed multi-constraint based media processing problem. The chapter concludes with a complexity analysis of the CSP approach.

4.1 CSP System Levels

With CSP we adopt an overlay-based service composition approach. A service will be realised by interconnecting the service source and sink through a sequence of processing modules interconnected by using an overlay network. To be more precise with regard to the overlay concept used we introduce first the concept of processing chains.

Definition 4 (CSP Processing Chain)**.** *A CSP Processing Chain is an ordered sequence of processing modules (PMs) connecting a service source and a service sink. Further it is demanded that no two distinct PMs of the chain implement the same functionality.*

Based on the service chain concept we now define a service overlay as follows.

Definition 5 (CSP Service Overlay)**.** *A CSP service overlay corresponds to a directed acyclic graph, connecting a service source and sink, in which each path is a processing chain.*

To address the creation of service overlays we adopt and extend the concept of distributed hash tables. In figure 4.1 it is illustrated how the service and network specific planes of CSP are positioned around the developed CSP DHT concept. To explain the figure we start from its top. The set of all pos-

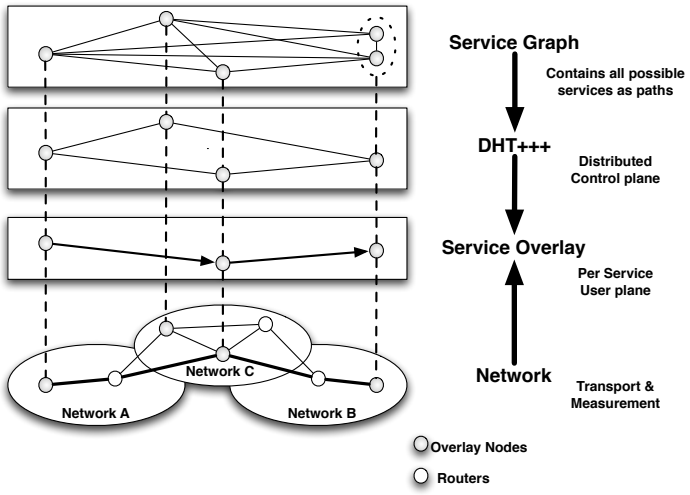


Figure 4.1: CSP Levels

sible services which can be realised by a given CSP system are contained in a graph structure we call *service graph*. A more detailed description of how a service graph can be constructed based on the information about the available service modules will be provided in section 4.3.1. The property of a systems service graph we need for now is the fact that *every service that can be instantiated by the system corresponds to a path in its service graph*[1]. Using a service specific indexing principle (c.f. section 5.2), the service graph structure is mapped into the address space of a DHT [RFH+01] which is extended to be used as the distributed control plane of the CSP system. As the most important DHT extension, support for recursive search requests combined with the verification and propagation of QoS constraints is added. In this context, the recursive search approach is used to find the paths in the embedded service graph which are candidates to be instantiated as service

[1]This approach is different to the one described in [JN04], where the term service graph is used to describe the set of all possible instantiations of a single service.

overlays. The prerequisite of this search based approach is a methodology to reduce the related routing problems to search problems. The development of such a methodology is the first core subject addressed in this PhD. The discussion of different search principles and their evaluation will be presented in chapter 5. As the second core topic, a detailed study of QoS constraint verification techniques based on estimations has been performed. A description of the impact of estimations to CSP signalling and the obtained results are presented in chapter 6.

Given now a request for a composed service, the first requirement is a methodology to decompose it into a set of sub- services. In principle, there are two main possibilities to address this, which are:

- On-demand decomposition of a service at time of service request.
- Offline decomposition of a service using a registration principle.

For CSP we selected an offline service decomposition principle based on service-level agreements, where the corresponding SLAs have to be established between a service provider and a CSP enabled third party provider in advance of the first service request. The main reason for this selection was our target at a lightweight solution without the requirement to involve service description languages. In addition by using SLAs, the actual service decomposition process can be integrated into SLA establishment as we will describe in the next section. As a result it can be expected that the resulting offline decomposition reduces the time from service request till delivery compared to an on-demand decomposition. In case of a P2P-based CSP scenario not involving providers, the service provider entity can be replaced by a peer willing to offer a service, while the third party provider can be substituted by a community of peers offering their processing capabilities.

Given a decomposed service, a service discovery process is started to locate nodes in the network which are hosting the appropriate services modules required for the instantiation of the requested service. To accomplish this service discovery, the service graph information stored in the CSP DHT+++ is used. After a successful discovery step it is required to interconnect the service source and sink through a sequence of processing modules in a way

that the QoS constraints of the target service are not violated. This step corresponds in its most general form to a multi-constraint based routing problem [KKKVM04],[KK01],[RS00], [Has92],[WC96]. In general, such routing problems are NP-complete already in case of one additive and one convex metric [GJ79b]. With CSP we address a comparable but more relaxed problem which will be introduced formally in section 4.4. In contrast to alternative service composition approaches as [XN02], [GNCW03], [JN04], [GN02a], CSP does not rely on a central entity with global knowledge during the task of service overlay setup. The central idea of CSP is to study how the DHT principle can be extended to realise a distributed management plane for the setup and maintenance of service overlays.

4.2 From Service Request to Service Overlay Construction

After the description of CSP core building blocks we will now focus one the required interactions between a Service Provider, Network Provider and a Client from service registration to service delivery. Before doing so, we start by recalling the entity model that is the underlying basis for the development of CSP. As shown in figure 4.2 it contains the three entities Service Provider, Network Provider and Client which are connected via three main interfaces to be used for inter-entity communication. The interfaces shown are called NCI for Network Provider to Client communication and vice versa as well as SCI and SNI for Service provider to Client and Service Provider to Network Provider communication respectively. The SNI interface is used to establish SLAs between Service and Network Provider regulating the cooperative service provisioning process. It can be realised in a flexible way using e.g. Web Services. With regard to the SCI and the NCI we assume in the remainder of this thesis that the SCI is realised using a web portal that is maintained by the Service Provider while the NCI is instantiated by a number of special CSP nodes called Service Bootstrap Nodes (SBNs). SBNs are assumed to be hosted by the Network Provider acting as entry points for

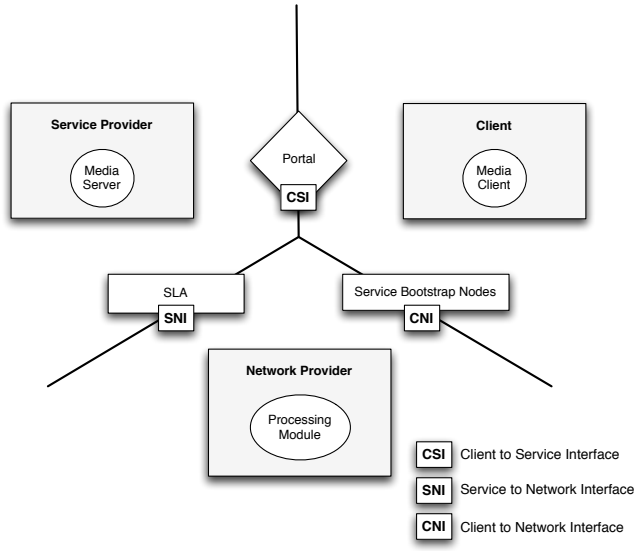


Figure 4.2: CSP Entities and Interfaces

services. There may be several SBNs per service and additionally each SBN can be responsible for different services at the same time.

The required interactions between the Service Provider, Network Provider and Client for service provisioning relate to three phases:

1. Service registration
2. Service request processing
3. Service delivery

In the following we will describe each phase in more detail.

4.2.1 Service Registration

The registration of a new service is initiated from a Service Provider by requesting a SLA application from a Network Provider.

4.2.1.1 Multimedia Transport Service Agreements

To motivate the set of required information and to describe the actual SLA establishment we use composed multimedia services as an example. Con-

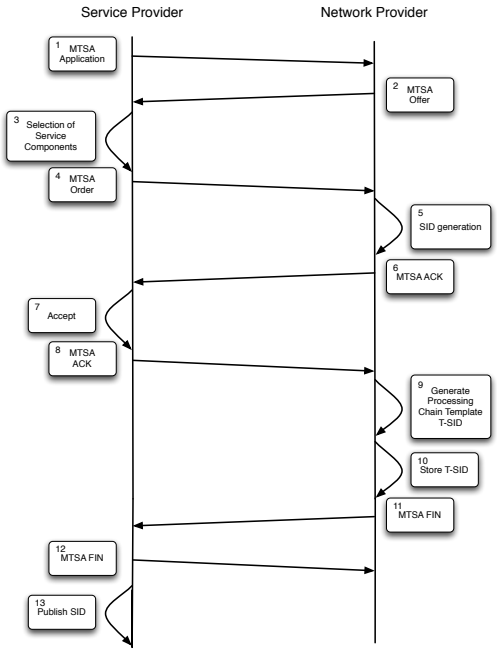
SLA field	Description
SID	A unique Service ID
S_{SID}	A list of transport addresses or unique names of media servers (MSs) and corresponding content format and content encoding (O_{MS}).
P_{SID}	An ordered list containing the processing steps associated with the service
C_{SID}	A vector of constraints associated with the service e.g. QoS constraints as *acceptable delay*, *required (bottleneck) bandwidth* and a monetary constraint i.e. maximal acceptable *cost of processing* in the form of a fee per processing demanded by a provider or enduser

Table 4.1: Required MTSA Fields

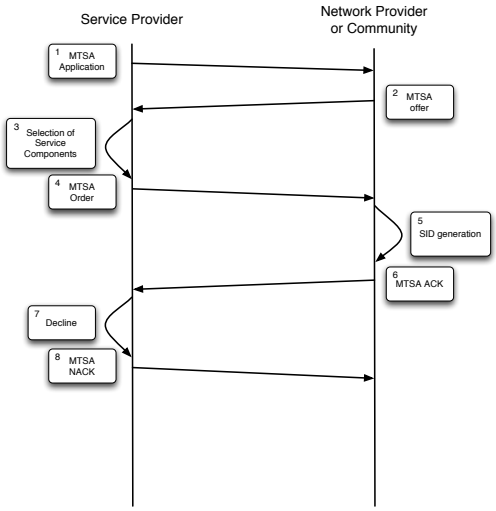
sequently we introduce the SLA principle as Multimedia Transport Service Agreement (MTSA). With MTSAs we address decomposable multimedia services as first downscaling of content with regard to the resolution of a given end user terminal, then tagging of the content for a special user id using e.g. watermarking and finally encrypting it. The contract is assumed to be established between a Multimedia Provider and a Network Provider. The specification and/or standardisation of a concrete MTSA protocol is not in the scope of this thesis. Instead we focus on the required information and signalling.

During the MTSA registration process a *service template*, containing a basic description of the service is generated. In addition, the service decomposition process is performed. At the end of the process a unique service identifier (SID) is generated, which is made public available and used to request the service. The minimal set of required information negotiated during a MTSA agreement is shown in table 4.1. To denote this set of information related to a service with id SID we will use the term *service template* for SID or its corresponding abbreviation T_{SID}. The information flow between service provider and network provider during MTSA generation is shown in figure 4.3. The figure contains two signalling diagrams showing a *successful MTSA negotiation* (c.f. figure 4.3(a)) as well as a case where the Service Provider declines the binding offer of the Network Provider for the MTSA (c.f. figure 4.3(b)). The steps shown in figure 4.3(a) are explained below.

1. *MTSA application:*
   The Service Provider requests a MTSA application (i.e. a list of offered value added transport services) from the Network Provider.



(a) *Successful MTSA negotiation*



(b) *Service Provider declines MTSA*

Figure 4.3: MTSA Signaling Diagram

2. *MTSA offer:*
   The MTSA offer contains a list of offered service components including arising expenses per service component.

3. *Selection of service components*
   The Service Provider selects a set of service components.

4. *MTSA order:*
   The Service Provider orders the list of selected service components, combined with an optional constraint vector.

5. *SID generation:*
   The Network Provider generates a temporary SID for the resulting transport service.

6. *MTSA ACK:*
   The Network Provider sends a binding offer summarising SID, selected service components and constraints.

7. *Accept/Decline:*
   The Service Provider either accepts or declines the offer by sending an MTSA ACK respective NACK.

8. *MTSA ACK:*
   The Service Provider accepts the binding offer.

9. *Generation of service template*
   The Network Provider generates and stores the corresponding service template T_{SID}.

10. *Store T_{SID}:*
    T_{SID} is stored and a set of SBNs is selected and associated with the MTSA.

11. *MTSA FIN:*
    The Network Provider sends the list of service bootstrap nodes and finalises the MTSA establishment.

12. *MTSA FIN*
    The Service Provider finalises the MTSA establishment.

13. *Publish SID:*
    The service provider publishes the SID and the corresponding SBN list via its Service Portal.

The decomposition of a service is done implicitly during the MTSA negotiation process. To see this, we refer to step 2 shown in figure 4.3(a), where the network provider offers a set of service building blocks from which the service provider selects a relevant subset in step 3. After the selection, the result is send back to the network provider (step 4). As a consequence it is always guaranteed that the network provider can assign the list of single processing steps to any negotiated service SID.

To summarise the main interactions to be performed during and after MTSA creation we use our entity model in figure 4.4. During a successful MTSA negotiation initiated by the Service Provider, the Network Provider generates a SID and selects a set of $SBNs$ to be used as contact points to access the new service. This information is encapsulated into a T_{SID} which is stored in a service directory as well as at the involved SBNs. Now the SID and the transport addresses of the responsible $SBNs$ are sent to the Service Provider who can update its service portal with regard to this information. After this it is possible to access the service by connecting to a SBN responsible and requesting SID.

4.2.2 Request Processing

In figure 4.5 the essential steps in case of a service request are shown. From the portal of the service provider the client (MC) receives the required information to access a service i.e. the SID and the address of at least one SBN responsible for the service. With this information, the Client can now send a service request message including the obtained SID to the SBN. It further includes a description of its capabilities denoted as I_{MC}, using e.g. protocols as the Session Description Protocol (SDP) [And02] or Composite Capabilities/Preferences Profiles (CC/PP) [IRRH03]. As soon as the SBN has received the request, it performs a lookup operation for the corresponding T_{SID} extracting all information required to instantiate the service.

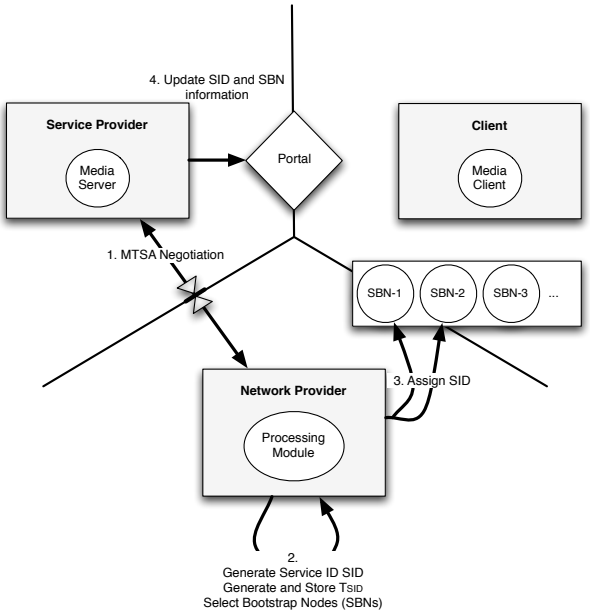


Figure 4.4: MTSA Establishment

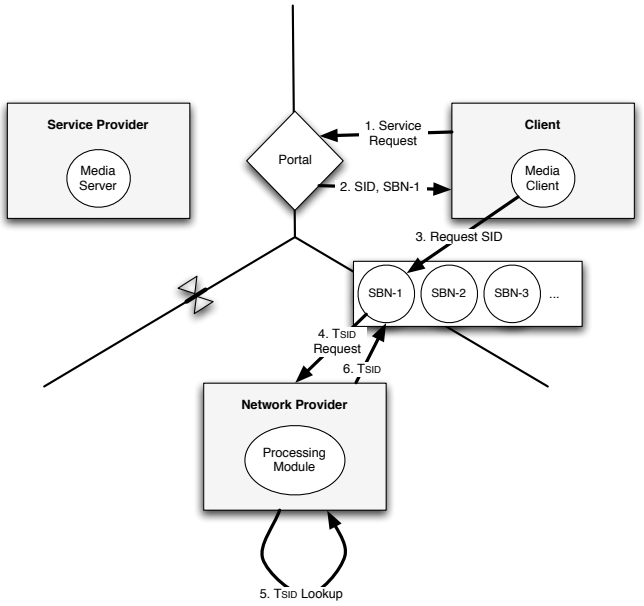


Figure 4.5: Service Request

4.2.3 CSP Overlay Construction and Service Delivery

At the point in time when the CSP overlay creation is about to start, the involved SBN has access to the following information:

1. **From** T_{SID}

   - *Service-specific:* Processing steps $P_{SID} = (P_1, ..., P_i)$ and QoS Constraints $C_{SID} = (C_1, ..., C_n)$.
   - *Server-specific:* At least one source and the corresponding output O_{MS}.
   - *Egress PM-specific:* Required output format O_{MS} and processing function of P_i.

2. **From** MC **request**

   - *Client-specific:* An acceptable input format for the client I_{MC}.

3. **Deduced:**

   - *Ingress PM-specific:* Required input format I and processing function of P_1.

The SBN now initiates the creation of a service overlay for the service with ID SID, and interconnects the service source, all required processing modules and the Client. While the former steps are in general based on the specification of an information model for service and network related data as well as signalling, this step is considered as the core problem to be addressed. In fact it can be interpreted as a generalised multiconstrained optimal path selection or multi constraint based routing problem (CBRP), which is proven to be NP-Complete already in case of two additive metrics [GJ79a], [WC96]. Several algorithms have been proposed for an exact or approximated solution of a given CBRP e.g. [KK01], [OS00], [RS00], [KKKVM04], [Has92]. In general, the suggested solutions are proactive and rely on global knowledge about the QoS situation in the underlying network.

However, in case of a service composition request just the QoS and resource-related information with regard to peers providing the demanded service components is required to find a solution of the corresponding CBRP. These peers may only form a small subset of the whole P2P or provider network. Therefore, it is more efficient to address the service composition for

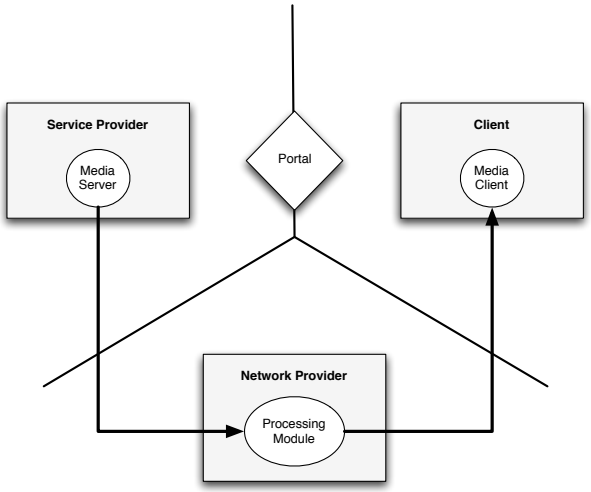


Figure 4.6: Service Delivery

large scale or P2P scenarios in a reactive manner. To describe the reactive CSP approach proposed in this thesis, a more formal approach to the before introduced service and network-related aspects is required. In the following two sections we provide the required notations and concepts.

4.3 CSP at Service Level

To be able to address the service related aspects, we first formalise multimedia transport and processing services [MKS06]. A Processing Module PM is formalised as a triple of the form (I, P, O) where I refers to the possible input formats the PM can read, P refers to the processing function provided by the PM, O refers to the output format of the PM. For simplicity we assume $I, O \in \{x_1, ..., x_m\} \subset \mathbb{N}$ and $P \in \{y_1, ..., y_n\} \subset \mathbb{N}$. Further we define that neither Media Clients nor Media Servers do media processing and thus, they are formalised using an (I, O) notation[2]. A MC requesting content from a MS can be served directly if and only if the input I of the client is compatible

[2] MCs and MSs are considered to be software instances running on a network node. The same node can also host independently a set of PMs performing media processing. Therefore a scenario where a server running a MS instance and several PMs is also possible.

to the output O of the server. In the case of non-compatibility, e.g. a PM implementing transcoding functionality has to be inserted between the MS and the MC to be able to start media delivery using a pipelining principle. During this thesis we use the following definition of compatibility:

Definition 6 (*Compatibility*)**.** *Two Processing Modules* $PM_1 = (I_1, P_1, O_1)$ *and* $PM_2 = (I_2, P_2, O_2)$ *with* I, P, O *values represented by natural numbers are compatible iff* $O_1 = I_2$ *and/or* $I_1 = O_2$*. In the first case we also write* $PM_1 \sim PM_2$*, and in the second case we write* $PM_2 \sim PM_1$*.*

To illustrate the notation introduced in definition 6, the before mentioned pipelining principle is shown in figure 4.7.

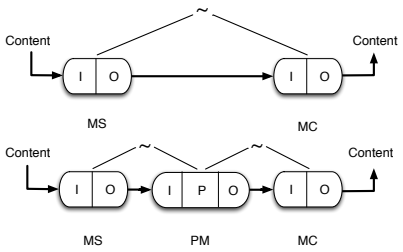


Figure 4.7: Media Delivery Using a Pipelining Principle

To realise for example a service with four processing steps it would be required to find a processing chain

$$PC = (MS, PM_1, PM_2, PM_3, PM_4, MC)$$

where PM_i implements P_i, $i = 1, ..., 4$ and one has

$$MS \sim P_1 \sim P_2 \sim P_3 \sim P_4 \sim MC.$$

The corresponding vector of processing steps $PT = (P_1, P_2, P_3, P_4)$ is also called the processing chain template for PC.

4.3.1 Service graphs

Using this notation, it is possible to represent the information about all possible services that can be realised with a set of PMs using a graph structure. We will also call this graph the *service graph (SG)* defined by a set of processing modules.

Definition 7 (*Service Graph*)**.** *Let $V = \{PM_1, PM_2, \cdots, PM_n\}$ be a set of n processing modules. The service graph associated with V is defined as the undirected graph $SG(V, E)$ with $e = (PM_i, PM_j) \in E :\Leftrightarrow PM_i \sim PM_j$.*

To illustrate the service graph principle, we consider the special case $I, O \in \{1, ..., m\}$ and $P \in \{1, ..., n\}$. For $m = 3$, $n = 4$ we define

$$\begin{aligned} V = \{&(1,1,1),(1,1,2),(1,1,3),(2,1,1),(2,1,2),(2,1,3),(3,1,1),(3,1,2),(3,1,3),\\ &(1,2,1),(1,2,2),(1,2,3),(2,2,1),(2,2,2),(2,2,3),(3,2,1),(3,2,2),(3,2,3),\\ &(1,3,1),(1,3,2),(1,3,3),(2,3,1),(2,3,2),(2,3,3),(3,3,1),(3,3,2),(3,3,3),\\ &(1,4,1),(1,4,2),(1,4,3),(2,4,1),(2,4,2),(2,4,3),(3,4,1),(3,4,2),(3,4,3)\}. \end{aligned}$$

That is, each processing function P is available with all possible I, O combinations. Consequently, each P engenders 9 $(m \cdot m)$ different nodes which leads in our example to a total of 36 $(n \cdot m^2)$ nodes in V. We further have that each node in V is compatible to 12 $(n \cdot m)$ other nodes which results in a total of 48 $(n^2 \cdot m)$ edges. The resulting example service graph is shown in figure 4.8.

Following definition 7, in case one wants to obtain all valid processing chains to realise the service

$$MS \sim P_1 \sim P_2 \sim P_3 \sim P_4 \sim MC,$$

this can be done by integrating MS and MC into the corresponding service graph and then collecting all possible paths from MS to MC including $PM_1, \ldots, PM_n$, with PM_i implements P_i for $i = 1, ..., n$.

For CSP, we approach this path calculation using a search and verify principle with the aim to distribute and parallelise the search for processing

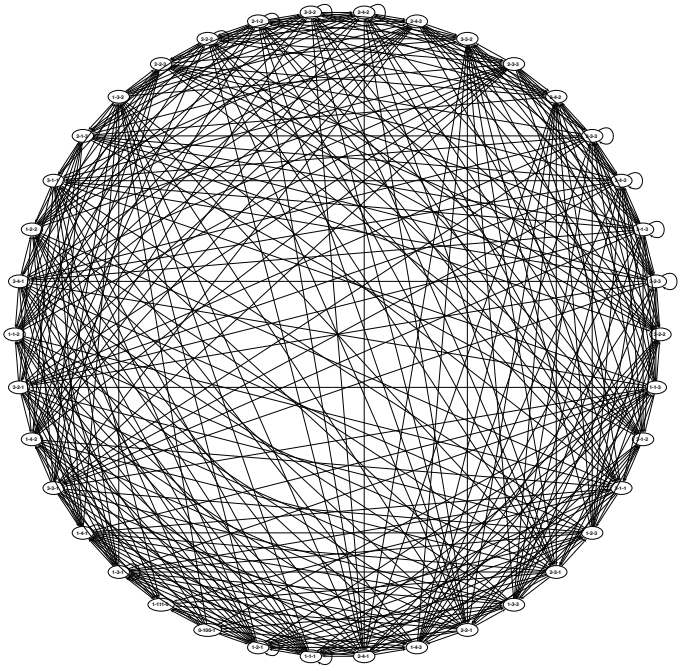


Figure 4.8: Example Service Graph

chains between the nodes forming the DHT+++ layer. The basic principle of the proposed approach is shown for the case of a single processing step in figure 4.9. We assume for now, that two QoS constraints for additive metrics are associated with the service and have been specified via $C_{SID} = (C_1, C_2)$. After the media source received the request for the service with $PC_{SID} = (P_1)$ it initiates a search for processing modules able to accomplish P_1. For each match a verify procedure is started measuring the values $H_1 = (h_{1,1}, h_{1,2})$ between the source and the PM found to verify the constraints. In case $h_{1,1} \leq C_1$ and $h_{1,2} \leq C_2$ the corresponding PM starts a new QoS verification task between itself and the destination with result $h_{2,1}, h_{2,2}$. If $h_{1,1}+h_{2,1} \leq C_1$ and $h_{1,2} + h_{2,2} \leq C_2$ the destination is contacted and informed about the possible chain found. In this simplified example, the client now reports all the possible service chains back to the source which selects the most adequate one based on QoS and cost values and initiates the data transfer. In case of more complex services, the set of all possible processing chain candidates in general defines a directed acyclic graph (DAG) connecting the source and

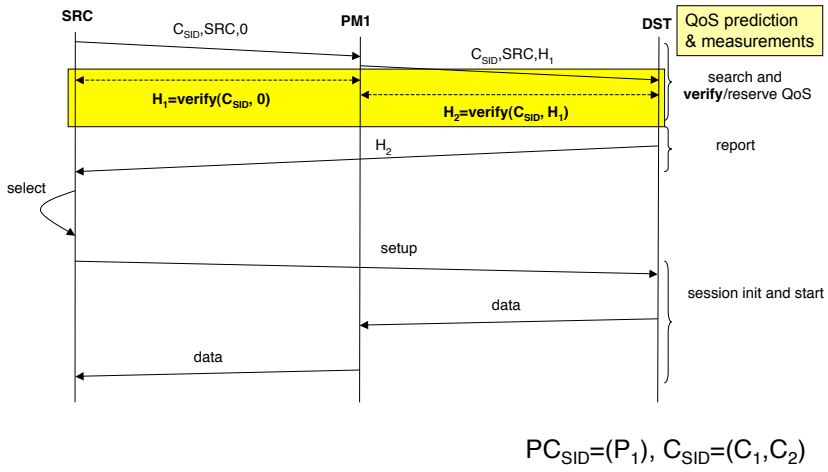


Figure 4.9: Search and Verify Principle

destination. We will call this DAG also the *search graph* associated with a service. Figure 5.1 shows the search graph associated with the service $P_{SID} = (1, 2, 3, 4)$, where we assumed the underlying service graph as shown in figure 4.8. An in-depth discussion of how the proposed search and verify principle can be realised based on a Content Addressable Network using different search principles will be the content of chapter 5.

4.4 CSP at Network Level

To be able to describe the addressed network and routing problems in more detail we will now provide a formal view of CSP at network level. To do this we model a computer network as a weighted, undirected and connected graph $G = G(V, E)$, where V denotes the set of nodes and E the set of edges (i.e. the links between the nodes). We assume that each $v \in V$ offers one and only one processing function p and that $V = \dot{V} \cup \ddot{V}$, where $\dot{V}$ denotes the set of nodes offering a trivial processing function $\hat{p}$ which stands for *no processing* and $\ddot{V}$ denotes the set of nodes offering a non trivial processing

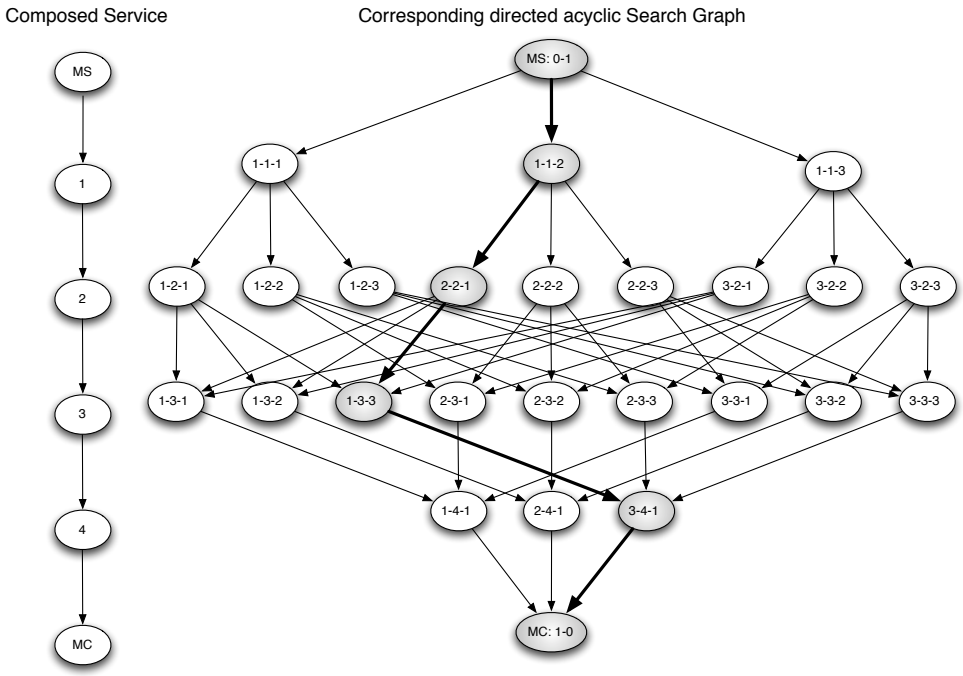


Figure 4.10: Example Search Graph

function. If $F(V)$ is the set of available processing functions at nodes in V we define the function pf as:

$$pf : V \longrightarrow F(V)$$
$$v \longmapsto \begin{cases} \hat{p} & : \ v \in \dot{V} \\ p & : \ v \in \ddot{V}, \ v \textit{ is offering } p \end{cases}$$

Each $v \in \ddot{V}$ has an associated value $v_c \in \mathbb{R}_0^+$ denoting to the costs of using its $pf(v)$. In case we want to focus on the cost values associated with $pf(v)$ we write (v, v_c) instead of just v. For $v \in \dot{V}$ we set $v_c = 0$. For the edges $e \in E$ between vertexes we select delay and bandwidth to be used as *reasonable* candidates in the context of distributed multimedia processing. Therefore, an edge $e \in E$ is written as a four tuple $e = (u, v, e_d, e_b)$, where u denotes the start and v denotes the end vertex of e. The value $e_d \in \mathbb{R}_0^+$ corresponds to the delay introduced by e while the bandwidth of e is denoted by $e_b \in \mathbb{R}_0^+$. In case of a simple graph or when no confusion is possible we omit delay and

bandwidth values just writing $e = (u, v)$. To be able to retrieve cost, delay and bandwidth related values we define the following functions:

$$\begin{array}{rcl@{\qquad}rcl} cost_v : V & \longrightarrow & \mathbb{R}_0^+ & delay_v : V & \longrightarrow & \mathbb{R}_0^+ \\ v & \longmapsto & v_c & v & \longmapsto & v_d \\ bandwidth_e : E & \longrightarrow & \mathbb{R}_0^+ & delay_e : E & \longrightarrow & \mathbb{R}_0^+ \\ e & \longmapsto & e_b & e & \longmapsto & e_d \end{array}$$

A path p in $G(V, E)$ is written as an alternating sequence of its nodes and edges, i.e. $p = (v_1, e_1, v_2, e_2, \ldots, v_k)$, where $e_i = (v_i, v_{i+1})$ $\forall i = 1, \ldots, k-1$. If a vertex v is contained in a path p we also say v *appears* in p. If $P(G)$ denotes the set of all possible paths in G and if $p \in P(G)$ we define the costs, the delay and the bandwidth of a path via the functions:

$cost_p : P(G) \longrightarrow \mathbb{R}_0^+$, defined as

$$cost_p(p) := \sum_{i=1}^{k} cost_v(v_i)$$

$delay_p : P(G) \longrightarrow \mathbb{R}_0^+$ defined as

$$delay_p(p) := \sum_{i=1}^{k-1} delay_e(v_i, v_{i+1})$$

and $bandwidth_p : P(G) \longrightarrow \mathbb{R}_0^+$ defined as

$$bandwidth_p(p) := \min_{i=1,\ldots,k-1} bandwidth_e(v_i, v_{i+1})$$

Following definition 4 in section 4.1, the sequence of processing steps required for realisation of a media service is strictly ordered. To be able to express order we define the function

$$pos : P(G) \times V \longrightarrow \mathbb{N} \cup \{\infty\}$$

returning the order of appearance of a vertex in a path relative to the other nodes. That is for a given $p \in P(G)$ and a $v \in V$

$$pos(p, v_j) = \begin{cases} j & : \ v_j \textit{ appears in } p \textit{ as the } j\textit{th vertex} \\ \infty & : \ v \textit{ does not appear in } p \end{cases}$$

In case all the functions required to realise a given processing chain template *pc* are offered by nodes in $G(V, E)$, the corresponding processing chain template is called *embeddable* into $G(V, E)$. A formal definition of the term is provided below.

Definition 8 (*Embeddable*)**.** *Let* $pc = (p_1, p_2, \ldots, p_l)$ *be a processing chain template and* $G(V, E)$ *be a connected graph. pc is called* embeddable *into* $G(V, E)$ *iff* $\forall p_i, i = 1, ..., l$ *which appear in* p $\exists v \in V$ *such that* $p_i = pf(v)$*.*

4.4.1 Constraint based Media Processing

Using this formalisation, we can formulate the *constraint based media processing problem* and its variants we address in this PhD. The most significant difference between a CBMP and a classical multi constraint based routing problem is the fact that we aim to find a feasible solution instead of the optimal one. The formal definition of a CBMP is:

Problem 1 (CBMP)**.** *Given a graph* $G(V, E)$ *and an ordered processing chain template* $pc = (p_1, p_2, \ldots, p_l)$ *that is* embeddable *into* $G(V, E)$*. Further given a service source* $u \in V$*, a service sink* $v \in V$ *and three constraints* C_{MAX}*,* D_{MAX}*,* $B_{MIN} \in \mathbb{R}_0^+$*. A CBMP is to find a path* $p = (u, \ldots, v)$ *in* $G(V, E)$ *realising pc such that the following constraints are fulfilled:*

1. *Cost constraint:*

$$cost_p(p) \leq C_{MAX}$$

2. *Delay constraint:*

$$delay_p(p) \leq D_{MAX}$$

3. *Bandwidth constraint:*

$$bandwidth_p(p) \geq B_{MIN}$$

4. *Order constraint: let* $1 \leq i < j \leq l$ *and* x, y *appear in* p *with* $p_i = pf(x)$, $p_j = pf(y)$ *then we have*

$$1 \leq pos(p, x) < pos(p, y) < \infty.$$

In addition it can also be required to search the *cheapest* path while fulfilling a delay and bandwidth constraints, or to find a solution having the lowest delay but having costs below and bandwidth above a given threshold which leads to the following two variants of Problem 1.

Problem 2 (Least Cost CBMP (LCBMP))**.** *A* least cost CBMP *is to find the optimal solution* $p = (u, \ldots, v)$ *with regard to the* $cost_p$ *metric in* $G(V, E)$ *for a CBMP as described in Problem 1.*

Problem 3 (Least Delay CBMP (LDBMP))**.** *A* least delay CBMP *is to find the optimal solution* $p = (u, \ldots, v)$ *with regard to the* $delay_p$ *metric in* $G(V, E)$ *for a CBMP as described in Problem 1.*

In the remainder of the thesis we will focus on the least cost CMBP when applicable since this problem combines in a natural way the network QoS requirements of service composition and a computation related metric identified with processing costs.

4.5 Complexity Analysis of CSP

After the description of service and network-specific aspects of CSP we will focus in this section on a complexity analysis. The main target of the analysis is to derive upper bounds for the communication overhead caused by the proposed search and verify principle. To do this, we start with the observation that each node hosting a possible member PM for a composed service

may also be actively integrated into the search process for the corresponding processing chain. Thus the number of processing chains available for a given service together with the processing chain template length can be used to calculate a bound for the *principle scope* of a CSP search process. In this context *principle scope* denotes the number of nodes *actively involved* into the search process for a processing chain of length n. A node is considered to be actively involved into a search in case it has to trigger a CSP-related search request, while the scope of underlying DHT routing operations is not considered. As the result we will provide a CSP related *principle scope* statement independent of the used DHT scheme.

In order to address principle scope in its broadest fashion, we assume a breadth first based search where no branch is stopped because the QoS constraints of the requested service are not fulfilled. We start by examining the question: "Given a relation between the I, P, O values of the available PMs in the system as well as a processing chain template of length n, how many possible processing chains can we expect?". To answer this, we consider the random-variables $I_j(\omega), O_j(\omega)$ which can take values out of the set $\{x_1, ..., x_m\}$. The x_i denote different as well as independent input and output formats. Further we assume n different processing functions P$\in \{y_1, ..., y_n\}$, and define the probabilities

$$\wp(O_j(\omega) = x_i) =: p^O_{j,i} \text{ and } \wp(I_j(\omega) = x_i) =: p^I_{j,i},$$

which denote the probability that $O_j(\omega)$ respectively $I_j(\omega)$ have the value x_i. Using this notation a processing chain, is any ordered vector

$$((I_1(\omega), y_1, O_1(\omega)), ..., (I_n(\omega), y_n, O_n(\omega))).$$

For $1 \leq j \leq n$, N_j denotes the number of occurrences of PMs $(I_j(\omega), y_j, O_j(\omega))$ with P $= y_j$. Therefore, the number of all possible processing chains is $\prod_{j=1}^{n} N_j = N_C$. A chain is valid iff $O_j(\omega) = I_{j+1}(\omega)$ for any $1 \leq j \leq n-1$, and the respective set of valid chains is denoted by C^*.

We are now interested in $\mathbb{E}(\#(C^*))$, the expectation of the random-

variable $\#(C^*)$[3].

Lemma 1. *Let M be a set of processing modules PM=(I,P,O). N_j denotes the occurrences of triples $(I_j(\omega), y_j, O_j(\omega))$ with $P = y_j$ and $I_j(\omega)$,$O_j(\omega)$ as independent random variables. Then we have that the number of expected valid processing chains $\mathbb{E}(\#(C^*))$ can be computed as*

$$\mathbb{E}(\#(C^*)) = \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}.$$

Given an additional start input, respectively end output of the form $I_1 = x_s$, $O_n = x_c$ one has

$$\mathbb{E}(\#(C^*)) = p^I_{1,s} p^O_{n,c} \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}.$$

Proof. We consider the random variable

$$\#(C^*) = \sum_{l=1}^{N_C} X(\omega_l)$$

denoting the number of valid processing chains as the sum of the random variable X defined as

$$X(\omega_l) := \begin{cases} 1, & \omega_l \textit{ is valid} \\ 0, & \omega_l \textit{ is not valid} \end{cases}$$

Thus, we can compute the expected number of valid processing chains as

$$\mathbb{E}(\#(C^*)) = \mathbb{E}(\sum_{l=1}^{N_C} X(\omega_l)) = N_C \mathbb{E}(X) = N_C \wp(O_1(\omega) = I_2(\omega), ..., O_{n-1}(\omega) = I_n(\omega))$$

.

[3]We don't use the notation $\#(C^*)(\hat{\omega})$ to avoid confusion with the ω used in case of $I(\omega)$ and $O(\omega)$

Because of the independence of $O(\omega)_j$ and $I(\omega)_j$ we have

$$\wp(O_1(\omega) = I_2(\omega), ..., O_{n-1}(\omega) = I_n(\omega)) = \prod_{j=1}^{n-1} \wp(O_j(\omega) = I_{j+1}(\omega))$$

and

$$\wp(O_j(\omega) = I_{j+1}(\omega)) = \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}.$$

Based on this result, the (expectation value for) the number of correct chains in a random sample can be computed as:

$$\begin{aligned} \mathbb{E}(\#(C^*)) &= \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \wp(O_j(\omega) = I_{j+1}(\omega)) \\ &= \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}. \end{aligned}$$

To obtain the expected number of valid chains with the given start input and end output we have to multiply this term with the probabilities $p^I_{1,s}$ and $p^O_{n,c}$. □

Combining this result with the fact that for a processing chain of length n, $n-1$ nodes[4] have to actively trigger a CSP search request we can state that the expected number of nodes that are actively involved into the corresponding search process is in the scale of:

$$(n-1) \cdot p^I_{1,s} p^O_{n,c} \cdot \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}.$$

As a direct consequence of lemma 1 and the argumentation made above, we can now formulate the following result for an upper bound of the principle scope of a CSP search.

Lemma 2 (Principle scope of a CSP Search)**.** *Under the premises of lemma 1, and assuming one PM per CSP node one has for the expected principle*

[4]We assume one PM per node.

scope (PS) of the search for a processing chain of length n:

$$PS \leq (n-1) \cdot \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i}. \tag{4.1}$$

Given an additional start input, respectively end output constraint of the form $I = x_s$, $O = y_c$, *then:*

$$PS \leq (n-1) \cdot p^I_{1,s} \cdot p^O_{n,c} \cdot \prod_{k=1}^{n} N_k \prod_{j=1}^{n-1} \sum_{i=1}^{m} p^I_{j+1,i} p^O_{j,i} \tag{4.2}$$

Proof. The proof of the lemma follows immediately from lemma1 and the fact that for a processing chain of length n, $n-1$ nodes have actively to trigger a CSP search request. The inequality sign in 4.2 is caused by the fact that in a distributed CSP search it is possible that two distinct branches of a search strike at the same node as shown in figure 4.11(b). In such a case, lemma 1 still expects two processing chains, but since these chains have PMs in common, the number of triggered search requests is less than in the case of disjoint chains as e.g. shown in figure 4.11(a). Consequently the values provided in 4.1 and 4.2 act as upper bounds for the principle scope of a CSP search. □

4.6 Summary

After this theoretical analysis we will now provide a more concrete view on the addressed problem space. To illustrate above results by an example we make the following simplifying assumptions:

(i) For each processing function we assume the same number of PMs, which is: $N = N_j$, for $1 \leq j \leq n$.

(ii) Each processing function appears k times with all possible I, O combinations, which gives: $p^I_{j,i} = p^O_{j,i} = \frac{1}{m}$ and $N = k \cdot m^2$.

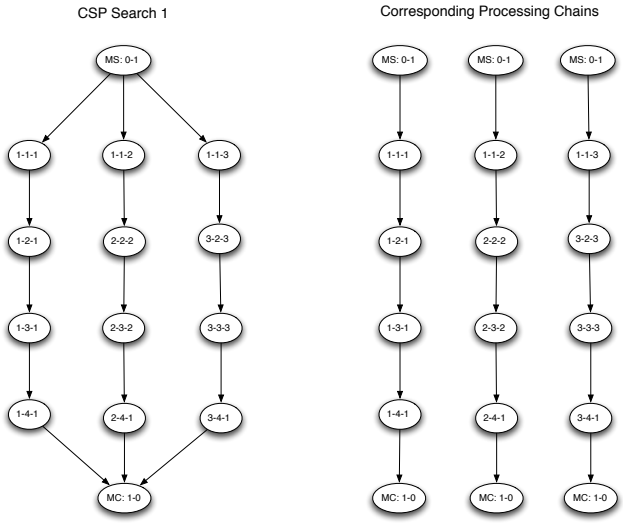
The statement of lemma 1 is now that under these premises we can expect:

$$\mathbb{E}(\#(C*)) = \frac{1}{m^2} \cdot \frac{N^n}{m^{n-1}} = \frac{k^n \cdot m^{2n}}{m^{n+1}} = k^n \cdot m^{n-1}$$

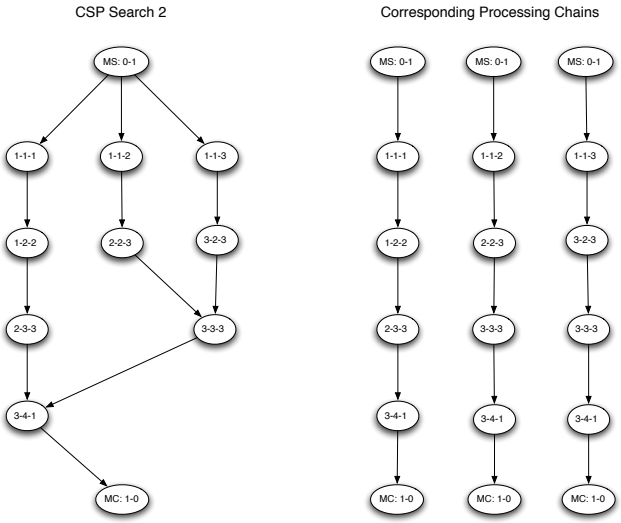
valid processing chains of length n connecting a given media server and client. In addition, for this example the number of nodes actively involved into a breadth first CSP search grows exponential with regard to n because of

$$n \cdot \frac{N^n}{m^{n-1}} = n \cdot k^n \cdot m^{n-1}.$$

In a real world scenario, each I or O value may denote a profile combining multiple parameters related to input or output tasks. Summarising for example common video formats as H.261, H.263, H.264, MPEG-2, RealVideo, WindowsMedia and DivX in just three possible resolutions already results in 21 possible I, O formats only for Video. On the other hand, I, O parameters can also be used to denote different Layer II technologies usable by a given Processing Module. Examples for possible processing functions are Transcoding, Caching, Watermarking, Encryption or Error Correction to mention just a few. To illustrate the size of the addressed problem space, we assume 3 different I, O formats, 5 possible processing functions appearing 10 times each with all possible I, O combinations. Following the results of our study we can expect to find $10^5 \cdot 3^4 = 8100000$ valid processing chains of length 5. To be able to reduce the *principle scope* of the corresponding CSP search process, we will start to investigate in the following chapter how different CSP search principles can be realised.



(a) *Principle Scope of 10*



(b) *Principle Scope of 9*

Figure 4.11: Different Search Processes Resulting in the Same Number of Processing Chains

Chapter 5

CSP Search Approach

As introduced in the last chapter, we address the creation of Service Overlays by combining a DHT-based search principle with a hop-by-hop QoS constraint verification and propagation technique. A DHT-based approach has been selected because of the following considerations:

- DHTs represent a well studied domain for search-based problems.
- The resulting system can be realised in a distributed way and inherits the self-organisation and self-repair properties of DHTs.
- In case CSP nodes hosting processing functionality are members of the DHT, it is possible to identify valid Processing Chains during the search process.

Common DHTs offer a functionality to map keys to transport addresses, where information about a given key can be fetched. An example for a key is the name of a file and the mapping is established by using a hashfunction. The used hashfunction, a structured overlay topology and an overlay routing scheme are combined to realise a search platform with provable search characteristics. To be able to adopt the DHT principle for CSP it is required to add more functionality. For realising the proposed CSP *search and verify* principle we need to be able to combine QoS verify with search functionality. In case we are looking for Processing Chains of length greater than one it is required to be able to search for chains of keys. As we will see later, it is also a benefit if the compatibility of two processing modules is to some extend invariant with regard to the used DHT hash function.

We will therefore extend a classical DHT towards a CSP DHT+++ supporting:

1. An indexing scheme that transfers the compatibility relations of available PMs into the DHT address space.
2. A recursive search scheme for Processing Chains.
3. Primitives for the on demand verification or estimation of QoS constraints between arbitrary pairs of CSP nodes.

To address the required extension in the remainder of this chapter we will first discuss and formalise the underlying concepts of a search based approach to CBMP. To do this we show how a standard *breadth first* (BF) search strategy can be applied. After this, we describe the realisation of a CSP DHT+++ based on a CAN DHT. We address the indexing question and derive a new CSP specific *range query* principle. A CAN has been selected because this DHT allows to realise the mentioned indexing and range query principles in an efficient and elegant way.

5.1 Reduction of CBMP to a Search Problem

The prerequisite of the proposed search based approach is a way to reduce the addressed CBMP to a search problem. To do this, we start at the point in time when the CSP overlay creation is about to start. Following the results of section 4.2.3 the set of information for a requested service with id SID available to the responsible Service Bootstrap Node SBN can be summarised as follows:

Service-specific:

List of processing steps $(P_1, ..., P_i)$
QoS constraints $(C_1, ..., C_n)$.

Server-specific:

At least one Media Server MS and the corresponding output format O_{MS}.

Ingress PM-specific:

The required input format $I = O_{MS}$ and processing function of PM_1.

Client-specific:

An acceptable input format I_{MC} for the Media Client MC.

Egress PM-specific:

The required output format $O = I_{MC}$ and processing function for PM_i.

Based on this information, it is possible to generate the following incomplete Processing Chain description for the requested service:

$$PC_{SID} = (I_{MS}, O_{MS}) \sim (I, P_1, *) \sim ... \sim (*, P_i, O) \sim (I_{MC}, O_{MC}).$$

Using the Service Graph concept introduced in the last chapter, we can determine the set of all Processing Chains available to instantiate PC_{SID}. To do this, we add the MS and MC to the Service Graph and collect all paths connecting MS and MC by a chain of PMs implementing P_1,...,P_i. The resulting graph structure that originates from collecting all these paths we also called the *search graph* for PC_{SID}. In the following we consider again the Service Graph defined by the following set of PMs:

$$\begin{aligned} V = \{&(1,1,1),(1,1,2),(1,1,3),(2,1,1),(2,1,2),(2,1,3),(3,1,1),(3,1,2),(3,1,3),\\ &(1,2,1),(1,2,2),(1,2,3),(2,2,1),(2,2,2),(2,2,3),(3,2,1),(3,2,2),(3,2,3)\\ &(1,3,1),(1,3,2),(1,3,3),(2,3,1),(2,3,2),(2,3,3),(3,3,1),(3,3,2),(3,3,3),\\ &(1,4,1),(1,4,2),(1,4,3),(2,4,1),(2,4,2),(2,4,3),(3,4,1),(3,4,2),(3,4,3)\}. \end{aligned}$$

The corresponding search graph for the incomplete Processing Chain description

$$(0,1) \sim (1,1,*) \sim (*,2,*) \sim (*,3,*) \sim (*,4,1) \sim (1,0)$$

is shown in the middle of figure 5.1. As a general property of search graphs we note that in case of an ordered incomplete Processing Chain PC_{SID} in which each processing function appears only once, the corresponding search graph is a directed acyclic graph (DAG). We substantiate this attribute of search graphs in the following lemma.

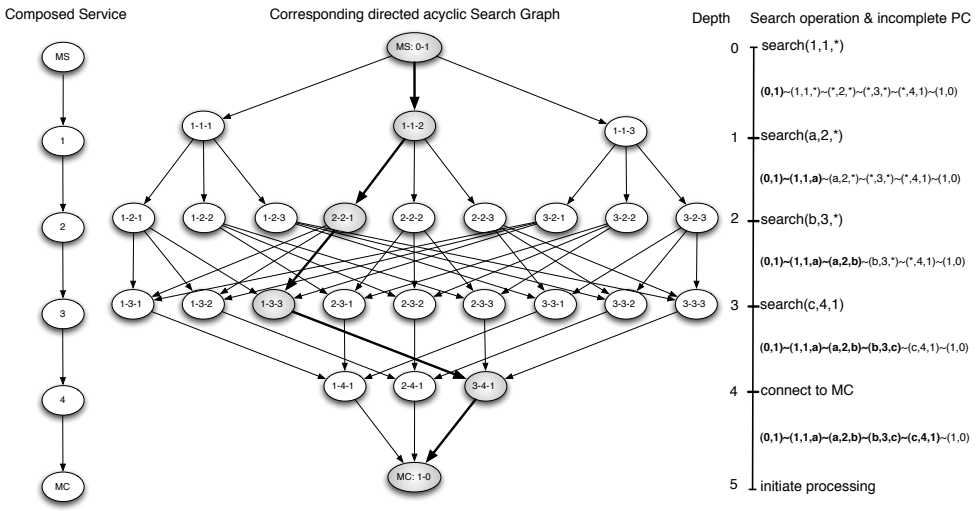


Figure 5.1: Search Graph and Required Search Operations

Lemma 3. *Let $SG(V, E)$ be a service graph and*

$$PC_{SID} = (I_{MS}, O_{MS}) \sim (I, P_1, *) \sim \cdots \sim (*, P_i, O) \sim (I_{MC}, O_{MC}) \quad (5.1)$$

an ordered processing chain which is embeddable into SG. If PC_{SID} has the property that each processing step appears only once then the search graph corresponding to PC_{SID} is a DAG.

Proof. The search graph is directed with edges of the form (PM_j, PM_{j+1}) for PM_j, PM_{j+1} implementing P_j respectively P_{j+1}. A cycle in the search graph would correspond to $P_l = P_m$, for some $l \neq m$ which is not possible since each processing step appears only once by assumption of the lemma. □

Since we consider only Processing Chains fulfilling the premises of lemma 5.1 (c.f. definition 4 in chapter 4) in this thesis, we can extract a search graph as illustrated on the right side of figure 5.1.

To put it more concretely: Given an arbitrarily connected service graph $SG(V, E)$ and an incomplete Processing Chain PC_{SID} of length l as input we start at the MS. As shown in the figure 5.1 we process PC_{SID} from left to right utilising a BF principle. At depth $l - 3$ the BF approach is stopped since now all information required to determine the corresponding next PM

in a unique manner are available. Because the premises of lemma 5.1 hold, the search process will continue without loops and terminate at the client.

Pseudo code for an algorithm to extract the search graph for a given PC_{SID} out of SG based on this approach is shown in figure 5.2. In the code *V[DAG]* and *E[DAG]* correspond to the *vertex* respectively *edge* set of the resulting search graph labelled *DAG*. *Adj(SG,u)* denotes the adjacency list of a vertex u in SG. The used functions pop, push, and shift are the standard list manipulation operations. The value *Nil* denotes an empty list.

```
SG-DAG(PC_SID, SG)
 1  MC ← pop(PC_SID)
 2  MS ← shift(PC_SID)
 3  push(next-round, MS)
 4  DAG ← NIL
 5  for p ∈ PC_SID
 6        do
 7           this-round ← next-round
 8           next-round ← NIL
 9           for u ∈ this-round
10                 do
11                    ADD-VERTEX(V[DAG], u)
12                    for v ∈ Adj(SG, u)
13                          do
14                             if get-p(v) = p and get-i(v) = get-o(u)
15                                then
16                                      ADD-VERTEX(V[DAG], v)
17                                      ADD-EDGE(E[DAG], u, v)
18                                      push(next-round, v)
19 ADD-VERTEX(V[DAG],MC)
20 this-round ← next-round
21 next-round ← NIL
22 for u ∈ this-round
23       do
24          for v ∈ Adj(SG, u)
25                do
26                   if get-p(v) = p and get-i(v) = get-o(u) and get-i(MC) = get-o(v)
27                     then
28                          ADD-VERTEX(V[DAG],v)
29                          ADD-EDGE(E[DAG],u,v)
30                          ADD-EDGE(E[DAG],v,MC)
```

Figure 5.2: Search Graph Calculation

5.2 Distributed Hash Tables as a CSP Search Domain

With the SG-DAG function described above we have been able to reduce the CBMP to a BF search problem. However, the algorithm operates on a systems Service Graph. We will now focus on the question how we can realise an algorithms which uses a DHT instead. To approach this we first collect the required DHT search functions to be realised. The search graph shown in figure 5.1 is used again as an example. As we can see on the right side of the figure, a DHT-based search function would need to support two kinds of search requests:

1. Queries of the form $\textsc{search}(I, P, O)$ to find all PMs with $PM = (I, P, O)$

2. Queries of the form $\textsc{search}(I, P, *)$ to find all PMs with a defined I and P while having an arbitrary output.

The second type of queries is also known as *range query*. In our case a range query principle is required to find all Possible Processing Chains available in a CSP system. To illustrate this we refer to the right hand side of figure 5.1. In case of the ingress processing module PM_1 there is not enough information available to formulate an exact search query. Thus, to find all possible Processing Chains we need to search for a list of available PMs capable of accepting the input format I with a processing function P_1, and an arbitrary output O. The corresponding range query is of the form $\textsc{search}(I, P_1, *)$. However, before it is possible to realise concrete search schemes for CSP, it is required to specify how to map the information related to a given PM into a DHT address space. The indexing schemes which are used for such tasks are in general tightly coupled with the DHT overlay topology, addressing scheme and routing. For this reason we now first introduce a CSP-specific indexing scheme for a CAN DHT [RFH+01].

5.2.1 CSP Specific Indexing

The address space of a CAN DHT as described in section 2.3.1 can be identified with a d-dimensional torus. A CAN supports basic hash table operations for *key* and *value* pairs. In our case the *key* is derived from the properties of a given PM and the value corresponds to the network address of the node hosting the PM.

Since we can determine relevant properties of each PMs by its independent I, P and O values, we use an own dimension for each of these parameters. The address space of the corresponding three-dimensional CAN is defined as the cube $[0,t] \times [0,t] \times [0,t] \subset \mathbb{N}^3$ for a suitable $t \in \mathbb{N}$.

During the CAN join procedure a new CSP node n obtains a CAN overlay address in the form of a three-dimensional coordinate out of $[0,t]^3$ as e.g. described in [Rat02]. To be able to publish information about a PM_1 with values (I_1, P_1, O_1) hosted by n we define the CSP indexing function as:

$$\text{HASH}_{CSP}(I_1, P_1, O_1) := (\text{h}(I_1), \text{h}(P_1), \text{h}(O_1)) \in \mathbb{R}^3,$$

where $\text{h} : \mathbb{N} \mapsto [0,t] \subset \mathbb{N}$ denotes a collision free hash function.

The advantage of using a three-dimensional CAN together with this indexing approach is the fact that the $\sim$ relation is invariant with regard to HASH_{CSP} since:

$$PM_i \sim PM_J \Leftrightarrow \text{HASH}_{CAN}(PM_i) \sim \text{HASH}_{CAN}(PM_J).$$

As a consequence, it is possible to derive compatibility of two PMs by comparing their hash values in the DHT address space, a fact that we can exploit in the next section.

5.2.2 CSP Specific Range Queries

As already motivated in the beginning of section 5.2, the required CSP DHT search function has to support two kinds of queries which are: Queries of the form $\text{SEARCH}(I, P, O)$ returning all PMs stored in the DHT with

$PM = (I, P, O)$ and secondly range queries denoted in the following as $\text{search}_{\text{RQ}}(I, P, *)$. In figure 5.3 we illustrate how such a query can be realised

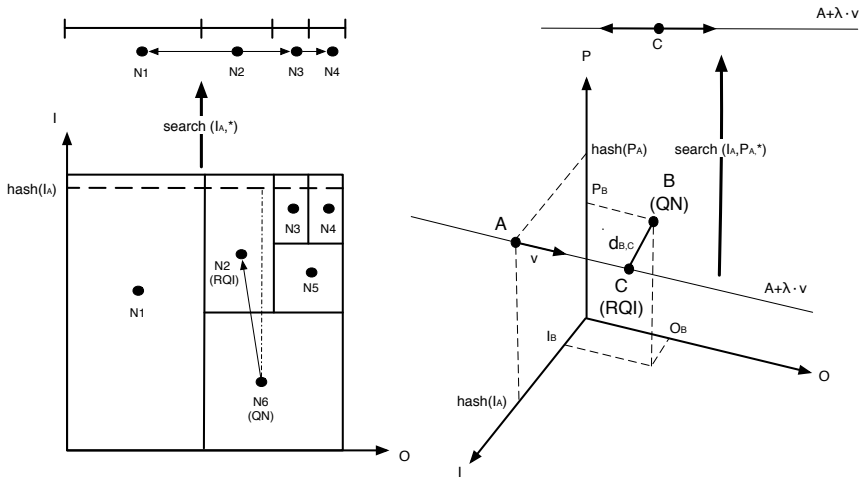


Figure 5.3: Range Queries in a 2D and 3D CAN

using a two or three dimensional CAN and the before introduced indexing scheme. Since the two-dimensional case shown on the left side of the figure is intuitive, we describe only the three-dimensional case for a query of the form $\text{search}_{\text{RQ}}(I_A, P_A, *)$.

Because of the properties of the used indexing scheme, information about all PMs with $PM_A = (I_A, P_A, *)$ is stored along the line $g(\lambda) = A + \lambda \cdot (0, 0, 1)$ for $A = (\text{h}(I_A), \text{h}(P_A), 0)$. For a query node (QN) B with CAN address (I_B, P_B, O_B), the closest point on g with regard to euclidean distance $d_{B,C}$ is $C = (\text{h}(I_A), \text{h}(P_A), O_B)$. We define the node owning the CAN territory including C as the range query initiator (RQI). As soon as this node receives the range query from B it forwards it to all its neighbours along the line g.

For a query of the form $\text{search}(*, P_A, O_A)$ we have (using the same argumentation) $g(\lambda) = A + \lambda \cdot (1, 0, 0)$ with $A = (0, \text{h}(P_A), \text{h}(O_A))$, and $C = (I_B, \text{h}(P_A), \text{h}(O_A))$. With regard to the scope of the described *range query* principle we provide the following lemma.

Lemma 4 (Scope of a range query in CANs)**.** *Given a Content-Addressable Network in a d-dimensional geometric space containing n nodes. We further*

assume that the CAN is partitioned into territories of equal size. The average scope of a CSP range query is $\left(\frac{d}{4}+1\right) * n^{(1/d)} - 1$ *and is therefore* $O(n^{1/d})$.

Proof. We note that for a CAN in a d-dimensional space partitioned into territories of equal size, the average routing path length is $(d/4) * n^{(1/d)}$ [RFH+01]. The scope of a range query can be identified with the scope of two separate CAN routing requests. The first one, with source QN and destination RQI, can be assumed to be of average scope $(d/4) * n^{(1/d)}$. Now the RQI node is forwarding the range query along a line in $\mathbb{R}^d$ crossing the territory of $n^{1/d} - 1$ nodes (see also figure 5.3). Therefore we have an average message complexity of $\left(\frac{d}{4}+1\right) * n^{(1/d)} - 1$ which is $O(n^{1/d})$. □

5.2.3 A DHT-based Approach to CBMP

Utilising the above introduced indexing and range query principle we are now able to describe a DHT-based approach to CBMP. The underlying idea of the presented scheme is to distribute the search for Processing Chains between the set of nodes hosting the required processing functionality in the form of PMs.

We will describe how a BF and a *depth first* (DF) [CLR90] search principle can be realised based on a CAN equipped with the described indexing scheme and range query algorithm. In contrast to BF, the DF principle is based on two functions controlling the search process in the form of a *forward checking* and *backtracking* function. For the actual backtracking and forward checking functions we assume that each PM has an associated cost value. The cost of a Processing Chain is calculated as the sum of costs of its PMs. For *forward checking* and *backtracking* we always select the PM leading to the lowest cost of the current fraction of the Processing Chain. As a result we always have a single search branch in the case of the DF approach. We further include constraints for *delay* and *bottleneck bandwidth* which have to be verified during the search process. The proposed BF algorithm has three phases which correspond to *Initialisation*, *Expansion* and *Contraction*. We start with the description of the Initialisation phase.

Initialisation The Initialisation begins when the CSP overlay creation is about to start. At this point in time we can assume to know the partly completed Processing Chain description of length l with $i = l - 2$ PMs:

$$(I_{MS}, O_{MS}) \sim (I, P_1, *) \sim \cdots \sim (*, P_i, O) \sim (I_{MC}, O_{MC})$$

I addition we know the IP address of the MC (IP_{MC}) and three additional scalar constraints C_{Max} for acceptable *costs*, D_{Max} for acceptable *delay* and B_{Min} for required *bottleneck bandwidth* of the composed service.

Base on this information, a media server MS receiving the service request can generate an initial CSP search query of the form:

$$\begin{aligned} &\text{search}_{\text{CSP}}((I, P_1, *) \sim \cdots \sim (*, P_i, O))), IP_{MC} \\ &\{c < C_{Max}\}, \{d < D_{Max}\}, \{b > B_{Min}\} \\ &c = 0, d = 0, b = \infty \end{aligned}$$

Expansion As the root of the search graph for the Processing Chain the MS initiates the query $\text{search}_{\text{RQ}}(I, P_1, *)$, piggybacking all information required to recursively continue the search i.e.:

$$Q_0 := \begin{cases} \text{search}_{\text{CSP}}((*, P_2, *) \sim \cdots \sim (*, P_i, O))), IP_{MC} \\ \{c < C_{Max}\}, \{d < D_{Max}\}, \{b > B_{Min}\} \\ c = c_s, d = 0, b = \infty \end{cases}$$

In Q_0 the value c_s denotes the actual cost of the media provisioning by the MS. Each DHT node passed by the range query and holding information about a node n hosting a required $PM = (I, P_1, O_n)$ forwards the piggybacked search query and the IP address of the MS to n (c.f. figure 5.4 where the Query Node is the MS). Any node n can now actively determine the delay and bottleneck values d_1, b_1 using active measurements or estimation techniques between itself and the Query Node (in this case the MS). In addition n can get the value c_1 corresponding to the cost for using its PM implementing the processing function P_1.

If $c_s + c_1 < C_{Max}, d_1 < D_{Max}$ and $b_1 > B_{Min}$, the node n creates a new

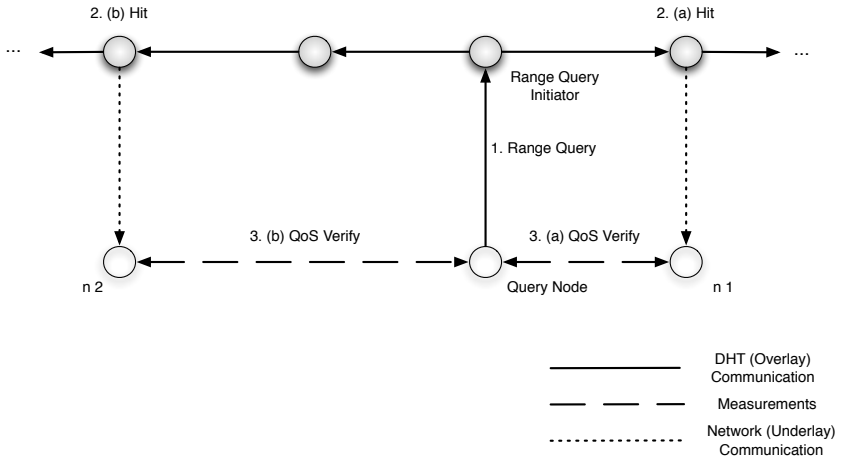


Figure 5.4: Range Query Step for BF-based Search

range query of the form $\text{search}_{\text{RQ}}(O_n, P_2, *)$ piggybacking:

$$Q_1 := \begin{cases} \text{search}_{\text{CSP}}(*, P_3, *) \sim \cdots \sim (*, P_i, O)), IP_{MC} \\ \{c < C_{Max}\}, \{d < D_{Max}\}, \{b > B_{Min}\} \\ c = c_s + c_1, d = d_1, b = b_1. \end{cases}$$

This process is continued in a recursive manner until search depth $l - 3$ is reached.

Contraction At depth $i = l - 3$ the search arrives at a node n hosting the $PM = (I_n, P_{i-1}, O_n)$. The Query Q_{l-4} obtained from its predecessor is:

$$Q_{l-4} := \begin{cases} \text{search}_{\text{CSP}}((*, P_i, O)), IP_{MC} \\ \{c < C_{Max}\}, \{d < D_{Max}\}, \{b > B_{Min}\} \\ c = c_s + c_1 + ... + c_{i-2}, d = d_1 + ... + d_{i-2}, b = \min\{b, b_{i-2}\} \end{cases}$$

Thus the remaining incomplete Processing Chain is:

$$(*, P_i, O) \sim (I_{MC}, O_{MC})$$

After a successful constraint verification, the node n can formulate an exact search query of the form search(I_n, P_i, I_{MC}) piggybacking:

$$Q_{l-3} := \begin{cases} IP_{MC} \\ \{c < C_{Max}\}, \{d < D_{Max}\}, \{b > B_{Min}\} \\ c = c_s + c_1 + ... + c_{i-1}, d = d_1 + ... + d_{i-1}, b = \min\{b, b_{i-1}\} \end{cases}$$

Each node found by this query performs the remaining constraint verifications and sends a report message to the MC, including information about the final c , d and b values. Based on this information, the MC can select the actual Processing Chain to be instantiated backwards from MC to MS.

5.2.4 Modifications for a DF-based Search

To realise a DF search-based version of the described algorithm we modify the range query in the *expansion* phase. By adding an additional step, we let the range query Initiator select the node n to continue the search after collecting the results of the QoS measurements. The required additional information exchange is shown in figure 5.5. The next hop can either be

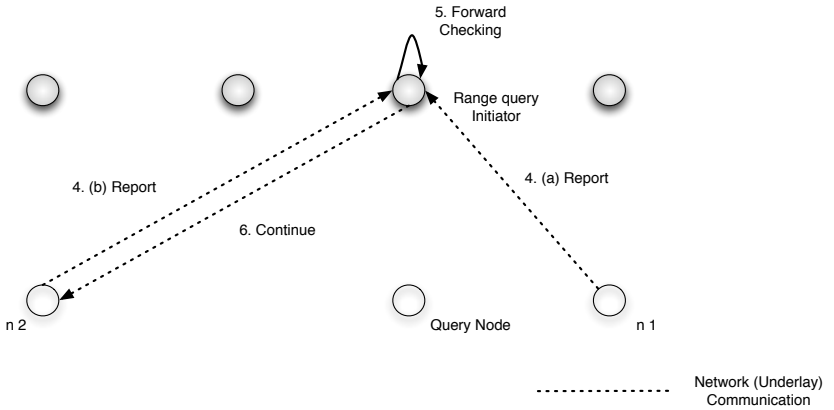


Figure 5.5: Forward checking for DF-based search

selected as the first node reporting a successful QoS verification or the RQI first collects all QoS verification results and selects the next hop based on a predefined forward checking algorithm. In the last case, it is required that

the RQI is able to determine when the range query is finished and it has received all reports. To do this we can exploit the fact, that in a CAN DHT it is possible for each CAN node to decide if it is responsible for a part of the DHT containing an address space boundary or not. As shown in figure 5.6, these nodes can send an indicator that the actual range query has reached the left or right boundary of the CAN address space. Based on this information, the RQI can decide if the range query is about to be finished or not.

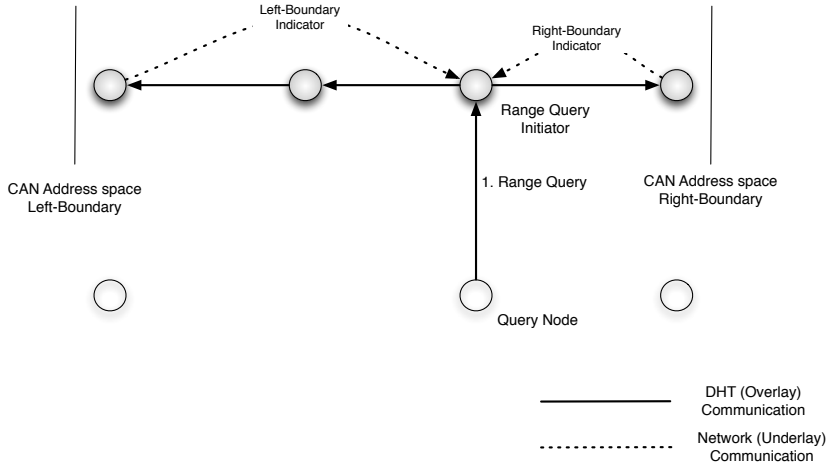


Figure 5.6: Boundary Indicators

If the QoS constraints associated with the composed service are violated, the RQI triggers backtracking by informing the Query Node as shown in figure 5.7.

If a predefined number k of independent branches of a DF search should be realised, it is required that the node starting the recursive CSP search process selects in advance a unique id e.g. in form of a number for each branch. After this, it starts k DF search processes, each with a unique id. If a node already involved in a DF search receives a request for a search with a different id, it ignores this request.

Based on the results of this chapter, we are able to realise the core of the distributed CSP control plane. In chapter 7 we will evaluate the performance of different search principles used in conjunction with the described CAN-based approach to CBMP. In the next chapter, we will address the question

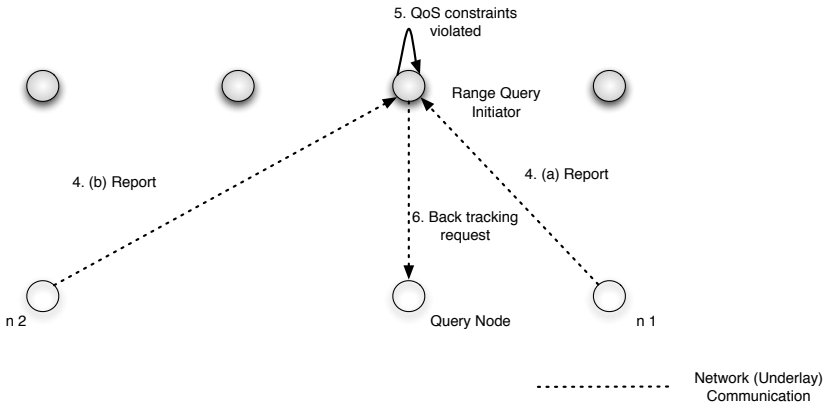


Figure 5.7: Backtracking for DF-based search

of how to realise the required verification of QoS constraints.

Chapter 6

CSP Verify Approach

In this chapter we will focus at the CSP verify operation. Besides the search principles studied in the last chapter, the verification of QoS constraints can be considered as one of the core tasks in the addressed Service Overlay scenarios. In fact, each search step is followed by a verify operation which aims to ensure that the QoS constraints associated with a composed service are not violated during Service Overlay setup. As motivated in section 4.4 of chapter 4 we use delay (rtt) and bottleneck bandwidth as the two network QoS reference metrics during this thesis. In addition, we incorporate *processing cost* as a computing resource specific metric, which is considered to be coupled to the actual load of an overlay node candidate as well as the resource requirements of a required processing step. In principle, there are two main ways to address the required network QoS constraint verification:

1. On demand or reactive QoS measurements (i.e. after a service has been requested).

2. Proactive QoS measurements (i.e. in advance of a service request).

Beneath the different pros and cons with regard to measurement overhead, time complexity and accuracy of the selected verify method, it is also important to take the characteristics of the DHT based CSP management into account as we will describe in section 6.1. Following the results of this section, the main focus of the remaining chapter is on how we can reduce the amount of required proactive QoS measurements while being still able to provide an acceptable *estimation* for network QoS-related metrics. Starting with delay (rtt) estimation, we first introduce Geometric Cluster Placement which is a result of the work related to this thesis. After a motivation of GCP design, it is compared with the state of the art network distance estimation schemes as described in chapter 3. As a result of this comparison we can state that GCP shows superior accuracy in case of estimating small delays. To cover more general scenarios where we have to estimate large and medium delays as well, we combine GCP with clustering and a triangulation heuristic first used by Hotz [Hot04] in this context. Because the resulting approach is independent of GCP, we introduce with *Netforecast* [EKP07] an own name for it. In a

next step we discuss if and how landmark based estimation techniques can be used for bottleneck bandwidth estimation.

6.1 Proactive Versus Reactive QoS Measurements

The pros and cons for reactive and proactive QoS verification in our context are comparable to those discussed in the routing area as e.g. in [RT99]. In principle it is a trade-off between speed and overhead. The proactive collection of QoS information can allow a faster service overlay setup, because in an ideal case no measurements are required during a CSP search process since such information has been collected proactively. The drawback is that this method relies on a periodic updating mechanism involving a constant measurement and propagation of QoS information. For a reactive verification approach, no such update and propagation method is required. On the other hand, the required on demand QoS measurements for verification add an additional delay because of the time requirements of the measurement process.

In addition to this general argumentation, we also have to consider the interaction of the CSP verify operation with the CSP search functions as described in chapter 5. A description of the required steps to include a verify operation based on on-demand active measurements into the CSP search process is shown in figure 6.1 for a single search and verify step. The figure illustrates the basic message exchange required for a node (Querying Node) to determine its successor in a service chain. Since CSP is based on a DHT principle, a redirection step (step 2) is required in order to return the IP address of the overlay candidate to the querying node. Only after this step is it possible to initiate the measurements required for network QoS verification (e.g. utilising OpenIMP [ope08]), and to exchange information with regard to processing costs.

In case of a proactive collection of QoS measurements, the situation changes. Since QoS related information for the network between querying

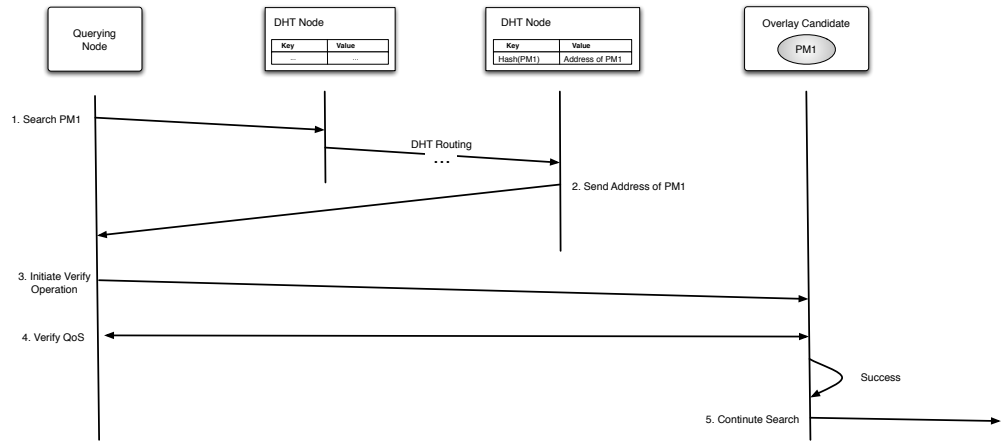


Figure 6.1: On-Demand Verification of QoS Constraints

node and overlay candidate is already known, it is possible to start the verify operation before step 2 as we will describe below in more detail. With regard to the expected measurement overhead of a proactive approach one has to consider that collecting the end-to-end QoS related information for a network with n nodes comes with a measurement complexity of $O(n^2)$. In case of the addressed Peer-to-Peer scenarios, where n is usually in the range of *millions*, the required measurement traffic will fastly reach an unacceptable scale.

One possibility to be still proactive is to reduce the requirements on the accuracy of the QoS information. In fact the problem of finding scalable methodologies for predicting QoS related information as e.g. rtt in the addressed scenarios with a measurement complexity of $c(n) << n^2$ has evolved to become a research topic in its own right and for a state of the art we refer to chapter 3. To realise a corresponding estimation function for CSP we will describe an approach for proactive QoS estimation based on QoS coordinates in the remainder of this chapter. The actual QoS coordinates for each node in the system will be determined using a scheme from the area of *landmark based (network) distance estimation* which will be introduced in the next section. As a result of using QoS estimations we note that beneath the anticipated reduction of service overlay setup time because of available QoS estimations, such QoS coordinates also allow to avoid the DHT redirection step. As shown in figure 6.1 it is now possible to handle the search for a

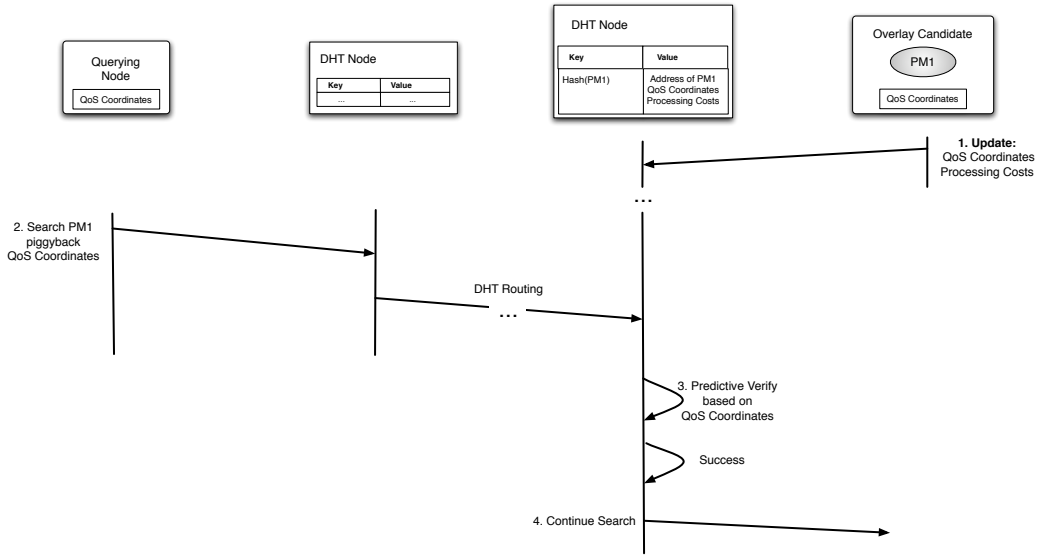


Figure 6.2: Verification of QoS Constraints Based on QoS Estimation

service chain candidate completely in the DHT address space which will have a further positive impact to CSP performance. An analysis of the accuracy of an estimation based CSP scheme will be provided in section 7.

6.2 Delay Estimation with Geometric Cluster Placement

Before we start with a description of Geometric Cluster Placement [KZ04], we first recall the principle of landmark based distance estimation. For a state of the art with regard to landmark based distance estimation or more general network distance estimation, we refer to section 3.2.

Landmark based distance estimation schemes associate each node n in the network with a point $x = (x_1, \ldots, x_k)$ in a metric space based on an embedding process, whereas a metric space is defined as follows (c.f. [LLR95]):

Definition 9 (Metric Space)**.** *Let X be a nonempty set, the pair (X, d), where d is a mapping $d : X \times X \mapsto R_0^+$ is called a Metric Space if and only*

if:

$$d(x,y) = 0 \quad \Leftrightarrow \quad x = y \tag{6.1}$$

$$d(x,y) = d(y,x) \quad \forall \quad x,y \in X \tag{6.2}$$

$$d(x,y) \leq d(x,z) + d(z,y) \quad \forall \quad x,y,z \in X. \tag{6.3}$$

The target is to predict the communication delay (rtt) between any two nodes in the network after this embedding by just measuring their corresponding distance in the metric space which is in most of the cases the k dimensional vector space of real numbers, $\mathbb{R}^k$. The corresponding metric used is in general one of the l^p norms:

$$\begin{aligned} l^p(x,y) &:= \left(\sum_{i=1}^{k} |x_i - y_i|^p \right)^{1/p}, 1 \leq p < \infty \\ l^\infty(x,y) &= \max_{i=1,\ldots,k} |x_i - y_i| \end{aligned}$$

Most common is l^2, which corresponds to the Euclidean case.

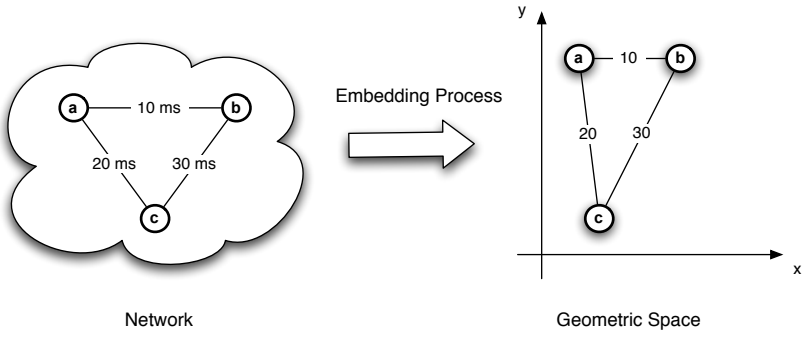


Figure 6.3: Network Embedding

The concept of this embedding is built around the notion of landmark nodes, which are represented by a set of selected nodes. These nodes can be used by other members of the network as measurement beacons to determine their relative position in the network. The cardinality of the landmark set is most of the time greater than the dimensionally of the target metric

space to assure the uniqueness of the calculated coordinates. The underlying principle of this approach is comparable to a triangulation as e.g. used in case of the Global Positioning System and we describe it in the following for Global Network Positioning (GNP) [NZ02]. Using a static landmark set, GNP was a pioneering work in the area of landmark based distance estimation. Based on a two-part approach each node in the network can calculate its geometric coordinates as illustrated in figure 6.4 and 6.5. If the dimension of the geometric target space is k, GNP requires a minimum of $k+1$ landmark computers (i.e. measurement points). First the coordinates of the landmarks $L = \{l_1, l_2, ..., l_{(k+1)}\}$ are calculated by measuring the inter landmark distances (rtt) using a standard ping or traceroute tool. If C_{l_i} denotes

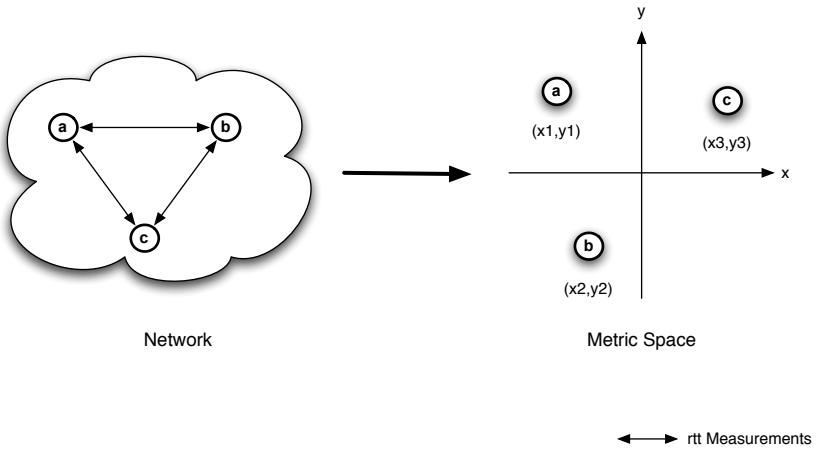


Figure 6.4: GNP Landmark Operations

a preliminary coordinate for the landmark l_i, $d_{i,j}$ denotes the measured rtt between landmark l_i and l_j and $\hat{d}_{i,j}$ the distance of their corresponding preliminary coordinates in the metric space, the final coordinates of the GNP landmarks are calculated by minimising the objective function:

$$f_{obj}(C_{l_1}, C_{l_2}, \ldots, C_{l_k}) = \sum_{l_i, l_j \in L, i \neq j} \left(\frac{d_{i,j} - \hat{d}_{i,j}}{d_{i,j}} \right)^2 .$$

The actual minimisation is performed in [NZ02] using the Simplex-Downhill

Algorithm [NM65]. From now on, the landmarks together with their coordinates play the role of a frame of reference for the other nodes in the network. If node n wants to calculate its coordinates C_n, the network distance of n to the $k+1$ landmarks is measured using again ping or traceroute. The results of this measurement together with the corresponding landmark coordinates are now used to calculate the coordinates of node n by minimising the following objective function [NZ02]:

$$f_{obj}(C_n) = \sum_{l_i \in L} \left(\frac{d_{i,n} - \hat{d}_{i,n}}{d_{i,n}} \right)^2$$

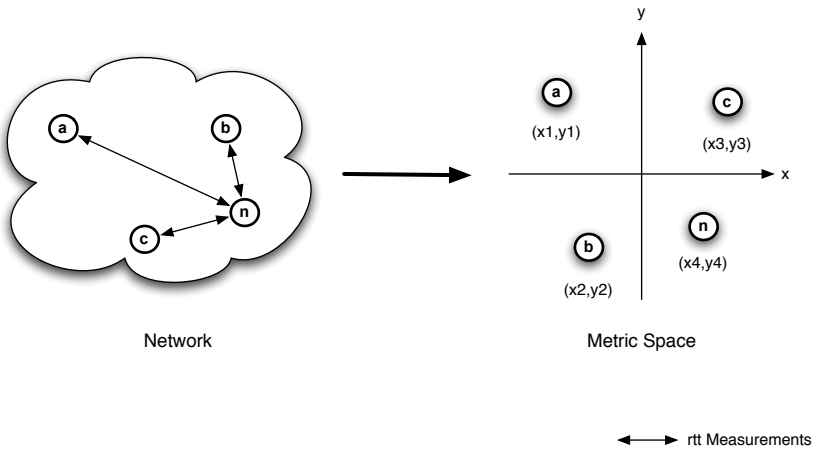


Figure 6.5: GNP Normal Node Operation

6.2.1 Impact of Landmark Selection

There are two main drawbacks that can be identified analysing the GNP approach:

1. A static set of landmarks is used to define the frame of reference for the embedding process. For a real world implementation this results in a single point of failure problem. In addition it is not clear how to replace a landmark in case of failure.

2. The selection of initial landmarks is done more or less incidental which can lead to cases with high estimation error as we will describe below.

To address the first point with GCP we performed experiments with a dynamic landmark principle to realise a scheme more resilient against failure of landmarks. With regard to the second issue we focused on the impact of landmark selection to the quality of the embedding. One of the most critical situations in this context is shown in figure 6.6. The situation shown

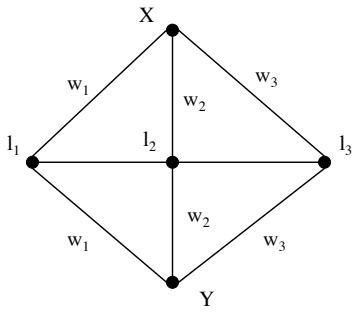


Figure 6.6: Critical Situation

in the figure can be described as follows: If for two nodes x, y the distances to the landmarks out of the set $L = \{l_1, l_2, l_3\}$ are similar or equal (i.e. $d_{x,i} = d_{y,i} = w_i$, $i = 1, 2, 3$), both points will get in general the same or a very similar coordinate in the metric space, while having a network distance unequal to zero.

The underlying idea of the dynamic GCP landmark selection approach for avoiding such a situation is illustrated in figure 6.7[1]. If the distance $D_s(x, L)$ of a point x to the landmark set L is defined as $\min_{l \in L} d_s(x, l)$, the situation of figure 6.7 is the following: Since $D_s(x, L) < r_1 < r_2$ and $D_s(y, L) > r_2$, a perfect mapping function F not causing distortion would calculate coordinates $F(x)$ and $F(y)$ with

$$\hat{d}_{F(x),F(y)} > r_2 - r_1 > 0.$$

[1]See also the proof of Theorem of Bourgain in [Bou85].

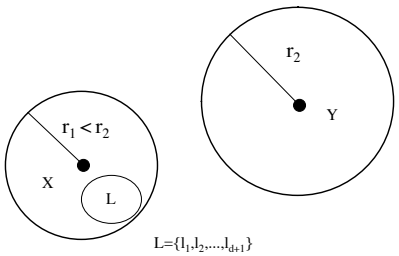


Figure 6.7: Geometric Argument for GCP Landmark Selection

However, in practice we cannot assume to have a perfect function F, and in case we use a GNP like mapping function F we have to consider a degree of *distortion* introduced by the actual mapping process. This error has also an influence to the values r_1 and r_2 which have to be calculated after applying the GNP mapping function as

$$\begin{aligned} r_1^{'} &= \min_{F(l)} \hat{d}_{F(x),F(l)} \\ r_2^{'} &= \min_{F(l)} \hat{d}_{F(y),F(l)}. \end{aligned}$$

The heuristic we use in case of GCP is that we still assume to have:

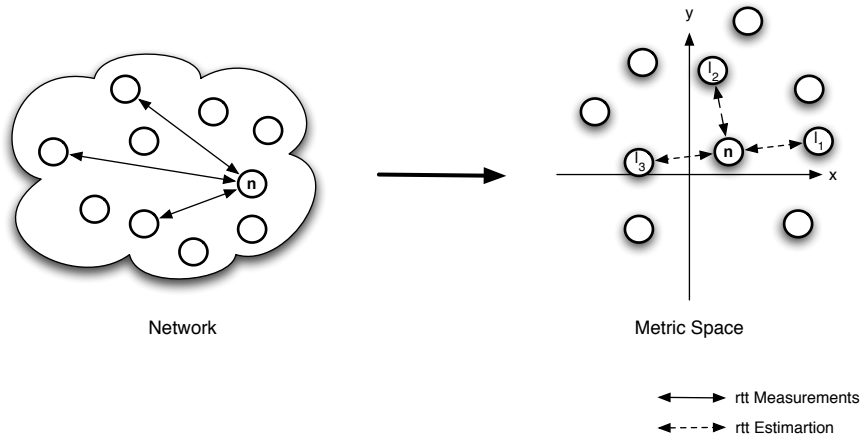
$$\hat{d}_{F(x),F(y)} > r_2^{'} - r_1^{'} > 0.$$

6.2.2 A Framework for the Realisation of GCP

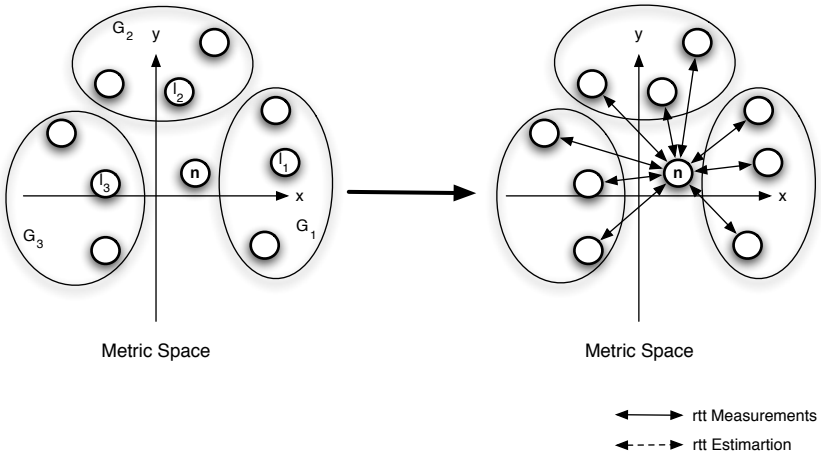
We will now describe the framework used by GCP to realise an embedding function incorporating the concepts described above. As a prerequisite the nodes in the target network (i.e. the nodes to intend to apply GCP) need to have information about their neighbours. Such information is available if those nodes are organised in a CSP CAN DHT.

As with GNP, if k is the dimension of the geometric target space a set

with a minimum of $k + 1$ nodes is required to initialise the system. In practice, the first $k + 1$ joining nodes can be used for this task and there are no restrictions or special requirements for this initialisation step other than the set size. After the system is initialised, figure 6.8 illustrates the working procedure for a newly joining node which we explain in more details in the following:



(a) GCP Step 1 and Step 2



(b) GCP Step 3 and Step 4

Figure 6.8: GCP Normal Node Operation (Cluster Size $c = 3$)

1. In the first step, a preliminary coordinate of the node n about to join the network is calculated using $k + 1$ random nodes already in the

system as landmarks.

2. With the help of this first coordinate, the node determines the set of its $k+1$ geometric closest neighbours as landmarks for the second measurement step. In case of a CAN DHT created as in [RHKS02] those nodes are usually the direct DHT Overlay neighbours or reachable via these direct neighbours.

3. Each l_i of this new landmark nodes determines now a cluster G_i out of its c closest neighbours in the corresponding metric space with the property that
$$\cap_{i=1}^{k+1} G_i = \emptyset.$$

4. The node n is now measured again by all the nodes in each cluster, and the coordinate of n is calculated with the help of the $k+1$ measurement results
$$d_s(n, G_i) := \min_{g \in G_i} d_{x,g}.$$

A comparison of GCP with GNP using delay data provided by RIPE NCC TTM [ttm08] has been presented in [KZ04]. As a summary we state that GCP shows significant better results e.g. with regard to average error of the embedding step. However GNP was a pioneering approach in the area of landmark based distance estimation and after the publication of the original GNP paper [NZ02] several other solutions followed with the goal to improve the accuracy of the actual network embedding. To have more relevant insights into the performance of GCP, we compare if with state of the art network distance estimation schemes in the next section.

6.3 Comparison of GCP with State of The Art Schemes

Table 6.1 shows a comparison of the most relevant schemes proposed for network delay estimation. To allow a first comparison of these schemes we use the following metrics:

- *Measurement overhead:* The number of measurements needed to realise the basis for the delay estimation.
- *Prerequisites:* Any special requirements for implementing and/or using the scheme.
- *Churn recovery:* Does the scheme include churn recovery strategies or not.

Scheme	Measurements overhead	Prerequisites	Churn recovery
GNP	$O(L^2 + L \cdot H)$	A set of landmarks	No
PIC	$O(L^2 + L \cdot H)$	P2P substrate	No
GCP	$O(L^2 + L \cdot H + C_s \cdot H)$	P2P substrate	No
ICS	$O(L^2 + L \cdot H)$	A set of landmarks	No
Lighthouses	$O(L^2 + L \cdot H)$	Frame of reference	No
Vivaldi	-	Traffic between nodes	Yes
Meridian	$O(N \cdot m^2)(GI) + O(log^2 N)(RI)$	-	Yes
Netvigator	$O(L \cdot H + R \cdot H)$	Routers support for traceroute	No

Table 6.1: Characteristics of State of The Art Delay Prediction Schemes

In Table 6.1 the following abbreviations are used: L corresponds to the number of landmarks used by a scheme and H to the total number of participating hosts. In the case of Meridian, N denotes the number of nodes, m the number of rings per node, GI is the overhead per gossip interval, and RI is the overhead per ring management interval. For GCP, C_s denotes the cluster size. For Netvigator, R denotes the number of milestones (routers) in use. No information is provided about the measurement overhead for Vivaldi since it is not using active measurements, but only piggyback latency information to the application traffic.

Based on above mentioned criteria, we selected Vivaldi [DCKM04] and PIC [CCRK04] as most appropriate schemes for a direct comparison with GCP because of the following reasons:

- None of the schemes requires explicit support from the underlying network infrastructure.
- There is no need for fixed landmarks, thus all candidates can work in a fully decentralized manner.

- The candidates exhibit low overhead when compared with other approaches like Meridian.

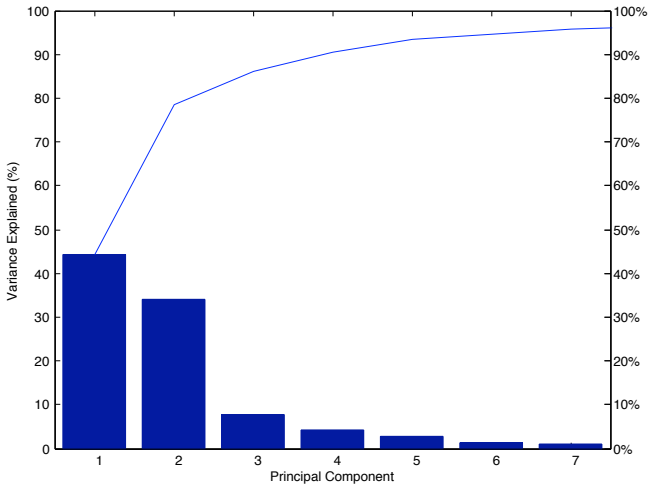
- The candidates are suitable to be used directly or with less customisation in a DHT context, compared to for instance Meridian, which is orthogonal to DHT-like structures.

For a comparative study of the selected schemes, a simulation based study has been performed as presented below.

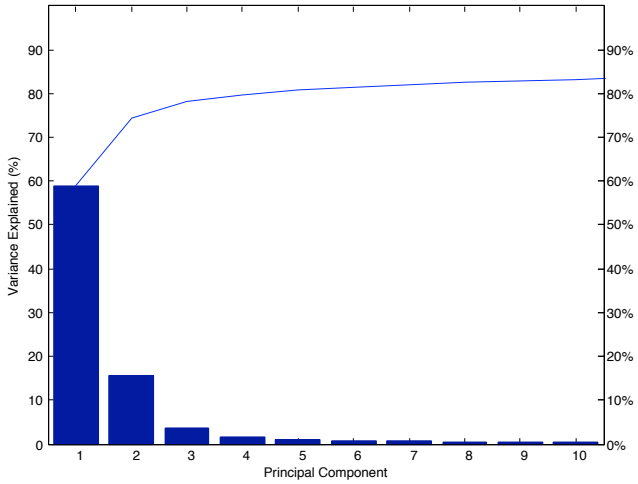
6.3.1 Performed Simulations

A modular packet level simulator has been developed to examine the selected delay predication candidates. We have used two data sets for this study. The first data set is the publicly available P2PSim King [p2p06] data set, which represents real internet measurements collected using the King method [GSG02]. This data set is advantageous because of the network and geographical diversity. We have also found that this data set has a very low number (i.e., 0.84%) of missing measurement pairs and a low number (i.e., 4.1%) of the triples violating the triangle inequality. A detailed description and analysis of this data set and its generation process has been provided in section 2.5. The second data set is based on a synthetic transit-stub topology generated by using the popular GT-ITM topology generator [ZCB96a]. The transit-stub topology model focuses on reproducing the hierarchical structure of the topology of the internet. To generate an end-to-end delay matrix out of this topology, we applied shortest path routing using the link weights as generated by GT-ITM. The GT-ITM parameters used for the generation of the topology are provided in section 2.5.

To determinate the optimal dimensionality of the target metric space for our simulations we performed a Principal Component Analysis of the P2PSim King and the GT-ITM based delay data. A scree plot to identify the number of the eigenvalues with significant magnitudes is shown below. The x axis in figure 6.9 corresponds to the eigenvalues in decreasing order. That is 1 corresponds to the largest eigenvalue, 2 to the second largest and so on. At



(a) Scree plot GT-ITM



(b) Scree plot P2PSim

Figure 6.9: Scree Plots

the y axis the eigenvalue magnitude normalised with respect to the greatest eigenvalue magnitude is shown. From the scree plot we find no significant difference for both topologies between eigenvalue magnitudes after the eighth eigenvalue. This suggests using 8 dimensions for the target metric space which coincides with the results in [CCRK04]. For the simulations performed,

we configured each scheme using the optimal parameters as described in the corresponding publications. More concrete:

- **Vivaldi:** For each Vivaldi Node we assigned 16 neighbours, where 4 are close neighbours with regard to delay and 12 are selected by random. Vivaldi nodes randomly exchange application level messages with their neighbours and we assume that the system converges after 32 iterations.
- **PIC:** For the landmark selection in PIC, we used the hybrid method as described in 3.2.3.2. After a PIC node has calculated its initial coordinates using random landmarks, this coordinate is refined measuring its distance to the 4 closest and 12 random neighbours.
- **GCP:** For GCP we simulated the following procedure using a cluster size of 5:

  1. When bootstrapping, a GCP node selects $d+1$ random landmarks for calculating initial coordinates.
  2. Once a GCP node finished the first step, it finds close nodes to it in the corresponding space of dimension d.
  3. After these neighbours are known, they form clusters, each with five nodes from immediate neighbours.
  4. As a final step, the lowest measurement result from each cluster is used to recalculate the node coordinates.

For PIC and GCP we further assume that the nodes are joining in an iterative but random manner.

6.3.1.1 Evaluation Framework

To establish the basis for a comparison we adopted metrics commonly used in the the area of network based distance estimation. A definition and description of these metrics is provided below:

- *Directional Relative Error (DRE):* This metric is an overall performance measure for the quality of embedding, which was introduced

in [NZ02]. It measures the magnitude of the deviation between the network distance before and after embedding for each pair of measurements. These values are then aggregated to characterise the system.

Definition 10. *Let x and y be two nodes in a network. If $d_{x,y}$ denotes the measured distance and $\hat{d}_{x,y}$ the estimated distance between x and y, the Directional Relative Error (DRE) of the estimation is given by:*

$$DRE(x,y) = \frac{|d_{x,y} - \hat{d}_{x,y}|}{\min(d_{x,y}, \hat{d}_{x,y})}$$

- *Stress:* Stress is an overall performance metric used to measure the quality of an embedding [CC01]. It measures the magnitude of the deviation between the distance before and after the embedding over all node pairs in the system. The stress is inherently an aggregate measure.

  Definition 11. *If for each pair of nodes x, y in a network N, $d_{x,y}$ denotes their measured distance and $\hat{d}_{x,y}$ their estimated distance, then the* stress *of the corresponding estimation is defined as:*

$$\frac{\sum_{x,y \in N} \left(\hat{d}_{x,y} - d_{x,y}\right)^2}{\sum_{x,y \in N} d_{x,y}^2}$$

- *Relative Rank Loss (RRL):* This metric was first introduced in [LGP+05] to evaluate an embedding from the perspective of relative ranking preservation. I.e. if node x has two neighbours y and z where y is closer to x than z, then RRL verifies if the relative closeness remains unaffected when mapping the three nodes to points in a low dimensional metric space.

  Definition 12. *Let*

$$R(x,y) := \begin{cases} -1 & if \quad x < y \\ 0 & if \quad x = y \\ 1 & if \quad x > y \end{cases}$$

If $d_{x,y}$ denotes the measured and $\hat{d}_{x,y}$ the estimated distance between two nodes x and y we say that x and y are swapped *with respect to a third node z if and only if*

$$R(d_{z,x}, d_{z,y}) \neq R(\hat{d}_{z,x}, \hat{d}_{z,y}).$$

The latter case is also denoted by swapped(z,x,y)*. If X denotes the set of all nodes in the network we further define*

$$P_z := \{(x, y) \mid x, y, z \in X,\ x \neq y \neq z,\ swapped(z, x, y)\}.$$

By using this notation, the RRL with regard to a node z can be defined as

$$RRL(z) = \frac{2 \cdot P_z}{(\|X\| - 1)(\|X\| - 2)},$$

where $\|X\|$ denotes the number of elements in the set X.

- *Closest Neighbor Loss Significance (CNLS):* This metric was initially introduced in [LGP+05] to check whether the closest neighbour in the network is the same as the one in the metric space after embedding. We use an improved version of CNLS to make it more expressive by incorporating the actual loss percentage. The result is more information about the significance of a closest neighbour loss.

  Definition 13. *Let X denote the set of nodes in a network. For an arbitrary node $x \in X$ let $y \in X$ denote the closest node to x in the network and $z \in X$ the closest node to x after embedding. The closest neighbor loss significance at x is given by:*

$$CNLS(x) = \frac{|d_{x,y} - d_{x,z}|}{d_{x,y}}$$

To be able to provide a statement about the suitability of QoS estimations for the CSP verify procedure we will introduce a further metric which has a direct relation to the CSP (least cost) constrained-based media processing problem introduced in section 4.4.1. To find a solution to such a problem,

CSP combines the verification of the QoS requirements of a multimedia flow with an optimisation with regard to costs as e.g. resource utilisation. In practice, the QoS constraints of mutlimedia flows correspond to thresholds e.g. with regard to the maximal acceptable communication delay for the multimedia session. If during a CSP search process node n needs to verify

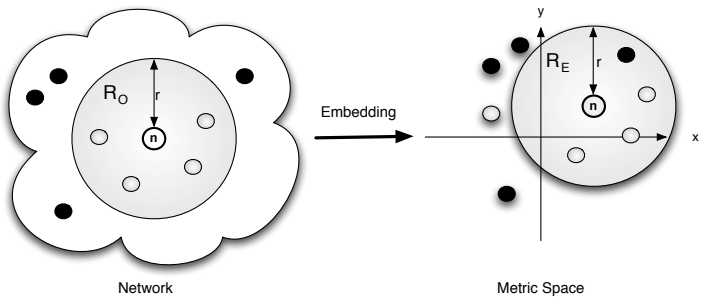


Figure 6.10: Threshold Verification Before and After Embedding

to which of the nodes providing a concrete service the delay is below r milliseconds. In case of figure 7.8 all such nodes are members of the set R_O. To analyse the impact of the described embedding process to such a threshold verification processes we use a metric called *Recal* [HS00b].

- *Recall:* The recall of an embedding is defined as follows. If R_E corresponds to the set of nodes having a distance r to n in the corresponding metric space after the embedding, then the recall of the embedding with regard to n and r is calculated as

$$recall_r(n) := \frac{\|R_E \cap R_O\|}{\|R_O\|}$$

  The outcome of this calculation corresponds to the number of correct results in R_E after the embedding process divided by the number of "correct" nodes in the network denoted by $\|R_O\|$. Thus a recall of 1 for all nodes in a network indicates that after the embedding process a threshold verification returns all correct results but possibly additional incorrect ones.

Based on the set of metrics defined in this section we state that a prediction scheme is suitable for CSP if it attains a high recall, i.e., close to 100% together with balanced values with regard to classical metrics as e.g. *relative error* and *stress*.

6.3.2 Simulation Results and Observations

We will now present an evaluation of the obtained simulation results based on the evaluation metrics as introduced above. All curves in this section show the cumulative distribution functions for the corresponding metric. For more results of this evaluation we refer to [Elm07].

6.3.2.1 Stress Comparison

Table 6.2 and 6.3 show the measured average stress values for the P2PSim and GT-ITM topology. While all schemes have similar stress values, for the P2PSim data, Vivaldi shows a slightly better performance than the other schemes. On the other hand, for the GT-ITM topology GCP has the

Scheme	**Vivaldi**	**Pic**	**GCP**
Stress	0.111645004	0.139883934	0.138243647

Table 6.2: Average Stress for P2PSim Data Set

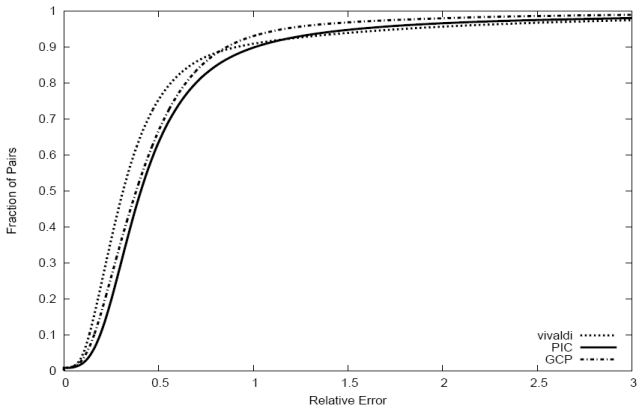
Scheme	**Vivaldi**	**Pic**	**GCP**
Stress	0.082234348	0.096939587	0.072605151

Table 6.3: Average Stress for GT-ITM Topology

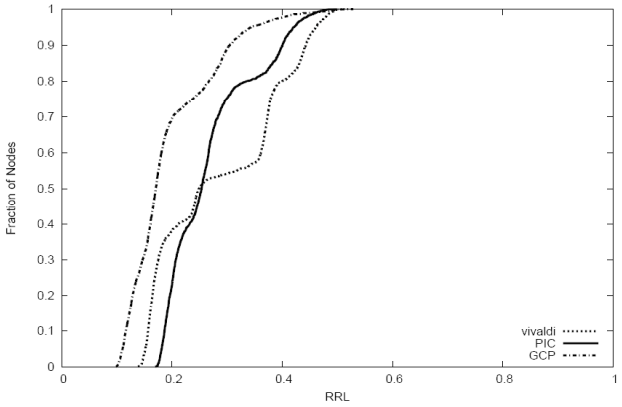
smallest average stress. We can further state that for the GT-ITM topology the stress related results are better. One reason for this is the fact that we simulated exclusively shortest path routing to generate this data. This guaranties that the triangle inequality with regard to delay always holds. For more information about the relation between triangle inequality and shortest path routing with regard to additive metrics we refer to section 3.2.5.

6.3.2.2 Directional Relative Error and Relative Rank Loss

Figure 6.11 and 6.12 show the obtained cumulative distribution functions for directional relative error and relative rank loss metrics. For DRE, all the schemes show comparable performance for the P2PSim data. On the other hand, for the GT-ITM data, GCP outperforms the other candidates with a slight margin. As for stress, candidate performance is better in case of the GT-ITM topology as one may expect. With regard to the RRL CDF plots



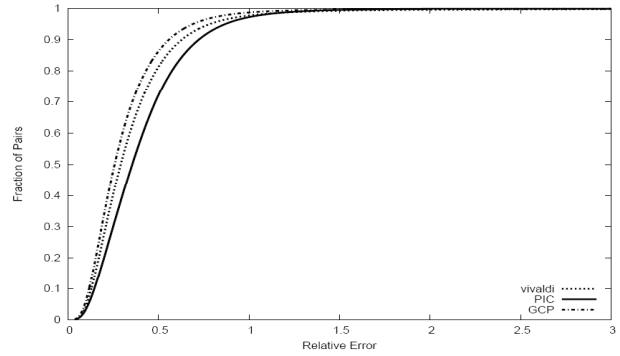
(a) DRE distribution



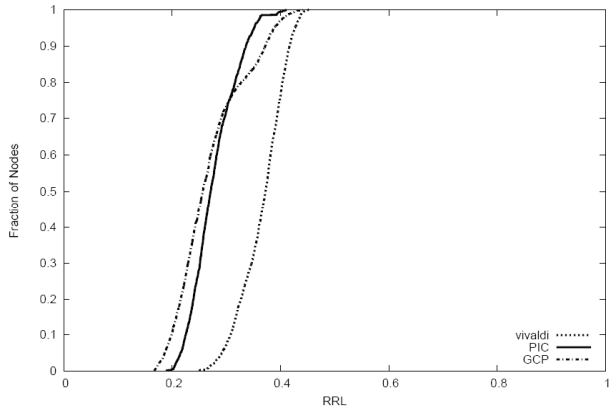
(b) Local RRL distribution

Figure 6.11: DRE and RRL Results for P2PSim Data

we can state that GCP exhibits better performance compared to the other two candidates.



(a) DRE distribution



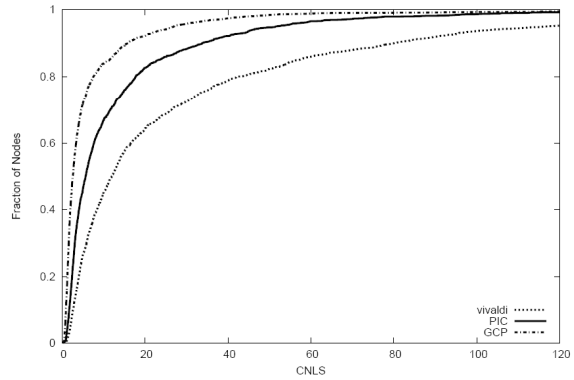
(b) Local RRL distribution

Figure 6.12: DRE and RRL Results for GT-ITM Topology

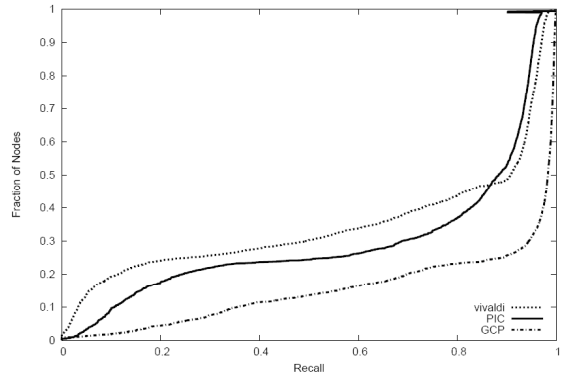
6.3.2.3 Closest Neighbour Loss Significance and Recall

With regard to the CDF for the CNLS we state that GCP shows significantly better results than PIC and Vivaldi for the P2PSim topology. In case of the GT-ITM data the situation changes. As we see in figure 6.15(a) PIC shows excellent results outperforming GCP and Vivaldi.

To study the characteristics of the evaluated schemes with regard to threshold queries we partitioned the network around each node n into three sectors. The first subset contains all the nodes with a network delay smaller or equal to $R1$ to n. The second subset contains all nodes with delay values greater than $R1$ and smaller than $R2$, while the third subset contains all



(a) CNLS distribution



(b) Recall distribution for the first range

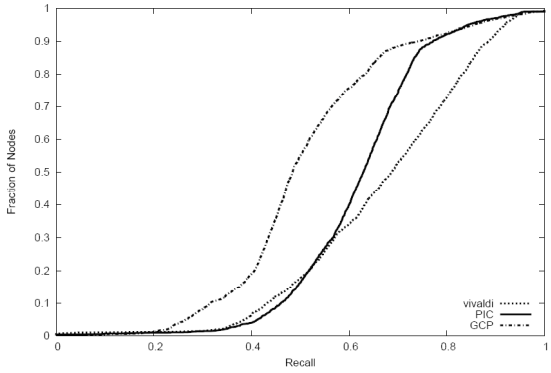
Figure 6.13: CNLS and Recall Range 1 for P2PSim Data

nodes with a delay greater than $R2$. Based on the different delay distributions of the P2PSim and GT-ITM data we have selected different values for $R1$, $R2$ as shown in Table 6.4 below.

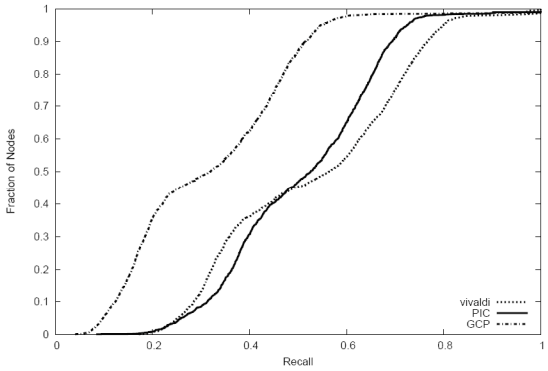
Ranges	**P2PSim data**	**GT-ITM**
R1	150 ms	200 units
R2	300 ms	400 units

Table 6.4: Recall Ranges

With regard to the CDF plots for the P2PSim data set we can make the following statements: As one can see in figure 6.13(b), GCP shows the best



(a) Recall distribution for the second range



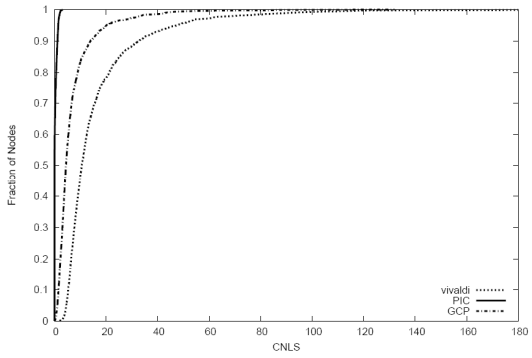
(b) Recall distribution for the third range

Figure 6.14: CNLS and Recall Range 2 & 3 for P2PSim Data

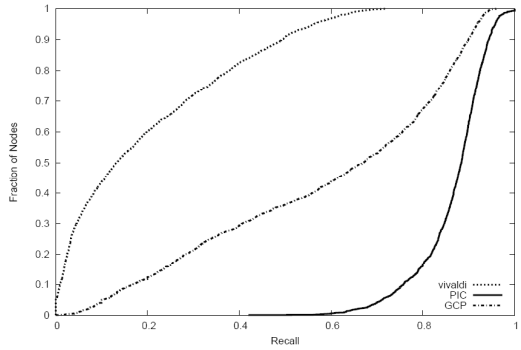
results for the first recall range, it outperforms the other two candidates with a significant margin. For the other two ranges shown in Figure 6.14(a) and 6.14(b) PIC and Vivaldi show a better behaviour. For the GT-ITM data shown, we note that PIC dominates for range one and three as shown in Figures 6.15(b) and 6.16(b). On the other, hand GCP shows superior results for range two as one can see in 6.16(a).

6.3.3 Summary

As a summary, we can provide the following set of concluding remarks and recommendations regarding the suitability of the selected schemes for accurate delay prediction. A general observation is that the embedding is more



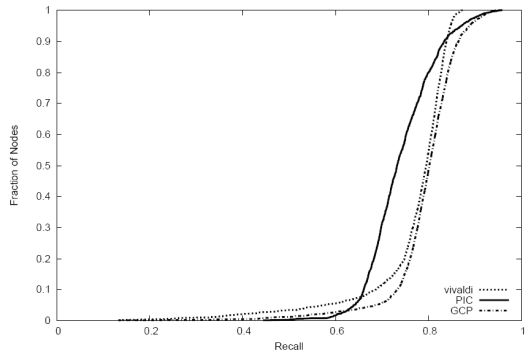
(a) CNLS Distribution



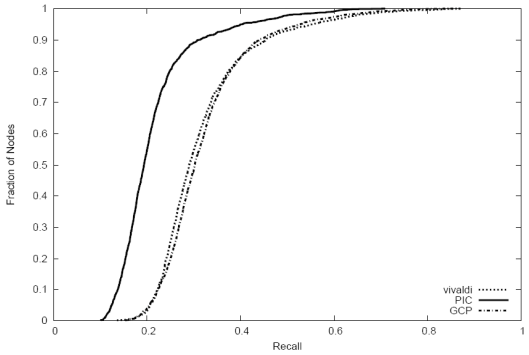
(b) Recall Distribution for the First Range

Figure 6.15: CNLS and Recall Range 1 for GT-ITM Topology

accurate for short distances than for medium and long distances. Based on the obtained results we can state that in our setting the landmark-based distance estimation techniques PIC and GCP performed better than Vivaldi. Comparing the results obtained when using the GT-ITM topology and the P2PSim data it is evident that Vivaldi and PIC tend to be more sensitive to triangle inequality violations than GCP. The selected schemes have been observed to perform poorly with regard to metrics like RRL and CNLS, especially when using the P2PSim King data set. An important aspect from the CSP perspective is the observation that the landmark based schemes have a better recall when considering short range queries.



(a) Recall Distribution for the Second Range



(b) Recall Distribution for the Third Range

Figure 6.16: Recall Range 2 & 3 for GT-ITM Topology

6.4 Improving Accuracy

Based on the results collected above, we will focus in this section on improving the prediction accuracy for medium and long network distances while conserving the comparably accurate predictions for short distances. To realise this, we combine schemes for landmark based distance estimation as e.g. GCP with clustering and a triangulation heuristic. Since this approach is independent of GCP, we introduce an own name for it and call it *Netforecast* [EKP07].

6.4.1 Netforecast

The main idea behind Netforecast is to group network nodes into circular clusters with an equal delay-based diameter as shown in figure 6.17. Inside each cluster, Netforcast applies independently a landmark-based distance estimation technique. As the result we have one local and non-exportable coordinate system for each cluster. To estimate delay between two nodes in the same cluster, the corresponding coordinate system is used. To predict the distance between two nodes form different clusters we apply a triangulation heuristic first used by Hotz [Hot04] in this context. As a result, we avoid using landmark-based distance estimation for predicting medium and long network distances and, as we will see later, the triangulation approach works well for medium and large distances.

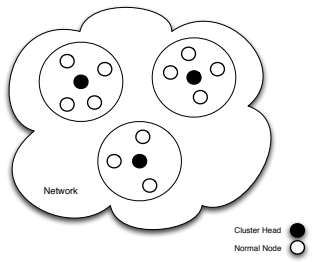


Figure 6.17: Netforecast Clusters

To realise this combination of two prediction schemes Netforecast is based on a hierarchical approach, where each node is characterized by two different coordinates: One for predicting short distances and the other one for predicting medium and long distances. Details of the Netforecast operation are as follows.

6.4.1.1 System Initialisation

Netforecast has been developed for predicting network delay in provider controlled environments, thus we assume the availability of information about the underlying network, e.g., network map, node distribution characteristics. A network administration entity is supposed to initially select a set of nodes

to be used as cluster heads. For this procedure the delay distribution in the network and the distribution of the network nodes are taken into consideration. Furthermore, the selected cluster heads are expected to be highly reliable and connected to the network via high capacity links.

6.4.1.2 Cluster Characteristics

Netforecast assumes clusters with the following characteristics:

- An unique $Cluster\_ID$ is assigned to the cluster head, either administratively or using a uniform hash function.
- Cluster membership is mutually exclusive.
- All clusters are circles with the same radius in terms of network delay.
- A cluster head allows an arbitrary node to join its cluster if the delay between itself and the node is less than or equal the cluster radius.

6.4.1.3 Node Clustering

A newly joining node receives upon bootstrapping a sorted list of cluster heads as well as the value for the cluster radius in the network. It then measures the RTT to each cluster head and arranges them in a specific tuple characterizing its location in the network with respect to the cluster heads. For example, this tuple can be expressed as $(d_{L_1}, d_{L_2}, ..., d_{L_n})$, where each element reports the measured RTT to the respective cluster head $L_1, ..., L_n$. Furthermore, as a final step the node seeks to join one of the existing clusters by following the steps listed below:

- Identification of the closest cluster head.
- Check if the distance between itself and the closest cluster head is less than or equal to the system clusters radius value.
- If the previous check result is true, it sends a cluster join request to the closest cluster head or, alternatively, it sets the $Cluster\_ID$ to -1 so denoting no cluster membership.

- Upon receiving a cluster join request, any cluster head may further check to assure the node eligibility to join its cluster. This is done by measuring its distance with the node. After verifying the eligibility of the requesting node, the respective cluster head sends back a join acceptance along with its $Cluster\_ID$ (which is used by the node as its new $Cluster\_ID$).

6.4.1.4 Network Distance Estimation

As stated before Netforecast estimates inter-cluster and intra-cluster distances using different approaches. [2]

1. *Intra-Clusters Distance Estimation:* NetForecast applies a suitable landmarks based distance estimation technique locally and independently at each cluster. Furthermore, different clusters have different non-related coordinates systems.

2. *Inter-Clusters Distance Estimation:* The concept of triangulated heuristics [Hot04] is used for the estimation of the network distance between any two nodes belonging to different clusters. The main reason for this is that we assume that at least one of the cluster heads is in the shortest path between any two nodes A and B. Assume for instance that we want to estimate the distance between any two nodes A and B, which are members in different clusters. A is characterized by the tuple $(d_{AL_1}, d_{AL_2}, ..., d_{AL_n})$, which represent its measurements to the cluster heads. B is characterized by the tuple $(d_{BL_1}, d_{BL_2}, ..., d_{BL_n})$, which represent its measurements to the cluster heads. The distance between A and B is then estimated by

$$min_{k \in 1,2,..,n}(d_{AL_k} + d_{BL_k}).$$

[2] Intra-cluster distances denote the distances between nodes within the same cluster. Inter-clusters distances denote distances between members of different clusters.

6.4.2 Performance

We have simulated Netforecast by using the same simulation environment and the same data sets as used in the comparative study presented in section 6.3. For each simulation run we selected ten cluster heads following the requirements as mentioned above. Based on the results obtained from the comparative study, we set the cluster radius to 150 ms in the P2PSim King data set and to 250 units for the GT-ITM topology. These values correspond to distances where landmark based distance estimation techniques have been observed to achieve a high recall. GCP is used for the intra-cluster coordinates system while experimenting with P2PSim King data set, PIC is used for the GT-ITM topology experiments. As a general observation, all schemes achieved better performance when using the GT-ITM topology. In the following we therefore report the results obtained with the P2PSim King data set only. Netforecast has been simulated and the performance compared with Vivaldi, PIC and GCP with regard to the following parameters:

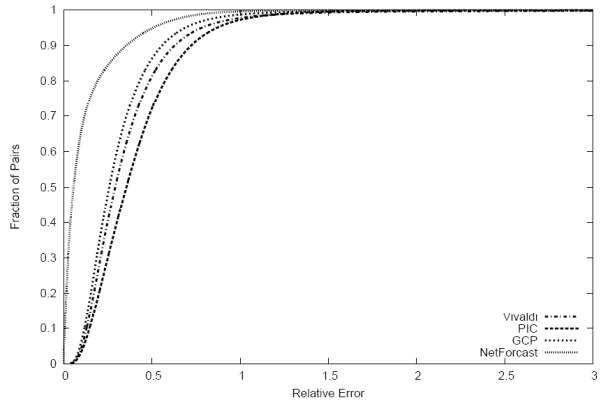
- Stress, reported in the form of the average stress measured over the whole data set (table 6.5).
- Directional Relative Error, reported in the form of a CDF (figure 6.18(a)).
- Local Relative Rank Loss, reported in the form of a CDF (figure 6.22(b)).
- Closest Neighbor Loss Significance, reported in the form of a CDF (figure 6.19(a)).
- Recall, reported in the form of a CDF. Figure 6.19(b) shows the recall CDF in the first range, figure 6.20(a) shows the recall CDF in the second range and figure 6.20(b) shows the recall CDF in the third range.

It is observed that Netforecast performs better than the other three candidates with reference to stress, DRE, RRL and CNLS. Furthermore, we also observe that Netforecast shows better recall performance than other schemes in the first range, for approximately 10 percent of the nodes. This is followed by a comparable performance with PIC and Vivaldi but less than GCP. On

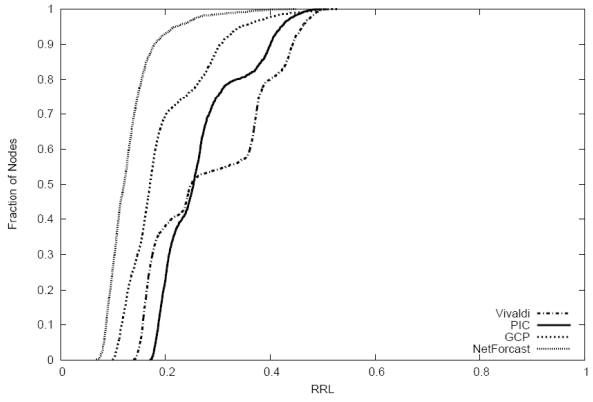
the other hand, it is observed that Netforecast outperforms the other three schemes with a considerable recall margin in the second and third range.

Scheme	**Vivaldi**	**Pic**	**GCP**	**NetForecast**
Stress	0.111645004	0.139883934	0.138243647	0.049862

Table 6.5: Average Stress

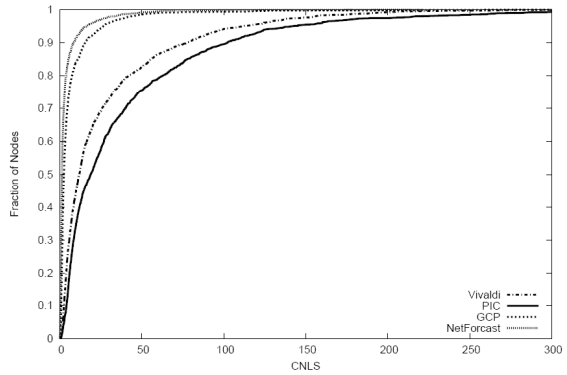


(a) DRE distribution

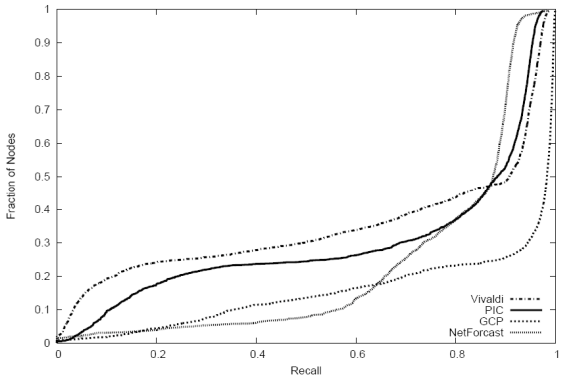


(b) Local RRL distribution

Figure 6.18: Results for DRE and LRE Metrics



(a) CNLS Distribution

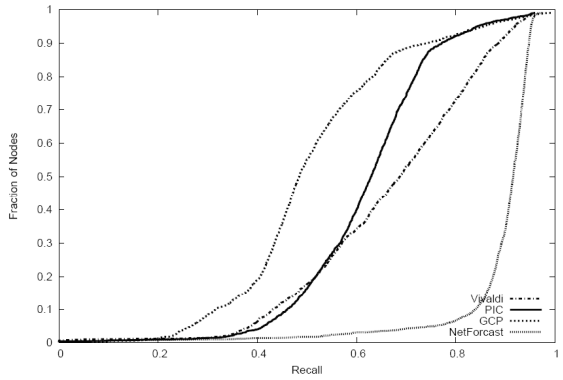


(b) Recall Distribution for the First Range

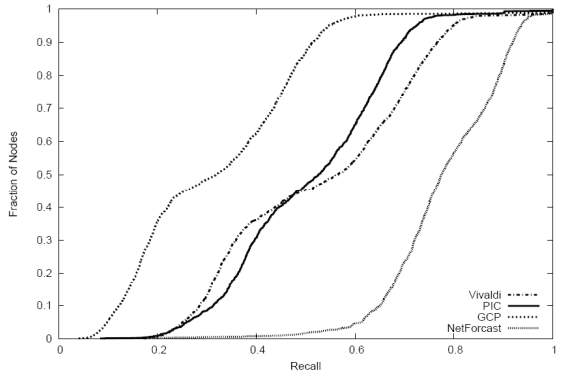
Figure 6.19: CNLS and Recall Range 1

6.4.3 Summary

Based on these results, we can state that the combination between clustering, landmark based distance estimation for short distances and the triangulation heuristic introduced by Hotz for medium and large distances definitely improves delay prediction. This statement is supported by the fact that, in most of cases, all evaluation metrics show better values for Netforecast than for the other schemes Vivaldi, PIC and GCP. It can be further deduced that Netforecast shows better results in case of triangle inequality violations than the other schemes evaluted. The best results with regard to performance are obtained if shortest path routing is used with regard to delay. Netforecast



(a) Recall Distribution for the Second Range



(b) Recall Distribution for the Third Range

Figure 6.20: Recall Range 2 & 3

depends however on several heuristics, with the consequence that we expect differences in performance from one network to another, depending upon the specific conditions. A set of guidelines for the implementation of Netforecast is presented in [Elm07].

6.5 Bandwidth Prediction

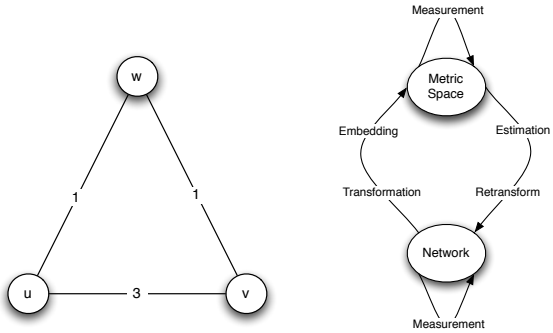
As motivated in section 4.4 of chapter 4, bottleneck bandwidth is a second network QoS metric we consider for CSP. In this section we will now evaluate if the delay estimation schemes studied in the last section can be generalised to bandwidth estimation as well. To approach this, we first analyse the

differences between delay and bottleneck bandwidth as metrics and later the properties of the bottleneck bandwidth metric in case of widest path routing.

In case of using a bandwidth related measure as e.g. *throughput* defined as *the amount of data delivered per time unit* the situation is different with regard to the triangle inequality. Throughput is a *concave* metric [RS07] and simply enforcing widest path (i.e. maximum throughput) routing inside a network does not guarantee the triangle inequality in general as one can see in figure 6.21(a). In this graph the maximum throughput between u and v is 3 and larger than 2 which is the sum of the maximum throughput between u and w and w and v i.e.

$$d(u, v) \geq d(u, w) + d(w, v)$$

and thus the triangle inequality does not hold. Fortunately it is possible to



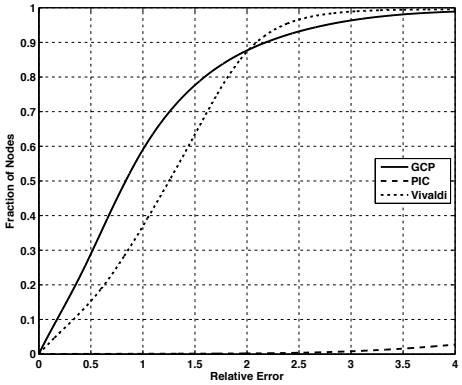
(a) Example Graph

(b) Transformation before Embedding and after Estimation

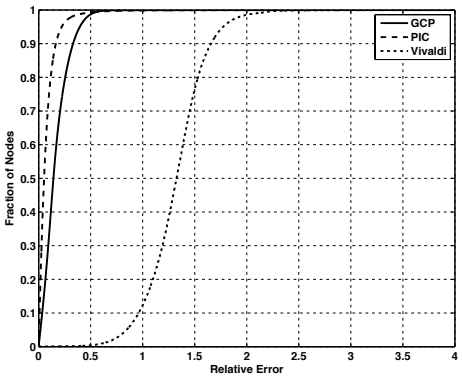
Figure 6.21: Example Graph and Transformation Principle

change this situation by using a transformation technique. The principle of this transformation is illustrated in figure 6.21(b). To explain the figure, let $\delta^t(n, l)$ denote the bottleneck bandwidth between a node n and a landmark l. To calculate the coordinates of n using a suitable landmark based distance estimation scheme we use the value $\frac{1}{1+\delta^t(n,l)}$. That is, after the measurements we apply the transformation function $f(x) = \frac{1}{1+x}$. In case we want to estimate the throughput between any two nodes u, v in the network using

their corresponding coordinates $\hat{x}, \hat{y}$ in the metric space (X, d) we calculate $\frac{1}{d(\hat{x},\hat{y})} - 1$. That is we apply the retransformation function $g(x) = \frac{1}{x} - 1$. As simple as this transformation may be as impressive it's impact to the accuracy of bottleneck bandwidth prediction. Figure 6.22 shows the obtained cumulative distribution functions for directional relative error in case of the GT-ITM topology with simulated widest path routing. As it is easy to see,



(a) DRE no transformation



(b) DRE with transformation

Figure 6.22: *DRE Results for GT-ITM Data*

the transformation significantly improves the accuracy of the prediction. E.g. for GCP the achieved improvement of average DRE is in the range of 85 percent, for PIC the improvement is even more drastic. In fact after applying the prediction technique, the quality of the estimation with regard to the DRE

metric is comparable to the delay related results. In the following we shown why the described projection technique works well in case of widest path routing by enforcing the triangle inequality. We start by recalling required notations as introduced in chapter 4 section 4.4. Let $G = G(V, E)$ denote a connected, weighted and undirected graph with positive edge weights where V corresponds to the set of nodes in G and E the set of edges between the nodes. A path $p = ((v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n))$ in G is written as the sequence of the edges $e \in E$ it includes and the function length(p) returns the number of edges. In case p is connecting $u, v \in V$ we write $u \overrightarrow{p} v$. Given a weight function $w : E \mapsto \mathbb{R}^+$ the measure for throughput of a path p in G is defined as

$$t(p) := \min_{i=1,\ldots,\text{length}(p)} w((v_i, v_{i+1})).$$

Assuming routing with regard to maximum throughput, we can calculate the bottleneck bandwidth between two arbitrary points in V as

$$\delta^t(u, v) := \begin{cases} \max\{t(p) | u \overrightarrow{p} v\} & , u \neq v \\ \infty & , u = v \end{cases}$$

The main reason for the fact that δ^t is not satisfying the triangle inequality is that $t(p)$ is by definition monotonic decreasing with regard to the path length. To overcome this, we modify the definition of $t(p)$ towards

$$t^*(p) := \max_{i=1,\ldots,\text{length}(p)} \frac{1}{1 + w((v_i, v_{i+1}))}.$$

That is, we apply the monotonic decreasing function $f(x) = \frac{1}{x+1}$ and use the fact that the composition $f \circ t$ of the two monotonic decreasing functions f and t is monotonic increasing. Further we define a corresponding measure δ^* as

$$\delta^*(u, v) := \begin{cases} \min\{t^*(p) | u \overrightarrow{p} v\} & , u \neq v \\ 0 & , u = v \end{cases}$$

Using this definition one has

$$\begin{aligned}
\delta^*(u,v) &= \min_{u\overrightarrow{p}v} \max_{i=1,\ldots,\text{length}(p)} \frac{1}{1+w((v_i,v_{i+1}))} \\
&= \min_{u\overrightarrow{p}v} \frac{1}{1+\min_{i=1,\ldots,\text{length}(p)} w((v_i,v_{i+1}))} \\
&= \frac{1}{1+\max_{u\overrightarrow{p}v} \min_{i=1,\ldots,\text{length}(p)} w((v_i,v_{i+1}))} \\
&= \frac{1}{1+\delta^t(u,v)}
\end{aligned}$$

Based on this result we can provide the following two statements about the measure δ^*:

1. A one-to-one transformation between measurements with regard to δ^t and δ^* is possible using above described transformation.

2. A Network where exclusively non asymmetric routing with regard to maximum throughput is used, equipped with the measure δ^* can be identified with a metric space.

A combination of these two properties allows us to use the described transformation technique for landmark-based throughput estimation as described in the beginning of this section. To summarize the described results we conclude this section with a proof of the following lemma.

Lemma 5. *A connected Network where exclusively non asymmetric routing with regard to maximum throughput is used equipped with the metric δ^* can be identified with a metric space.*

Proof. To proof the lemma we have to show that properties (1), (2) and (3) from definition 9 hold. Property (1) follows from the definition of δ^*. (2) holds because we require symmetric routing. To get (3), let u, v, w be three arbitrary nodes in the network. If p_1, p_2 are two paths having the property $u\overrightarrow{p_1}w$ and $w\overrightarrow{p_2}v$ both offering maximum possible throughput and

$p_1 \circ p_2$ denotes their concatenation it follows that:

$$\begin{aligned}
\delta^*(u,v) &= \min_{u \overrightarrow{p} v} t^*(p) \\
&\leq t^*(p_1 \circ p_2) = \max_{i=1,\ldots,\text{length}(p_1 \circ p_2)} \frac{1}{1 + w(v_i, v_{i+1})} = \max\{t^*(p_1), t^*(p_2)\} \\
&\leq \delta^*(u,w) + \delta^*(w,v)
\end{aligned}$$

which gives (3). □

Chapter 7

Evaluation

In this chapter we focus on the evaluation of the proposed CSP system. In contrast to the already presented dedicated simulation results for QoS estimation approaches, the main focus of this chapter is on the evaluation of search aspects and the impact of QoS estimation schemes on the CSP verify operation.

7.1 Setup of Experiments

For the aspired evaluation it is required to define a framework covering all the different layers addressed by CSP. In fact, we need to select parameters related to the following four parameter spaces:

1. *Service Specific Parameters*
2. *Search Specific Parameters*
3. *Verify Specific Parameters*
4. *Network Specific Parameters*

In the following we describe each parameter space and motivate the parameters selected for the performed experiments.

7.1.1 Service Specific Parameters

As shown in chapter 4 the number of available and valid Processing Chains that can be used to instantiate a composed service depends heavily on the number and kind of registered processing modules and their associated service graph. As a consequence, important parameters with regard to CSP search scope are processing chain length, the number of available PMs as well as the distribution of their I, P, O values. Based on practical considerations, we only treat service chains up to a length of ten steps. For the actual simulation of PMs we have chosen natural numbers to represent different I, O formats or processing functions P. In addition, we assume that there are ten different I, O and P values which are of equal popularity. How the actual values are chosen is described in advance of each performed simulation.

7.1.2 Search Specific Parameters

For the performed experiments we used a CAN DHT supporting the range query algorithm introduced in chapter 5. To evaluate different search principles used in conjunction with the DHT based approach to CBMP described in the same chapter a simulator has been developed which includes:

- A CAN simulation.
- An interface to implement different search schemes.
- A verify component.
- An interface to integrate network data.

A schematic overview of the simulator is provided in figure 7.1

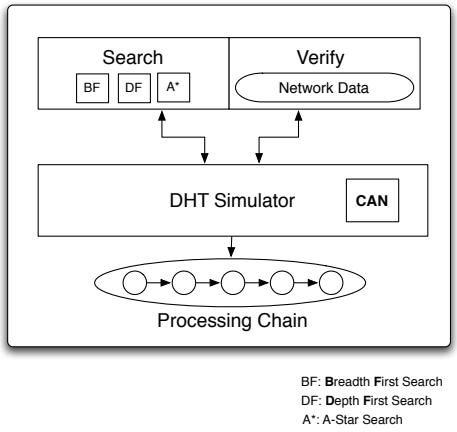


Figure 7.1: Simulator for Search Experiments, Schematic View

The initial CAN address space is represented by the cube $[0, 255]^3 \subset \mathbb{R}^3$ and the CAN DHT simulation supports the range query algorithm introduced in section 5.2.2. The three DHT primitives *Join, Register* and *Search* are realised in the simulation as follows:

Join By calling the join function, a CAN (overlay) address for a node about to join is calculated. Based on this addresses the CAN space is dynamically partitioned as described in section 2.3.1. After the join procedure each node

has two addresses in the simulation: A CAN overlay address in the form of a triple (x, y, z) with $x, y, z \in [0, 255]$ and a network transport address t determined on the basis of the provided network data.

Register After the join procedure, the new member n calculates for a PM it is hosting

$$hash_{CAN}(PM) := (hash(I), hash(P), hash(O)),$$

where $hash$ is a uniform hash function with values in $[0, 255] \subset \mathbb{R}$. The value of $hash_{CAN}(PM)$ is used to identify the CAN node, responsible for the CAN territory the point $hash_{CAN}(PM)$ belongs to. At this node we store information of the form: *Processing capability* (I, P, O) *is available at node* n *with network transport address* t.

Search The CAN routing algorithm used for the standard CAN search function (c.f. section 2.3.1) was greedy routing. We have implemented three different search function which are a BF (c.f. section 5.2.3) a DF and a DF principle running multiple concurrent branches (c.f. section 5.2.4). The used forward checking and backtracking functions target at lowest cost of a Processing Chain in a greedy manner. To realise the DF search with k concurrent branches, the node starting the search process is initiating k independent DF search branches.

7.1.3 Network Specific Parameters

For the required network simulation a packet-based, discrete event simulator has been developed similar to the one described in [CJK+03]. For the performed experiments we used a GT-ITM generated topology (c.f. section 2.5.2) since we consider delay and bottleneck bandwidth as metrics during this chapter. The corresponding delay and bottleneck bandwidth values have been calculated based on the GT-ITM generated link weights simulating shortest and widest path routing respectively.

7.2 Comparison of Different Search Principles

In chapter 5 we introduced a DF and a BF CSP search principle. In this section we perform the corresponding simulation based study. Target is an evaluation and comparison of the different search strategies with regard to:

Scope Scope is a common metric in P2P related research used e.g. to express the (average) number of nodes involved in a search request. For the used CAN DHT scope values are well known, and our main focus in this chapter is on the impact of required CSP DHT extensions.

Response Time An indicator for the response time of the resulting system.

Number of Solutions Following the results of the complexity analysis performed in section 4.5 the number of available Processing Chains can grow exponential with regard to the number of processing steps. For the performed simulations we are interested to verify this fact as well as to evaluate how many disjoint solutions are returned. As disjoint solutions we denote Processing Chains having no PM in common.

We will see that for short Processing Chains (e.g. up to four processing steps) a BF-based search is a promising option. With regard to a CSP specific scope measure approximately 15 percent of CSP nodes in the system have been involved in a search process. In addition the response time is close to optimal since no backtracking is required and all possible solutions are returned. For longer chains (from four to eight processing steps) a DF-based search with two or three concurrent branches has shown the most promising results.

In the following sections we provide a detailed description of the performed simulations and obtained results.

7.2.1 Performed Simulations

For each simulation run we randomly selected a subset of 500 vertexes as CSP nodes from the 1740 available vertexes of the used GT-ITM topology. Each

node hosted one PM with independent I, P, O values, randomly selected out of the set $\{1, ..., 10\}$ respectively. In addition a corresponding cost value for PM usage was selected randomly out of the set $\{1, ..., 100\}$. For each experiment performed, we selected a MS and a MC out of the set of available nodes. Now a Processing Chain of length $l = 3, \ldots, 10$ was selected out of the corresponding service graph and the corresponding constraint vector with regard to *Delay*, *Bandwidth* and *Cost* values was determined. After this initialisation step the following information was available:

1.	Service Source	MS
2.	Service Sink	MC
3.	Processing Chain Template	$(P_1, ..., P_{l-2})$
4.	CSP Nodes implementing P_is	$(N_1, ..., N_{l-2})$
5.	Delay Constraint	$D_{MAX} \leq delay_p(N_1, ..., N_l)$
6.	Bandwidth Constraint	$B_{MIN} \geq bandwidth_p(N_1, ..., N_l)$
7.	Cost Constraint	$C_{MAX} \leq cost_p(N_1, ..., N_l)$

We now used the information from point 1, 2, 3, 5, 6 and 7 to find all available Processing Chains based on the following search schemes.

1. *All Branches (breadth first):* Each PM found meeting the constraints continues the search and starts new branches in a BF manner.

2. *Single Branch (depth first):* Only the PM meeting the constraints with total lowest costs is used to continue the search. The same strategy is used for backtracking in case the search could not be continued because of constraint violations.

3. *K Branches:* Same as in the single branch case, but using K concurrent branches instead.

7.2.2 Evaluation Framework

The results obtained are evaluated based on the following metrics:

- **Active Nodes:** The number of nodes actively involved in a search process. A node is considered to be actively involved in case it triggers a search request. Nodes only involved in CAN routing are not counted.

- **Processing Chains returned:** The total number of Processing Chains (Paths) returned.
- **Disjoint Processing Chains returned:** The number of Processing Chains (Paths) returned not having any PM in common.
- **Min steps till response (hops):** The minimal number of search steps performed until a solution has been found, as a rough indicator for the time required to complete a request. In case of the *all branches* approach this is always $l - 1$, where l is the length of the Processing Chain.

For each chain length l we performed 100 experiments and show the averaged results.

7.2.3 Comparison of DF and BF Approach

We start with a comparison of BF and DF search principles for CSP. As shown in figure 7.2, CSP based on a DF search principle requires definitely less active nodes than a BF-based variant. Especially for long Processing Chains, the DF approach behaves superior with regard to the active nodes metric. For short Processing Chains with up to four processing steps, a BF search actively involves approximately 15 percent of available CSP nodes. We therefore consider it as an alternative, especially because the BF approach in parallel finds all Processing Chains available for the requested service. In case disjoint solutions are found it is possible to instantiated them concurrently for resilience. Figure 7.3 shows the number of disjoint paths returned by a BF search. As one can see in case of four processing steps between 3 and 3.5 disjoint Processing Chains are found in average. In contrast a DF-based CSP search returns only one Processing Chain candidate.

A BF approach further shows optimal results with regard to the hops metric as one can see in figure 7.4. The obvious reason for this is the fact that a BF search doesn't require any backtracking strategy. In contrast, the DF search requires backtracking which increases the hop count in case the current branch of the search violates the QoS constraints.

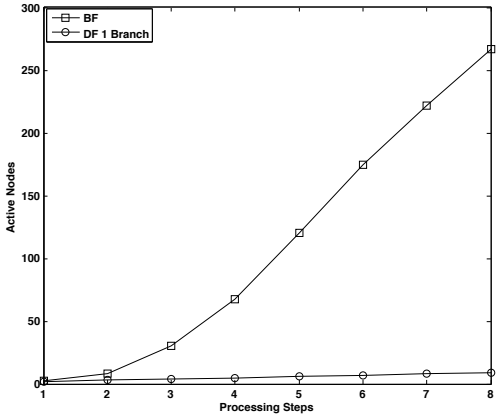


Figure 7.2: Active Nodes (BF DF)

As shown in figure 7.5 the results of our theoretical analysis from section 4.5 are confirmed by the simulation. In fact the number of returned solutions using a BF based CSP principle grows almost exponential with regard to the number of processing steps in a chain.

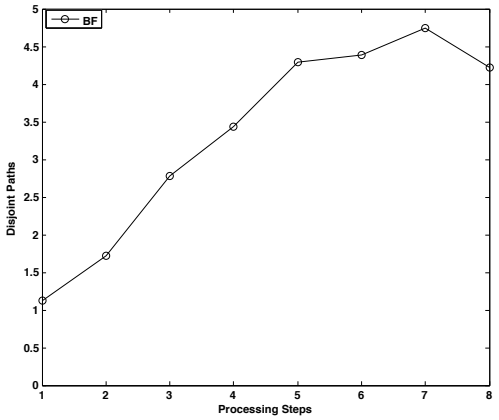


Figure 7.3: Disjoint Processing Chains (Paths) Returned (BF)

To summarise, as a general observation the BF approach is optimal with regard to hops (which translates to search time) and the number of returned solutions. In contrast, with regard to scope values, the DF search is close to optimal. While this result can somehow be anticipated we can state that a BF search can be considered as an interesting alternative in case of short

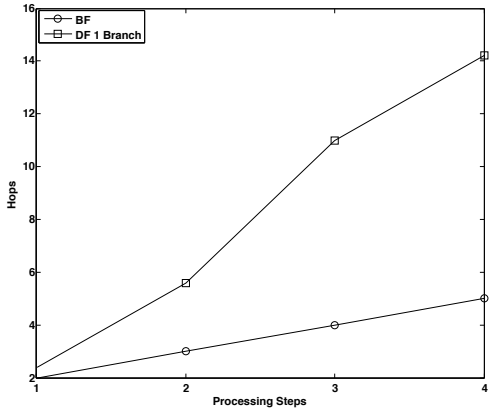


Figure 7.4: Min. Steps Till Response (BF DF)

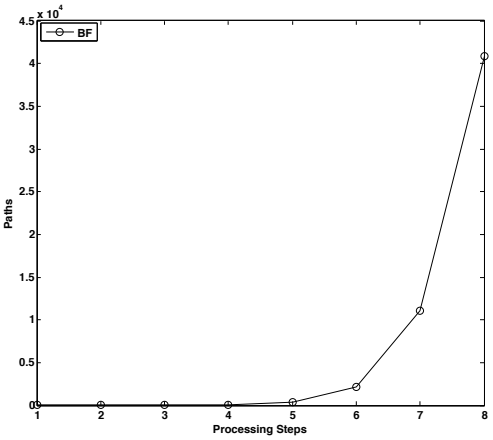


Figure 7.5: Processing Chains (Paths) Returned (BF)

processing chains. In the next section we try to combine the advantages of BF and DF search by evaluating a DF search principle allowing multiple concurrent branches.

7.2.4 DF Search with Multiple Branches

In the following we show the results obtained for a DF search based CSP with up to five concurrent search branches. Starting with figure 7.6, as somehow expected, one sees that the number of active nodes increases with the number of branches. Compared to the BF search in the last section, a DF search with

three concurrent branches involves less than 55 percent and a DF search with two concurrent branches less than 25 percent of active nodes.

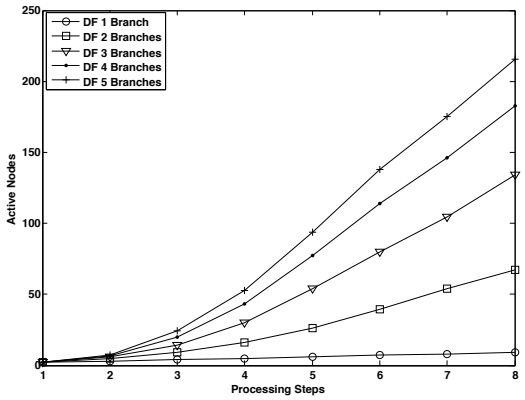


Figure 7.6: Active Nodes (DF)

With regard to the number of returned disjoint Processing Chains (c.f. figure 7.7) we see a similar effect. Keeping the scope values of a two and three branch approach in mind, we can state that the number of returned disjoint solutions is comparably high. In fact for long chains (i.e. eight processing steps), a three branches DF search returns already 60 percent of available disjoint Processing Chains.

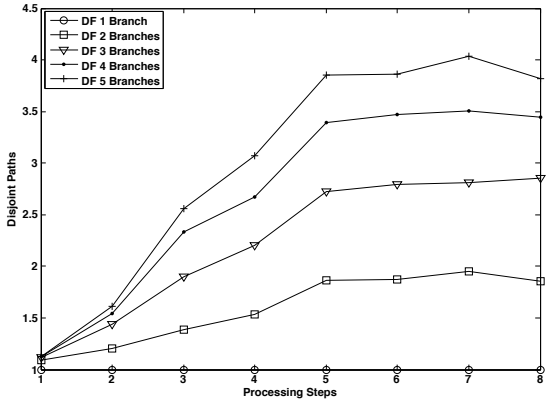


Figure 7.7: Disjoint Processing Chains (Paths) returned (DF)

7.2.5 Summary

In contrast to BF, the DF search principle is based on two functions for controlling the search process, namely a *Forward Checking* and a *Backtracking* function. Another significant difference between the DF and BF approach is that a standard DF will terminate after a predefined number or less valid Processing Chains are found, while the BF-based search will return all solutions in the corresponding Service Graph fulfilling the given QoS constraints. Important metrics to evaluate different search principles to be used for CSP are low search time (hops) and scope (active nodes). From the viewpoint of resilience, the scheme should return multiple disjoint Processing Chains. Based on the performed simulations we can state that for Processing Chains with a low number of processing steps a BF-based search has shown good results because of the low number of hops and the fact that all possible valid Processing Chains are returned. In case of long chains, a DF-based search with two or three concurrent branches shows the most promising results.

7.3 Impact of Delay and Bandwidth Prediction on CSP

In a second series of experiments, we studied the impact of estimation schemes to the CSP verify operation. To do this we analysed *recall* and *precision* values of an estimation based verify-process.

7.3.1 Performed Simulations

For the evaluation performed, we concentrated on an analysis of the impact of QoS prediction schemes. To do this, we focused on two core network scenarios which were:

1. A network where shortest path routing with regard to delay is used.

2. A network where widest path routing with regard to bottleneck bandwidth is used.

As a basis for the first scenario, we applied delay estimation and for the second scenario bottleneck bandwidth estimation. For the actual delay estimation we adopted the GCP scheme and for the bottleneck bandwidth estimation we used PIC combined with the projection technique introduced in section 6.5.

For the performed experiment we assigned to each node in the GT-ITM topology a processing module $PM = (I, P, O)$ with I, P, O values selected as *uniformly distributed random numbers* out of the set $\{1, 2, 3, 4, 5\}$. After this initialisation procedure, we applied CSP to search for processing chains with four up to eight processing steps. A single simulation run was performed as follows: In a first step, all possible composed services of a given length are collected using CSP in combination with a breadth first search. After this, the constraint vectors C_{Net} with regard to delay and bottleneck bandwidth were determined for each solution. In addition a vector with estimated constrains C_{Est} was determined using GCP for delay and PIC together with the developed transformation technique for bandwidth estimation. After this preparation, we determined with which precision and recall values we could have used QoS estimations to find the 10th to 90th percentile of the available composed services. To evaluate the results of the performed experiments we used the following evaluation framework.

7.3.2 Evaluation Framework

The QoS constraints related to service composition scenarios correspond to thresholds e.g. with regard to the maximal acceptable communication delay for the multimedia session. During a CSP search process for example, a node n needs to verify to which PMs the network delay is below r milliseconds. In case of figure 7.8, all such nodes are members of the set R_O. To analyse the impact of the described embedding process on such a threshold verification processes we used the recall metric as introduced in chapter 6. In addition, we define *precision* as the value:

$$precision_r(n) := \frac{\|R_E \cap R_O\|}{\|R_E\|}.$$

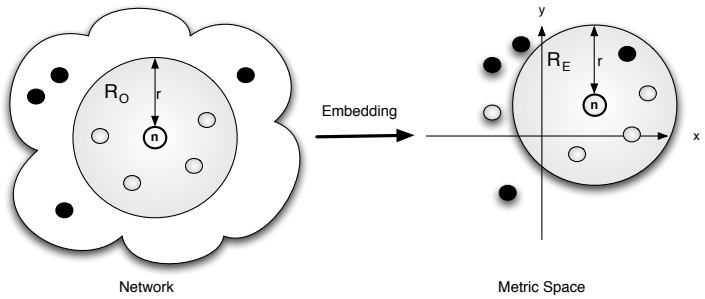


Figure 7.8: Threshold Verification Before and After Embedding

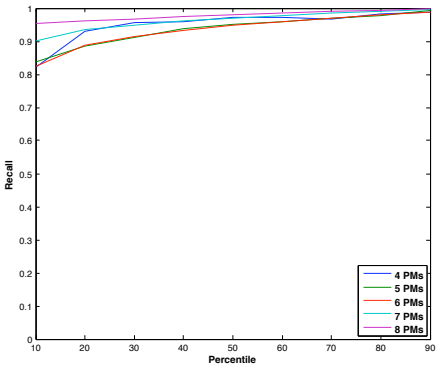
That is, a high precision corresponds to a low number of false positives in case of threshold verification. Using recall and precision as metrics we can state that a delay prediction scheme is suitable for CSP if it attains a high recall, i.e., close to 1 (100 percent) together with balanced values with regard to precision.

7.3.3 Simulation Results and Observations

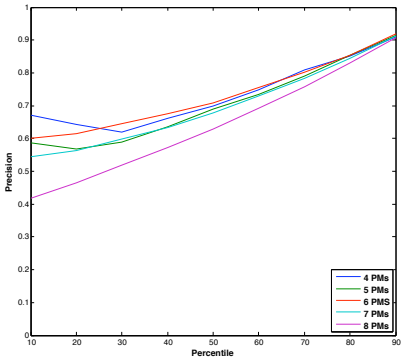
The averaged results of 100 simulation runs are shown in figure 7.9 and 7.10. Figure 7.9 shows that the performed delay estimation leads to high recall values in the range from 80 to 100 percent. With regard to precision we can see that even for Composed Services with stringent QoS constrains (i.e. from the 10th percentile) we have more than 50 percent precision up to a service length of 7 steps. For the performed bandwidth estimation, we see in figure 7.10 the recall values range from 55 to 95 percent. With regard to precision we have similar results as in the delay case.

Thus QoS estimation especially for delay has the potential to improve the response time of decentral Service Composition approaches as e.g. CSP. However, the most reasonable applications of QoS estimation may be in the context of filter and refine strategies, where with the help of estimation a set of Service Overlay candidates is determined and after this process, QoS measurements are used to select a suitable candidate.

As described in chapter 6 the error introduced by estimation schemes can be significant in some cases and we therefore do not consider estimation

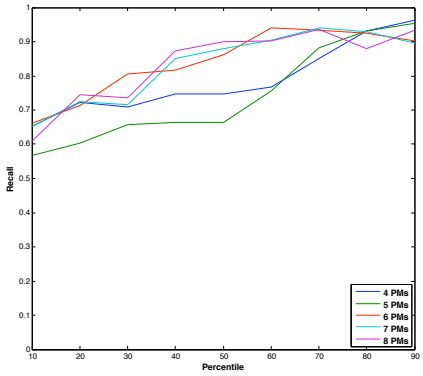


(a) Recall for Delay Estimation

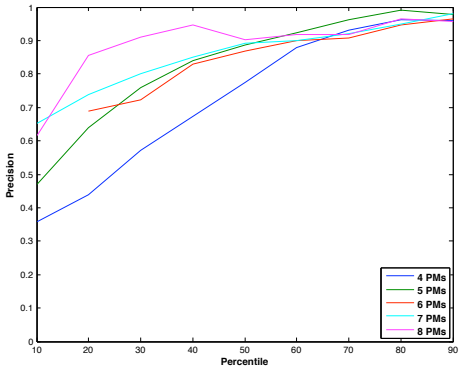


(b) Precision for Delay Estimation

Figure 7.9: Recall and Precision for Delay Estimation



(a) Recall Bandwidth Estimation



(b) Precision Bandwidth Estimation

Figure 7.10: Recall and Precision for Bandwidth Estimation

schemes as e.g. GCP equivalent to on demand active measurements. The main advantage of the described estimation techniques is the fact that they allow to narrow down the set of candidate solutions proactively. Thus they can be used to reduce the measurement overhead during a search process by excluding branches with a very low probability to result in a solution. To be able to use estimation schemes for such filter purposes we need to ensure that it has high recall values for the target scenario. For a not contractive embedding [HS00b] as GCP we can archive this by determining constants

$C_d, C_b > 0$ that we have for almost all nodes x, y in the network:

$$\begin{aligned} d_{x,y} &\leq \hat{d}_{x,y} + C_d \quad \text{in case of delay estimation} \\ d_{x,y} &\geq \hat{d}_{x,y} - C_b \quad \text{in case of bandwidth estimation} \end{aligned}$$

where $d_{x,y}$ denotes the measured and $\hat{d}_{x,y}$ the estimated delay or bandwidth values. To derive these constants for GCP and the GT-ITM Topology used in our experiments, we introduce the *Non- absolute Error* of an estimation scheme.

Definition 14. *Let x and y be two nodes in a network. If $d_{x,y}$ denotes the measured distance and $\hat{d}_{x,y}$ the estimated distance between x and y, the Non-absolute Error (NaE) of the estimation is given by:*

$$NaE(x, y) = d_{x,y} - \hat{d}_{x,y}$$

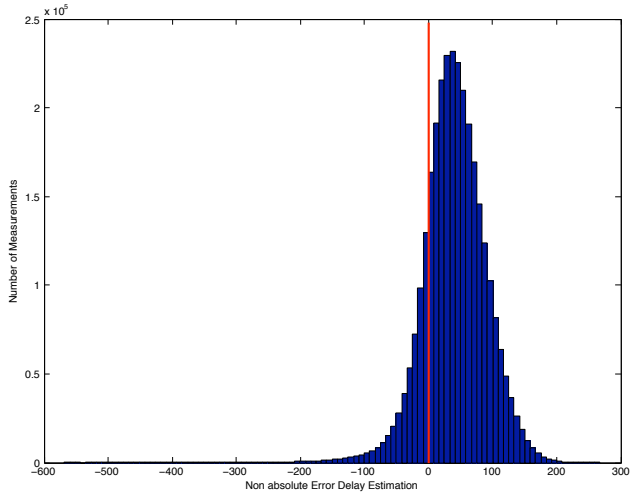
The *Non absolute Error* is a signed error measure, which allows to measure a network dependent error and provides information about the tendency of an estimation scheme to over- or under-estimate distances. In Figure 7.11(a) and 7.11(b) we show a histogram of the *Non-absolute Error* values for GCP and the used GT-ITM topology for delay and bandwidth estimation respectively. In both figures one can identify an obvious trend of GCP to underestimate distances. In addition table 7.1 shows positive average values

Estimation	**Delay**	**Bottleneck Bandwidth**
Av. Non absolute Error	40.4461	6.5537

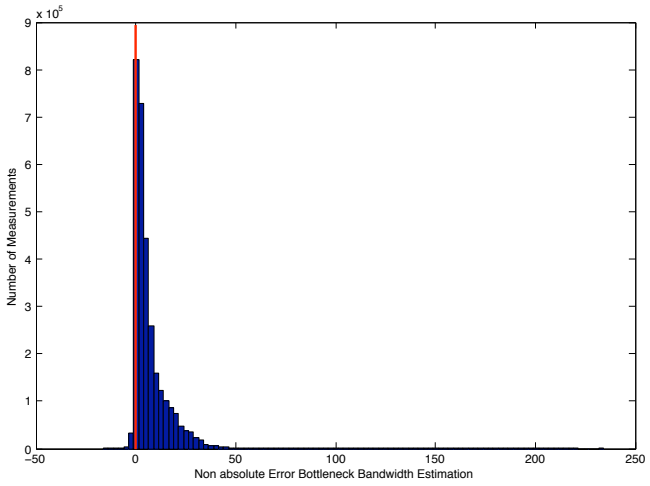
Table 7.1: Non-absolute Error GCP and GT-ITM Topology

for delay and bandwidth estimation. Based on this observation, the above mentioned constants C_d and C_b can be defined e.g. as:

$$\begin{aligned} C_d &= 40.4461 \quad units \\ C_b &= 6.5537 \quad units \end{aligned}$$



(a) Non absolute Error Delay Estimation



(b) Non absolute Error Bottleneck Bandwidth Estimation

Figure 7.11: *None-absolute Error GCP and GT-ITM Topology*

Chapter 8

Implementation

The implementation of a CSP prototype we describe in this chapter is centred around the concept of the *CSP node*. The core task required for the implementation of the CSP node has been interfacing with and extension of a P2P middleware called OMNIX [KKKD03] which has been used for the realisation of CSP DHT functions. As instances of the CSP node, a Media Client, a Media Server and a transcoding PM based on the open source VideoLAN Client [vlc08] as well as a Service Bootstrap Node have been developed. To be able to illustrate the DHT operations performed during the search for a Processing Chain a visualisation component has been included. The actual implementation was evaluated in the context of a media transcoding scenario using a Linux-based testbed. Since the required implementation work has been performed in the context of the Situated and Autonomous Service Control (SASCO) project founded by France Telecom R&D Orange Labs we used the project acronym SASCO instead of CSP inside the classdiagram in Figure 8.3.

8.1 The CSP Node

Following the P2P paradigm, CSP nodes are symmetric in functionality and can offer any combination of Media Client (MC), Media Server (MS), Media Processor (PM) or Service Bootstrap Node (SBN) functionality. An overview of the CSP Node is shown in figure 8.1. As we can see, a CSP node is composed out of three main parts related to:

Identity: This part implements the different logics for MC, MS, Processor or SBN behaviour. A single node may run any combination of these core identities.

Processing: The processing part is defined by a set of PMs hosted by the node.

CSP DHT: The CSP DHT+++ layer is the domain for the distributed control plane concepts of CSP. It corresponds to a management (Overlay) layer, responsible to handle registration of PMs and for support of

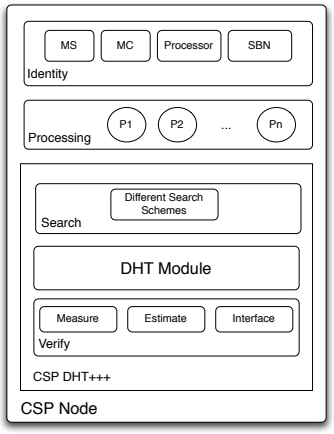


Figure 8.1: CSP Node

different search schemes and range queries.

For the implementation of the described node concept a modular approach was selected. For the CSP DHT Layer a CAN DHT was implemented utilising the OMNIX P2P Middleware [KKKD03]. Implementation details as well as a description of OMINX will be provided in section 8.3.3 below. The multimedia-specific components of MCs, MSs and PMs are realised based on VideoLAN Client Software [vlc08]. To include VideoLAN into the CSP node framework, a VideoLAN Controller with XML based control interface has been developed. For a detailed description of the processing component we refer to section 8.2. The resulting software modules and interfaces are collected in figure 8.2. The five main interfaces show in the figure are:

1. *OMNIX interface:* The OMNIX interface is the main interface used to trigger all P2P network-related functions of the software. It is defined by OMINX, and was extended with CSP specific calls to support CSP DHT primitives as different recursive search principles and range queries.

2. *VLC controller interface:* An XML-based interface provided by the developed VLC controller software.

3. *VLC interface* The standard VLC client command line interface used to trigger transcoding and streaming operations.

4. *Identity interface:* The Identity interface is used for communication among different CSP Nodes. It utilises an XML-based protocol for MS to MC, MC to PM or MC to SBN communication.

5. *Converter:* This internal interface converter is used to convert CSP-specific DHT operations to OMNIX interface calls.

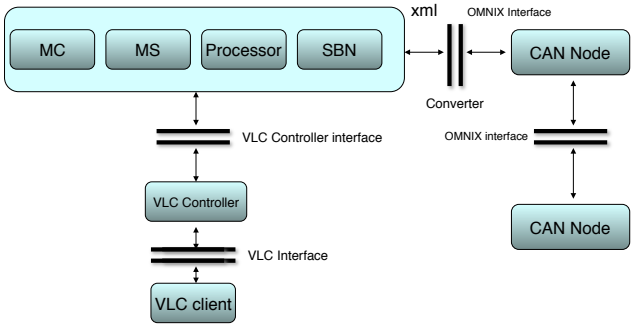


Figure 8.2: CSP Node Interfaces [Wan08]

Except the VideoLAN Client Controller software which is written in C, the major part of the described node framework has been implemented in Java. A simplified class diagram of the implementation containing only the most important classes and the interfaces among them is shown in figure 8.3. The classes serve the following functions:

`at.ac.tuwien.infosys.cradle.net:Module`
: OMNIX class, for a standard P2P topology module, defines interfaces for the implementation of new P2P topologies.

`can:CanPeer`
: Encapsulates data and functions of the CSP DHT peer using a CAN

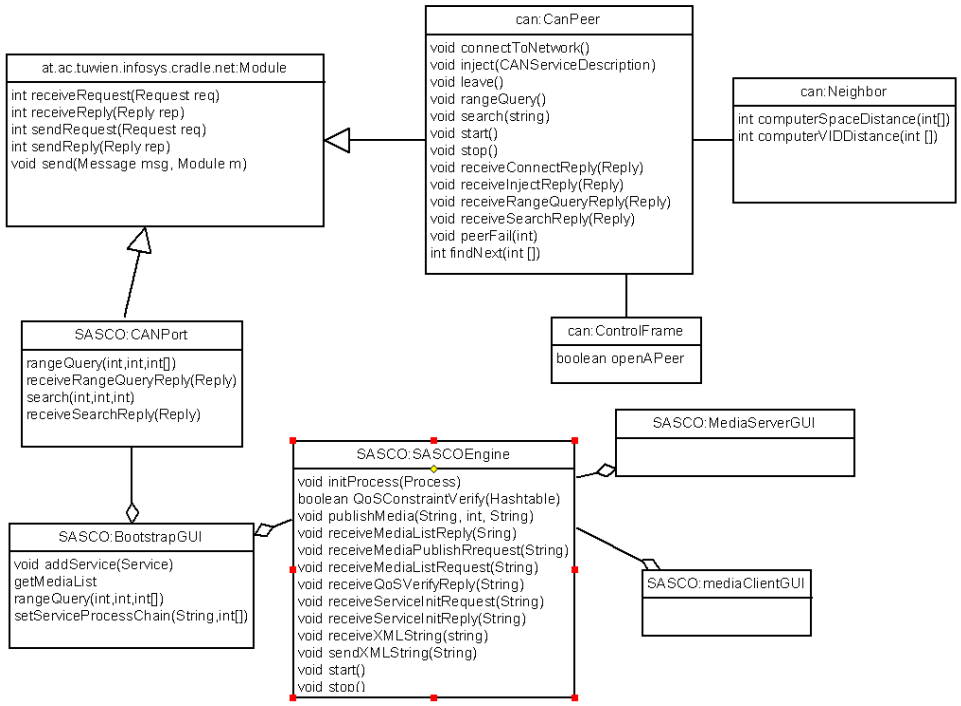


Figure 8.3: Simplified Class Diagram for CSP software [Wan08]

DHT. The class inherits `at.ac.tuwien.infosys.cradle.net:Module` to use the underlying communication interfaces and provides P2P interfaces upwards to higher layers.

`can:Neighbor`

Used to encapsulate the data and functions of a neighbour of a CAN peer.

`can:ControlFrame`

For visualization of the CSP DHT. Several CAN peers may be instantiated to build a CAN system.

`SASCO:SASCOEngine`

Engine to drive CSP-related operations. The entire operative schemes for CSP are maintained in this class.

`SASCO:CANPort`
: Used as internal protocol converter between identity modules and CAN layer.

`SASCO:BootstrapGUI`
: Management GUI used to manage and configure services provided by the system.

`SASCO:MediaServerGUI`
: GUI for the Media Server.

`SASCO:MediaClientGUI`
: GUI for the Media Client.

`SASCO:BootstrapGUI`
: GUI for the Bootstrap Node.

8.2 Processing Component Implementation

To realise multimedia-specific tasks of the processing component, we adopted the VideoLAN Client (VLC) [vlc08] as third party software. The VLC media player is a highly portable multimedia player supporting various audio and video formats as e.g. MPEG-1, MPEG-2, MPEG-4, DivX, mp3, ogg as well as DVD and VCD playback and several streaming protocols. The player has server as well as client functionality and is released under the General Public License (GNU). To integrate VLC into our prototype we developed a wrapper application called VideoLAN Client controller (VCC) which offers a XML based interface for VideoLAN control. Figure 8.4 shows how VCC integrates into the CSP Node architecture. The current version of VCC supports functions as e.g `startStreaming`, `stopStreaming`, `startTranscoding`, `stopTranscoding` which can be accessed from applications via an UDP socket using an XML based control protocol. The developed XML protocol is based on a request/response principle where a $flow_{id}$ parameter is used to identify the stream to operate on. A response contains a status code to inform about the success/failure of the request command. Requests can

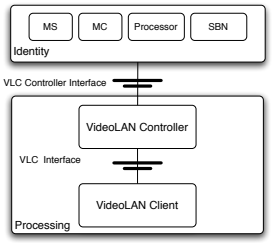


Figure 8.4: VideoLAN Cient Controller (VCC)

```
<message type="request">
<flow id="flow_id"/>
<start>
<source />
<destination>
<network proto="udp" ip=IPMC" port=PortMC"/>
<transcoding interlace="yes/no">
<codecs video="" audio="" />
<size width="" height="" />
<bitrate video="" audio="" />
<transcoding/>
</destination>
</start>
</message>
```

Figure 8.5: VCC Transcoding Request

be `start` or `stop` where a `start` request provides a media source and one or more destinations in the form of `IP:Port` pairs. Each destination can also be transcoded. If the source is network, a `start` response will return the `IP:port` where the media must be sent to by the streaming server. If figure 8.2 and 8.2 we included an example for a transcoding request respectively the corresponding response message.

For the streaming of multimedia data from MS to MC with and without

```
<message type="reply">
<flow id="flow_id"/>
<status code="" />
<start>
<destination>
<network proto="udp" ip=IPPM" port=PortPM"/>
</destination>
</start>
</message>
```

Figure 8.6: VCC Transcoding Response

processing, VCC is used as illustrated in figure 8.8 and 8.7 respectively.

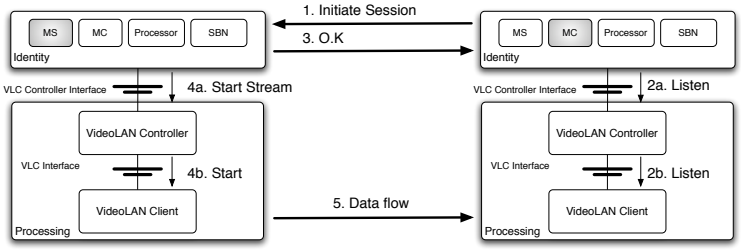


Figure 8.7: VCC based Streaming (Simplified)

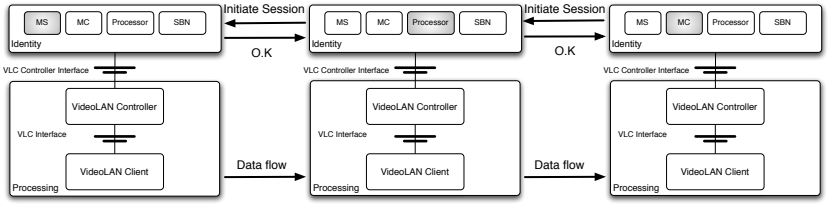


Figure 8.8: VCC based Processing (Simplified)

8.3 CSP DHT Implementation

The implemented CSP DHT was realised as a CAN in a 3-dimensional Cartesian coordinate space. At any point in time, the entire coordinate space is dynamically partitioned among all the nodes in the system so that every node owns its individual, distinct zone within the overall space. Each node maintains the IP addresses and coordinates of its CAN neighbours. The set of immediate neighbour coordinates serves as a coordinate-based routing table that enables routing between arbitrary points using a CAN specific greedy routing principle. For an algorithmic description how to realise the CAN specific *join*, *publish* and *search* we refer to [RFH+01] or [Rat02]. In this section we focus on our implementation of the *repair* function [Wan08] for the CAN topology which is robust to the simultaneous failure of multiple nodes and lightweight compared to the one described in [Rat02]. Before we

can start we need to introduce the required concepts using a two dimensional CAN as a reference.

8.3.1 Partition Trees and Virtual Identifiers

Each CAN can be identified with a binary tree also called partition tree. As shown in Figure 8.9, an existing zone corresponds to a leaf of the tree and the internal vertices represent zones that no longer exist, but have been split in the past as the result of a node join operation. The children of a tree vertex are the two zones into which it was split. Following the description in [RFH+01], the edges in the partition tree are labeled as follows: an edge between a parent and a child zone is labeled with 0 if the child zone occupies the lower half of the dimension along which the parent zone was split. Otherwise (i.e., if the child zone occupies the upper half of the dimension along which the split occurred) the edge is labeled with 1. Based on this labeling scheme, the position of a zone in the tree is uniquely defined by the path from the root of the partition tree to the leaf corresponding to that zone. The binary string obtained by concatenating the resulting partition tree labels is also denoted as the virtual identifier (VID) of the node responsible for this zone. It is important to note that none of the nodes in a CAN needs to maintain the partition tree of the entire CAN. In fact only partial knowledge of the partition tree is sufficient to realise repair functions for the CAN topology in case a node is leaving or fails.

8.3.2 Leave Operation

In a CAN DHT, any node may leave the system gracefully at any time. In addition it is also possible that a node does not gracefully leave the CAN because of software or network failure. For a robust implementation of a CAN system both situations have to be handled. In the following we describe the realisation of a CAN leave operation for the *graceful* and *non-graceful* case. We start with the description of the leave operation from [Rat02] and afterwards our implementation of a function for the non-gracefully leaving of

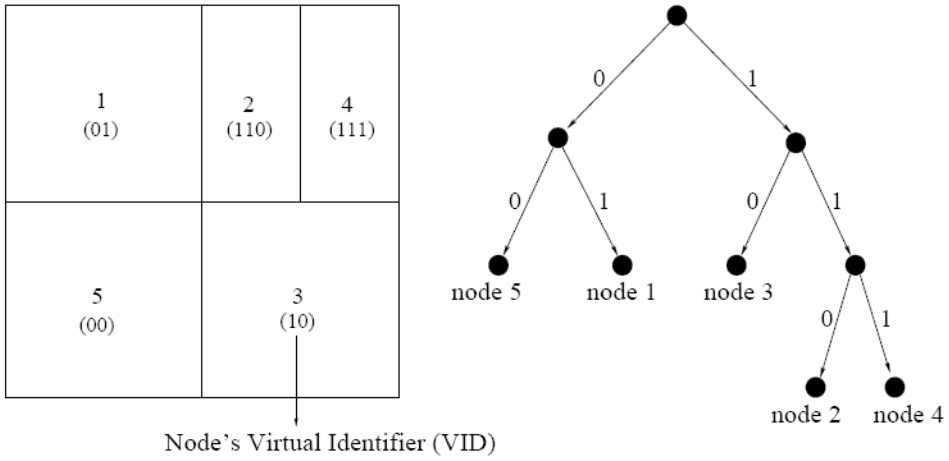


Figure 8.9: CAN and corresponding Partition Tree [RFH+01]

nodes which is more robust to the simultaneous failure of multiple nodes as the one presented in [Rat02].

To keep a CAN in a valid state when a node leaves, it is required to ensure that the zone the node had occupied is taken over in a consistent way. In case a node is gracefully leaving it explicitly hands over the zone and its associated state (i.e., its VID, the list of neighbour VIDs and IP addresses) as well as the stored (key, value) database to a specific node called the *takeover node*. The takeover node has to be selected in a way that its own zone can be merged with the departing nodes zone to produce a new valid single CAN zone[1]. To describe the way how a takeover node is selected we need the concept of *sibling* of a CAN node. Conceptually two CAN nodes are siblings in case their zones originating from the same *previous* zone. Thus the sibling of a leaving node is the default candidate for the takeover node. To illustrate the sibling relation, in Figure 8.10, node 4 and node 7 are siblings because they are direct children of same vertex in the partition tree, while node 2 and node 4, or node 6 and node 3 are not. Using VIDs, the sibling relationship can be described as follows: Two nodes at depth D of a CAN Partition Tree

[1]If this is not possible for some reason, a takeover node can temporarily handle both zones.

are siblings if and only if they share the same D-1 digit VID prefix and the last digits of their VID are complementary.

Besides the concept of sibling, we also need to define the VID distance of two nodes. To do this we assume that the maximum length of a node VID for an arbitrary CAN is L. If a node corresponds to a zone at depth $D < L$ of a CANs partition tree, we also say the *valid VID length* of this node is D. The VID distance of two nodes is now defined as the numerical difference of the Max_D digit binary prefix of the two node VIDs, where Max_D denotes the bigger one of two node's valid lengths. To illustrate this, in Figure 8.10 the VID of node 2 is 01‖000 and its valid length is 2. The VID of node 7 is 111‖00 and its valid VID length is 3. Thus Max_D is 3 and the VID distance of the two nodes the difference of the binary numbers 111 and 010 which equals 5.

8.3.2.1 Gracefully Leaving Nodes

Using the terminology introduced above, we describe the implemented leave operation using an example. Figure 8.10, Figure 8.11 and Figure 8.12 depict the departure of node 7 and node 2. When node 7 is leaving it selects its sibling node 4 as the takeover node. It sends a leave request with its zone state and associated database to node 4. As node 4 receives the leave request it merges the departure nodes zone and its own zone and takes over the (key, value) pairs stored at node 7. Finally node 4 removes node 7 from its neighbour list and sends an update request to all remaining neighbours of node 7. The CAN resulting after the departure of node 7 is shown in Figure 8.11.

We now consider the case that node 2 is leaving which has no sibling. In this case the takeover node is selected as the immediate neighbour with the shortest VID distance. In our example the VID distances of all neighbours of node 2 are listed in Table 8.1. Thus the takeover node of node 2 is node 4. Node 4 can not directly merge the two zones. Instead, node 2 is kept as a virtual node in the CAN, i.e. a node which does no longer exist in the CAN system but all its responsibilities are handled by node 4 as the virtual node

handler. Node 4 removes node 2 from its neighbour list and sends a notify message to all remaining neighbours of node 2. The resulting CAN after the departure of node 2 is shown in Figure 8.12. As described, node 2 exists as a virtual node and is handled by node 4. During the node join procedure, a JOIN request is always forwarded to neighbours which are virtual neighbour handlers if possible.

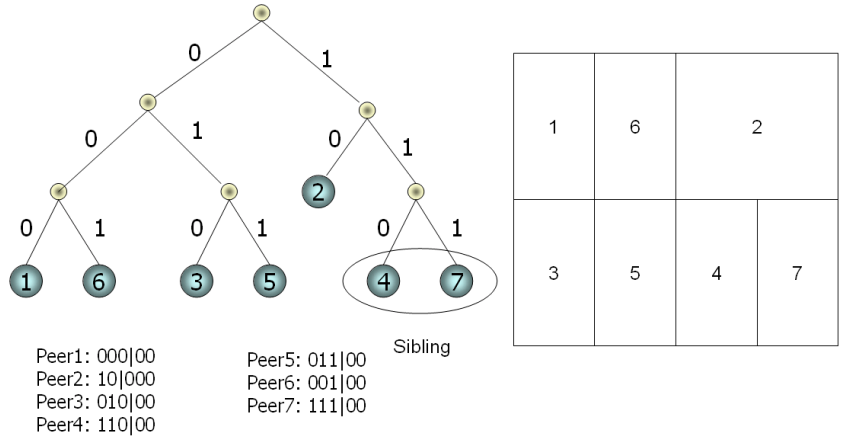


Figure 8.10: Situation Before Node 7 Leaves the CAN

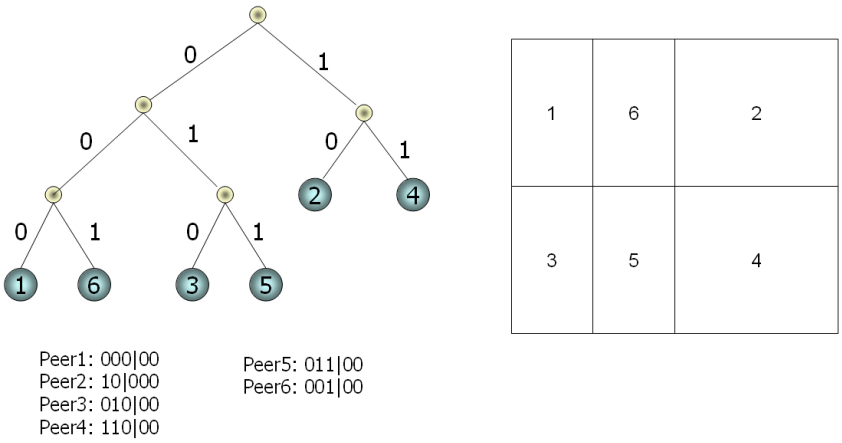


Figure 8.11: Node 7 Has Left

Nodes	VID Distance	VID Length
2, 1	4	3
2, 4	2	3
2, 7	3	3
2, 6	3	3

Table 8.1: CAN VID Distance Table

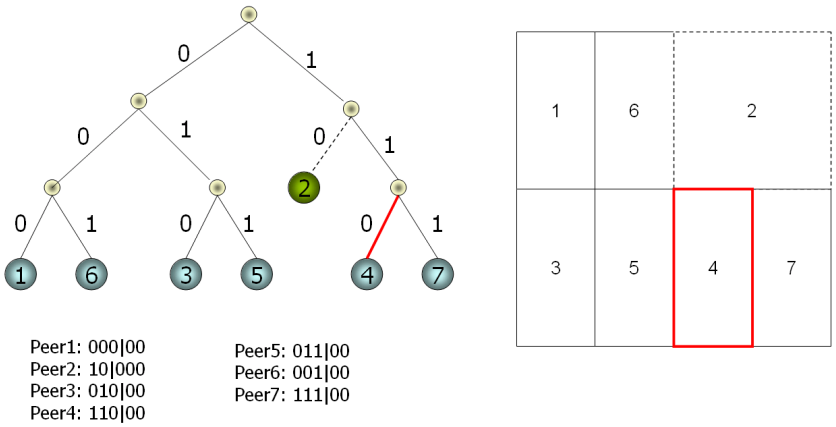


Figure 8.12: Node 2 Has Left

8.3.2.2 Not Gracefully Leaving Nodes

To detect node failures in a CAN a heartbeat principle is used. As part of its normal operation, a CAN node sends heartbeat messages periodically to each of its neighbours. The prolonged absence of a neighbour's heartbeat signals its failure. When a node detects a failed neighbour, it removes the node from its neighbour list and attempts to contact the dead neighbours takeover. Since it can not be assumed that the VID or IP address of the takeover node are known by the node detecting the failure, the core task of the recovery from not gracefully leaving nodes is to find the right takeover node. As described above, the takeover node is the closest immediate neighbor of the dead node in term of VID distance.

We start with the description of the solution presented in [Rat02] which is initially not robust in case of the failure of multiple nodes at the same time. The main reason for this is the fact that it relies on CAN routing. In contrast, our solution proposed utilises a time competition mechanism to

solve this problem.

Original approach to handle not gracefully leaving nodes To identify the right takeover node, a recovery message is sent to the neighbour closest to the dead node in terms of VID-distance. The recovery message is now routed incrementally closer to the dead nodes VID until it arrives at the takeover node which has the smallest possible VID distance to the dead node. Based on this fact, the takeover node can infer that it has to occupy the dead nodes zone and that the source of the initial recovery message is a neighbour of this newly acquired zone. Since each of the dead nodes neighbours will independently initiate such a recovery message, the takeover node discovers all its new neighbours and vice versa. Unfortunately this algorithm fails if multiple nodes fail simultaneously. In case of the failure of multiple nodes it is possible that a recovery message might dead-end at a node other than the takeover node. To address these cases in [Rat02] an idea from Chord is adopted. In addition to its regular neighbour list, every node maintains also links to its immediate successor and predecessor in the VID space and utilises the Chords stabilisation algorithm to actively maintain this ordered link list of nodes. Given that this chain of VIDs is maintained, a recovery message is guaranteed to arrive at the takeover node. However, this approach adds additional complexity to the resulting implementation because of the required Chord algorithms and the fact that each node needs to know its immediate successor and predecessor in the numerical ordering of VIDs. To keep our implementation as lightweight as possible, we developed an alternative algorithm which is described in the following.

Our approach to handle not gracefully leaving nodes As a prerequisite of the approach it is required that each node maintain the IP addresses and zone information of its immediate neighbours as well as the neighbours of its neighbours. In addition for every neighbour a lifetime value is assigned which is initialised equal to its VID-distance. In case of the prolonged absence of a heartbeat we have to distinguish two cases:

Dead node is sibling of detecting node: In this case, the detecting node takes over the dead node's zone directly and sends a notify message to all its new neighbours which are known as mentioned in the prerequisites.

Dead node is not a sibling of detecting node: In this case the lifetime value described above comes into play. If the lifetime of the dead neighbor is 0, the detecting node becomes the takeover node and sends a notify message to all its new neighbours. If the lifetime is greater than "0", the lifetime of the dead neighbour is reduced by 1 and its heartbeat time is refreshed.

Since the lifetime of a node is equal to the VID-distance, the node closest to the dead node has the shortest lifetime value for the dead node and is thus the first one starting the takeover procedure. In addition, once a node receives a notify that one of its neighbours is dead and its zone is handled by another node, it removes the dead node from its neighbour list and adds the takeover as a new neighbour. Because of this lifetime based *time competition approach* and the fact, that our solution does not rely on routing it is also robust in case of the failure of multiple nodes.

8.3.3 The OMNIX Peer-to-Peer Middleware

For the implementation of the required CAN DHT principles different P2P middleware software has been evaluated. In general, a P2P middleware provides an abstraction layer located between P2P applications and the underlying P2P overlay network. Examples for existing P2P middleware software are eComP [KMRW02], VPP [VMF02], JXTA [jxt08] and OMNIX [KKKD03]. From these candidates for a CSP implementation, only JXTA and OMNIX provided a straight forward way to integrate new P2P topologies into the middleware. After a comparison of different frameworks and libraries our decision was to use OMNIX. OMNIX is an open, topology-independent P2P middleware framework designed by Roman Kurmanowytsch and described in his PhD thesis at Vienna University of Technology. OMNIX provides an

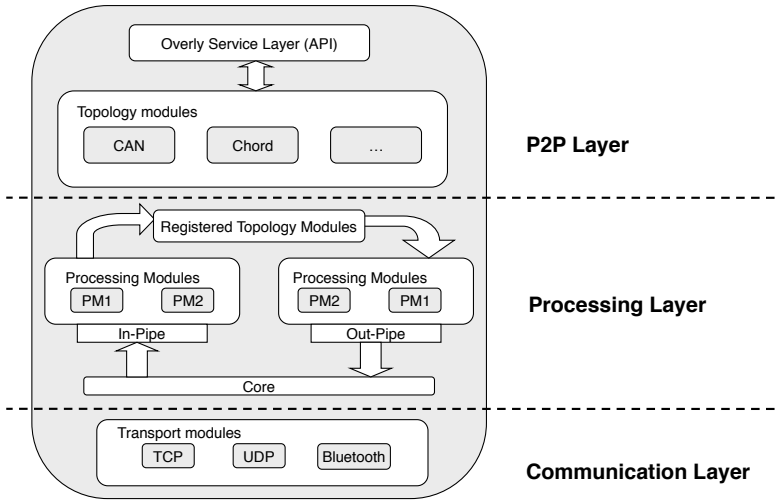


Figure 8.13: Layered and Modular Architecture of OMNIX

abstraction layer between P2P applications and the underlying network. In addition, it offers the possibility to develop modules for new P2P network schemes. As a consequence, it provides a consistent application programming interface to the application layer while the underlying P2P topology can be changed without making code modifications in the application necessary. An overview of OMNIX is shown in figure 8.13. When an application wants to send a message to another application residing on a peer across the network, it calls the `invoke()` method specified in the OMNIX API. This method requires information such as the IP address of the target peer and the content of the message. The topology module then creates a self-describing message object out of this information and puts it into the outgoing message queue (i.e. the out-pipeline) of the processing layer, where it gets processed by the installed processing modules. One processing module adds security-related features i.e. a digital signature. Once the message has passed the outgoing processing pipeline, it is given, via the OMNIX core, to the appropriate transport module which conveys the message over the network

The core is responsible for choosing the correct transport module accord-

ing to the systems configuration. On the target peers side, the message is received by a transport module, parsed and put into the incoming message pipeline. In the incoming pipeline, the message is processed in the same way as in the outgoing pipeline on the senders side. The digital signature is verified by one of the installed processing modules. After leaving the pipeline, the topology register gets hold of the message. According to the meta information stored in the message, the message is dispatched to one of the topology modules installed. Finally, the message is forwarded to the application. It is also possible for an application to directly access the transport layer. In this way, applications enjoy an even higher flexibility, which might be necessary if the OMINIX API does not provide the required means for a specific functionality. To implement CAN under OMNIX, we identified two main tasks to be done. First we had to implemented a new UDP transport module and secondly we implemented a topology module. The new transport module is UDP-based and provides a send and receive function to the higher layers of OMNIX. The CAN topology module implements all the topology information and routing principles of a CAN as described above. It uses the OMNIX internal in-pipeline and out-pipeline to send and receive messages. For higher level applications, the topology module provides the five essential DHT functions:

1. Join
2. Publish
3. Search
4. Delete
5. Leave

CAN functionalities, such as dynamic partitioning and merging of the coordination space, routing of messages on the CAN P2P overlay layer and the described rapair functions are transparent for the application layer.

8.3.4 OMNIX Message Protocol

OMNIX adopts an HTTP-like protocol to exchange messages. There are two types of messages: Request and reply. Both message types have a single header line, multiple header fields, an empty single line signalling end of header fields, and a content (c.f. figure 8.14). The code in figure 8.15 shows an example OMNIX message. As we can see, at the first line of the mes-

```
Message = requestline / replyline
*Messageheader
CRLF
[ Message Body ]
```

Figure 8.14: Basic Form of OMNIX Message [Kur04]

```
CAN_CONNECT 192.168.0.21:10002 CAN/1.0
MsgID: 56992928907622082864
Content-Length: 0
PID: 38139470762470863951
Destination Coordinate: 426.2277.3605
Transaction: 56992928907622082864
```

Figure 8.15: Example OMNIX Message

sage it contains 3 properties: Request Type, Source Address (IP:Port), and Topology Protocol (Name/version). Every message has a unique ID called MsgID, and also a content length field. For implementing a CAN, we introduced additional headers, as peer ID (VID), CAN coordinates of destination and a transaction ID. The OMINIX protocol mainly covers communication between two peers. The sending peer sends a request to the remote peer, which in turn sends back a response, providing information about the processing of the request. There are no semantics associated with a request. It may be a search request, an instant message, the current time, etc. The response may be a status report, a requested artifact, etc. For every request received, a peer must send back a response. On the left hand side of Figure 8.16 from [Kur04] we have illustrated the message flows in case peer A wants to route a message to peer C via peer B in the CAN. On the right hand side, the message flows in case of a DHT-like redirection based communication is shown.

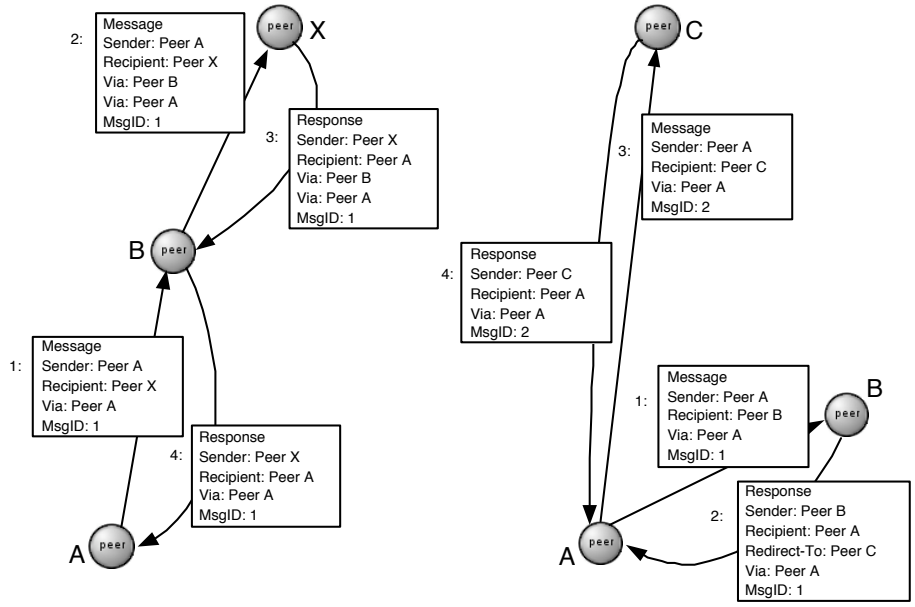


Figure 8.16: OMNIX Message Flows [Kur04]

8.4 Evaluation of Software

To evaluate the developed software, we realised an on-demand media transcoding scenario. The scenario is realised by a distributed system running multiple instances of the developed CSP node software. As in a classical P2P network new CSP nodes can join or leave the system at any time. In addition, the software includes the above described repair operations for the DHT in case a node leaves the system not gracefully. To be able to demonstrate this self-repair capability, we have included an additional node hosting a single CSP node (c.f. Singe Peer node with IP 192.168.0.4 in Figure 8.17). To simulate the failure of this node it is possible to either unplug its network cable or just to stop the CSP node software. With regard to the recovery time it is important to note that each node checks the integrity of its neighbourhood using a heartbeat principle. In case a neighbour does not respond to a heartbeat message, the node assumes this neighbour is no longer alive and starts a repair algorithm as described above. The heartbeat frequency in the current implementation is set to 10 seconds. Therefore, a delay of 10

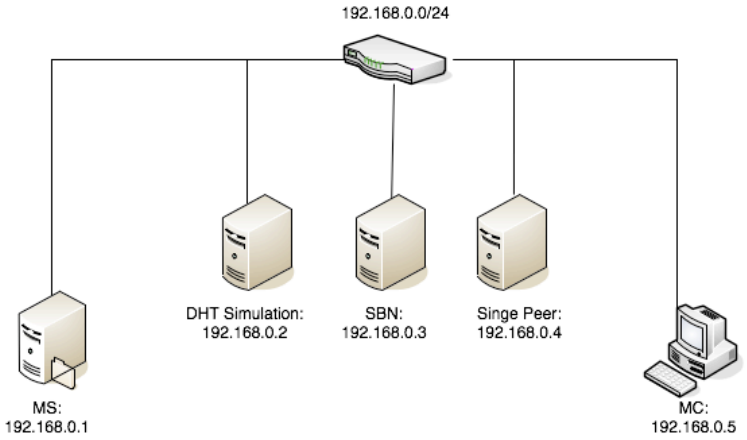


Figure 8.17: Testbed Used as Reference for This Section

seconds before the repair can be started is possible. If required, it is possible to increase the heartbeat frequency. In order to show how the CSP DHT

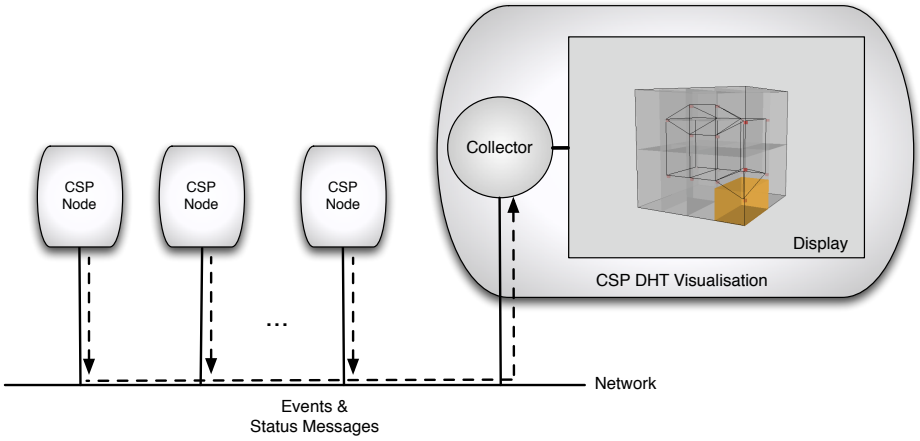


Figure 8.18: Real-time Visualisation of CSP DHT

grows while peers join or how the structure changes while peers leave a real-time visualisation software has been developed. The software is composed out of a collector and a display component as shown in figure 8.18. In case visualisation is enabled, each CSP Node periodically sends event and status information to the collector component. Based on the information received, the collector generates the corresponding display requests and sends it to the display component. Events that can be visualised include:

- Peer arrival
- Peer failure
- Repair operation
- Publish operation
- Search operation

To illustrate this, figure 8.19 shows a set of screen shots from the visualisation of a DHT repair operation in a CAN with 20 nodes.

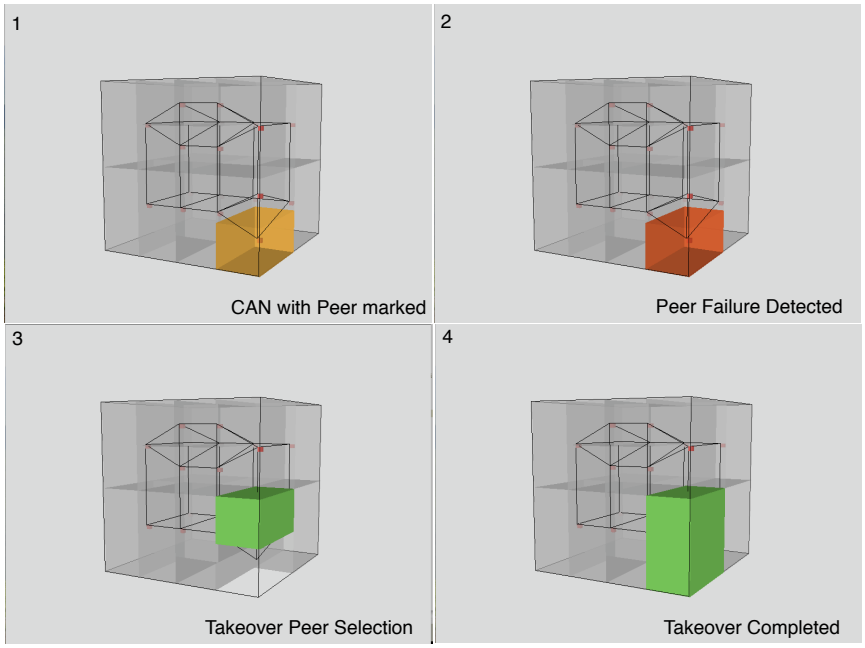


Figure 8.19: Visualisation DHT Repair Operation

To verify the correct operation of the developed CSP Software, we performed an evaluation in a scenario where 50 CSP nodes are connected in a system with 100 PMs distributed among them. We performed a set of experiments to evaluate the behaviour of the implementation with regard to the *scope of a join operation*, *maintenance costs* and *search costs*. The different metrics from each category are collected below.

Scope of a join operation:

1. ARH_{JOIN} Average number of nodes involved in a join operation

Maintenance costs and recovery time of the DHT ($k = 1, 2$):

1. $ANM_{RECOVER}$ Average number of messages needed for a single recovery operation
2. $AMT_{RECOVERk}$ Average time (in seconds) needed to recover from the failure of k nodes
3. $MAXT_{RECOVERk}$ Max time (in seconds) needed to recover from the failure of k nodes

Search Costs:

1. ARH_{SEARCH} Average number of nodes involved in a search operation
2. ANM_{QUERYk} Average number of messages required for a chain query when the length of the searched Processing Chain is set to $k = 1, 2, 3, 4, 5$

The experiments have been repeated 10 times and the results obtained are collected in Table 8.2, Table 8.3 and figure 8.20 below. As a result of

ARH_{JOIN}	ARH_{SEARCH}
3.7	2.6

Table 8.2: Average Scope of Join and Search Operation

$AMT_{RECOVERY1}$	$AMT_{RECOVERY2}$	$MAXT_{RECOVERY1}$	$MAXT_{RECOVERY2}$
15.2	22	22.4	31

Table 8.3: Time Complexity of Repair Operation in Seconds (Heartbeat Interval 10 seconds)

the performed evaluation we can state that the implementation behaves as expected following the simulation results. The costs of a join and single search

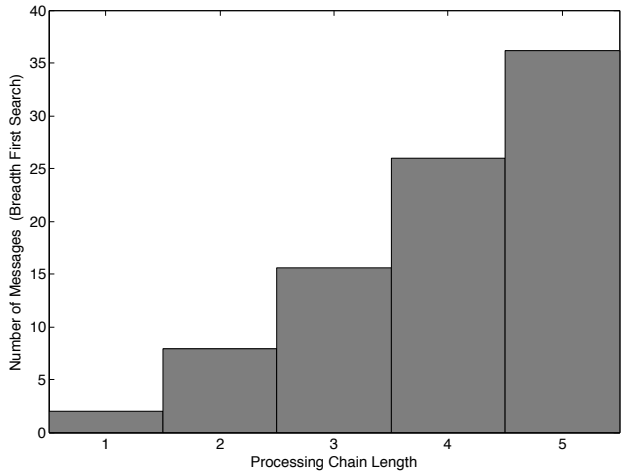


Figure 8.20: Message Complexity of CSP Search Requests (Breadth First Search)

operation are comparable low because of the utilised CAN DHT principle. In case of the search for processing chains of increasing length, the message complexity grows as expected since we used a breadth first based search algorithm for the implementation. With regard to the recovery time we can state that the system is able to recover from failures in a range of three heartbeat intervals. In the performed tests the time required to establish a processing chain with a single transcoding module was below one second. Since the tests have been performed in a somehow ideal environment, we refer to the evaluation in chapter 7 for an estimation of the response time in case of different search schemes and longer processing chains. As part of the performed evaluation we used the minimal number of search steps performed until a solution has been found as a rough indicator for the time required to complete a CSP search request.

Chapter 9

Conclusions

With this thesis we have addressed two challenges for a flexible media provisioning on top of today's Internet:

1. An end-to-end networking paradigm [SRC84] based on a service and data unaware transport.

2. The lack of *high level* communication abstractions [SAZ+02].

As the result we described a self-managing and self-repairing system for a cooperative service provisioning of media transport and processing services. CSP can be applied in classical Provider as well as P2P scenarios. The proposed platform is based on a DHT principle known from the area of P2P Networks. We have realised an *Overlay-based Service Composition*, allowing to combine and reuse existing service components for building new (composed) services. Candidate entities for hosting service components are end-systems as well as network nodes being part of the infrastructure of a provider.The main part of research and development work has been performed inside the project Situated and Autonomous Service Control (SASCO) founded by France Telecom R&D Orange Labs. Parts of the described CSP approach are filed as the pending patent FR 0851098 "Procédé et dispositif de contrôle de la qualité de service dans un réseau".

9.1 Contributions of this PhD

With CSP [KBE+07] we introduced a novel reactive and decentralised system for overlay based service composition. The creation of Service Overlays is addressed by combining a DHT-based search principle with a hop-by-hop QoS constraint verification and propagation technique. The DHT-based approach has been selected because the resulting system can be realised in a fully distributed way and inherits the self-organisation and self-repair properties of DHTs. As a prerequisite we reduced Service Overlay related routing problems to search problems. We provided a graph theoretic view to the service and network specific aspects of service overlay construction and a complexity analysis of the proposed CSP approach. For the DHT-based realisation of CSP we have extended a Content Addressable Network DHT with:

1. An indexing scheme that allows to encode a system's *Service Graph* into the CAN address space.

2. A $O(\log N)$ range query scheme (where N denotes the number of CAN members).

3. Recursive search principles for Processing Chain search.

4. Primitives for verification of QoS constraints between two arbitrary CSP nodes.

As the result of a simulation based study of different CSP search principles we can state that for short Processing Chains (e.g. up to three processing steps) a Breadth First-based search is the preferred option. With regard to a CSP specific scope measure, less than 10 percent of nodes have been involved in the corresponding search process. In addition the response time is close to optimal since no backtracking is required and all possible solutions are returned. When analysing chains from four to eight processing steps, a Depth First-based search with two or three concurrent branches has shown the most promising results.

A further task in the context of overlay-based Service Composition is to ensure that given QoS constraints of a service are not violated by the composition process. In contrast to other work we had a strong focus on the applicability of QoS estimation techniques. The reason to study QoS estimations was given by the fact that such techniques allow to decrease the time and measurement complexity of the actual service composition process. We use delay (rtt) and bottleneck bandwidth as the two reference metrics during this thesis. Starting with rtt estimation, we introduced Geometric Cluster Placement (GCP) [KZ04] which was developed in the context of this thesis. Based on a performed comparison with state-of-the-art delay estimation schemes we can state that GCP shows superior accuracy for estimating small delays. In a second step we combined GCP with clustering and a triangulation heuristic first used by Hotz [Hot04]. As a second set of experiments showed, the resulting *Netforecast* [EKP07] [Elm07] scheme further improved

estimation accuracy. The actual improvement is (depending on the used metric and network data) up to 18 percent.

The question of how to predict bottleneck bandwidth in large scale scenarios had not been addressed in a broader scope previously. In order to fill this gap we proposed a novel method for predicting bottleneck bandwidth using landmark based distance estimation combined with a transformation step. As an unexpected result, the used transformation approach improved the accuracy of landmark based bandwidth prediction by more than 85 percent when combined with GCP.

The development of a CSP prototype based on the Peer-to-Peer middleware software OMNIX [KKKD03] has been described. For the implemented Content-Addressable Network (CAN) DHT a new algorithm to repair the CAN structure in case of multiple not gracefully leaving nodes has been integrated. The actual implementation was evaluated in the context of a media transcoding scenario using a Linux-based testbed.

9.2 Further Work Items

A further work item would be how to address charging and AAA aspects. During the thesis, we assumed cooperation between peers involved in service provisioning. While such an assumption is valid in a provider-controlled network, the situation is different in classical P2P networks. For such scenarios further work is required on the development of a security and/or reputation framework able to identify misbehaving or bogus peers. While we proposed and implemented an alternative CAN repair algorithm, we did not describe any data replication schemes. While assuming that each CAN node is replicating its DHT related information at least with its neighbours, a throughout analysis of data replication schemes for CSP is subject of future work. We have provided a framework to include arbitrary search principles in CSP considering delay, bottleneck bandwidth and costs as QoS metrics. The incorporation of other QoS metrics such as jitter and loss would require further investigation. In addition, the problem of scalable loss or jitter estimation for large groups of nodes has not yet been addressed. Since routing in today's

internet is often controlled by business and not by communication aspects, a further interesting question would be what are the right mathematical models in order to estimate QoS.

Bibliography

[ABKM01] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris. Resilient Overlay Networks. In *Proceedings of Symposium on Operating Systems Principles*, 2001.

[aka08] Akamai Technologies (http://www.akamai.com), 2008.

[amb08] Ambient Networks Project (http://www.ambient-networks.org/), 2008.

[And02] F. Andreasen. Session Description Protocol (SDP) Simple Capability Declaration. RFC 3407 (Proposed Standard), October 2002.

[BBC+98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Service. RFC 2475 (Informational), December 1998. Updated by RFC 3260.

[Bou85] J. Bourgain. On Lipschitz embedding of finite metric spaces in Hilbert space. *Israel J. Math.*, 52:46–52, 1985.

[CC01] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, London, second edition, 2001.

[CCRK04] M. Costa, M. Castro, A. Rowstron, and P. Key. PIC: Practical Internet Coordinates for Distance Estimation. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS)*, Tokyo, Japan, 2004.

[CJK+03] M. Castro, M. Jones, A. Kermarrec, A. Rowstron, M. Theimer, H. Wang, and A. Wolman. An Evaluation of Scalable Application-level Multicast Built Using Peer-to-peer overlays. In *Proceedings of IEEE INFOCOM*, 2003.

[CLR90] T.. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[CRZ00] Y. Chu, S. Rao, and H. Zhang. A case for end system multicast. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2000.

[CTW01] S. Choi, J. Turner, and T. Wolf. Configuring Sessions in Programmable Networks. In *Proceedings of IEEE INFOCOM*, 2001.

[dai08] Daidalos Project (http://www.ist-daidalos.org/), 2008.

[DCKM04] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A Decentralized Network Coordinate System. In *Proceedings of ACM SIGCOMM*, 2004.

[DG99] S. Dasgupta and A. Gupta. An elementary proof of the Johnson-Lindenstrauss Lemma. Technical Report TR-99-006, International Computer Science Institute (ICSI), Berkeley, CA, 1999.

[EKP07] A. Elmokashfi, M. Kleis, and A. Popescu. NetForecast: A Delay Prediction Scheme for Provider Controlled Networks. In *Proceedings of IEEE GLOBECOM*, 2007.

[Elm07] A. Elmokashfi. Scalable Decentral QoS Verification Based on Prediction Techniques and Active Measurements. Master's thesis, Department of Telecommunication Systems, Blekinge Institute of Technology, 2007.

[emu08] eMule (www.emule-project.net), 2008.

[FJP+99] P. Francis, S. Jamin, V. Paxson, L. Zhang, D. F. Gryniewiczand, and Y. Jin. An Architecture for A Global Internet Host Distance Estimation Service. In *Proceedings of IEEE INFOCOM*, New York, 1999.

[GJ79a] M. R. Garey and D. S. Johnson. *A Guide to the Theory of NP-Completeness*. Freeman,California, USA, 1979.

[GJ79b] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[GN02a] X. Gu and K. Nahrstedt. A scalable QoS-aware service aggregation model for peer-to-peer computing grids. In *Proceedings of the 11th EEE International Symposium on High Performance Distributed Computing (HPDC)*, 2002.

[GN02b] X. Gu and K. Nahrstedt. Dynamic QoS-Aware Multimedia Service Configuration in Ubiquitous Computing Environments. In *Proceedings of IEEE 22nd International Conference on Distributed Computing Systems(ICDCS)*, 2002.

[GN06] X. Gu and K. Nahrstedt. Distributed multimedia service composition with statistical QoS assurances . *Multimedia, IEEE Transactions on*, 8(1):141–151, Feb. 2006.

[GNCW03] X. Gu, K. Nahrstedt, R. Chang, and C. Ward. QoS-Assured Service Composition in Managed Service Overlay Networks. In *Proceedings of IEEE 23rd International Conference on Distributed Computing Systems (ICDCS)*, 2003.

[gnu08] Gnutella network (http://www.gnutella.com), 2008.

[GNY04] X. Gu, K. Nahrstedt, and B. Yu. SpiderNet: An Integrated Peer-to-Peer Service Composition Framework. In *Proceedings*

of 13th IEEE International Symposium on High Performance Distributed Computing (HPDC), 2004.

[GSG02] K. Gummadi, S. Saroiu, and S. Gribble. King: estimating latency between arbitrary internet end hosts. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, 2002.

[GWvB+01] S. Gribble, M. Welsh, R. von Behren, E. Brewer, D. Culler, N. Borisov, Steven E. Czerwinski, Ramakrishna Gummadi, Jon R. Hill, Anthony D. Joseph, Randy H. Katz, Z. M. Mao, S. Ross, and Ben Y. Zhao. The Ninja architecture for robust Internet-scale systems and services. *Computer Networks*, 35(4):473–497, 2001.

[Has92] R. Hassin. Approximation schemes for the restricted shortest path problem. *Math. Oper. Res.*, 17(1):36–42, 1992.

[Hot04] S. Hotz. *Routing Information Organization to Support Scalable Interdomain Routing with Heterogeneous Path Requirements*. PhD thesis, University of Southern California, 2004.

[HS00a] G. Hjaltason and H. Samet. Contractive embedding methods for similarity searching in metric spaces. Technical Report TR-4102, Computer Science Department, University of Maryland, 2000.

[HS00b] G. Hjaltason and H. Samet. Contractive Embedding Methods for Similarity Searching in Metric Spaces. Technical Report TR-4102, Computer Science Department, University of Maryland, 2000.

[HWJ05] J. Han, D. Watson, and F. Jahanian. Topology aware overlay networks. In *Proceedings of IEEE INFOCOM*, 2005.

[IRRH03] J. Indulska, R. Robinson, A. Rakotonirainy, and K. Henricksen. Experiences in Using CC/PP in Context-Aware Systems. In

Proceedings of 4th International Conference on Mobile Data Management (MDM), 2003.

[JN04] J. Jin and K. Nahrstedt. Source-Based QoS Service Routing in Distributed Service Networks. In *Proceedings of IEEE International Conference on Communications*, 2004.

[jxt08] JXTA Technology (http://www.sun.com/software/jxta/), 2008.

[kaz08] Kazaa (www.kazaa.com), 2008.

[KBE+07] M. Kleis, K. Büttner, S. Elmoumouhi, G. Carle, and M. Salaün. CSP Cooperative Service Provisioning Using Peer-to-Peer Principles. In *In Proceedings of International Workshop on Self-Organizing Systems (IWSOS)*, 2007.

[Kel04] R. Keller. *Self-Configuring Services for Extensible Networks – A Routing-Integrated Approach.* PhD thesis, Computer Engineering and Networks Laboratory (TIK) ETH Zurich, 2004.

[KK01] T. Korkmaz and M. Krunz. Multi-Constrained Optimal Path Selection. In *Proceedings of IEEE INFOCOM*, 2001.

[KKKD03] R. Kurmanowytsch, E. Kirda, C. Kerer, and S. Dustdar. OMNIX: A Topology-Independent P2P Middleware. In *Proceedings of CAiSE Workshops*, 2003.

[KKKVM04] F. Kuipers, T. Korkmaz, M. Krunz, and P. Van Mieghem. Performance evaluation of constraint-based path selection algorithms. *Network, IEEE*, 18(5):16–23, 2004.

[KMRW02] A. Kameas, I. Mavrommati, D. Ringas, and P. Wason. eComP: An Architecture that Supports P2P Networking Among Ubiquitous Computing Devices. In *Proceedings of the Second IEEE International Conference on Peer-to-Peer Computing*, 2002.

[KSU99] Evangelos Kranakis, Harvinder Singh, and Jorge Urrutia. Compass Routing on Geometric Networks. In *Proceedings of 11th Canadian Conference on Computational Geometry*, 1999.

[Kur04] R. Kurmanowytsch. *Omnix: An Open Peer-to-Peer Middleware Framework*. PhD thesis, Technical University of Vienna, 2004.

[KZ04] M. Kleis and X. Zhou. A Placement Scheme for Peer-to-Peer Networks Based on Principles from Geometry. In *Proceedings of the IEEE Fourth International Conference on Peer-to-Peer Computing (P2P'04)*, 2004.

[LGN07] J. Liang, X. Gu, and K. Nahrstedt. Self-Configuring Information Management for Large-Scale Service Overlays . In *Proceedings of IEEE INFOCOM*, 2007.

[LGP+05] E. K. Lua, T. Griffin, M. Pias, H. Zheng, and J. Crowcroft. On the Accuracy of Embeddings for Internet Coordinate Systems. In *Proceedings of the ACM IMC*, Berkeley,CA, 2005.

[LHC03] H. Lim, J. C. Hou, and C.-H. Choi. Constructing Internet Coordinate System Based on Delay Measurement. In *Proceedings of the ACM IMC*, 2003.

[LLR95] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.

[LN05] J. Liang and K. Nahrstedt. Service composition for advanced multimedia applications. In *Proceedings of SPIE/ACM Multimedia Computing and Networking Conference (MMCN)*, 2005.

[LNS02] J. Liebeherr, M. Nahas, and Weisheng Si. Application-layer multicasting with Delaunay triangulation overlays . *Selected Areas in Communications, IEEE Journal on*, 20(8):1472–1488, Oct 2002.

[MKS06] B. Mathieu, M. Kleis, and M. Song. A P2P Approach for the Selection of Media Processing Modules for Service Specific Overlay Networks. In *Proceedings of International Conference on Internet and Web Applications and Services (ICIW)*, 2006.

[Moy98] J. Moy. OSPF Version 2. RFC 2328 (Proposed Standard), 1998.

[MPZ96] T. Magedanz and R. Popescu-Zeletin. *Intelligent Networks: Basic Technology Standards and Evolution*. Thompson Computer Press, 1996.

[MS04] Y. Mao and L. K. Saul. Modeling Distance in Large-Scale Networks by Matrix Factorization. In *Proceedings of ACM IMC*, October 2004.

[MSG+07] B. Mathieu, M. Song, A. Galis, L. Cheng, K. Jean, R. Ocampo, M. Brunner, M. Stiemerling, and M. Cassini. Self-Management of Context-Aware Overlay Networks for Ambient Networks . In *Proceedings of 10th IFIP/IEEE International Symposium on Integrated Network Management* , 2007.

[NB05] K. Nahrstedt and W. Balke. Towards building large scale multimedia systems and applications: challenges and status. In *Proceedings of the first ACM international workshop on Multimedia service composition*, 2005.

[Net08] CAIDA NetGeo: The Internet geographic database (http://www.caida.org/tools/utilities/netgeo/), 2008.

[NM65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[NZ02] S. Ng and H. Zhang. Predicting Internet Network Distance with Coordinates-Based Approaches. In *Proceedings of IEEE INFOCOM*, 2002.

[ope08] OpenIMP (www.ip-measurement.org/openimp/), 2008.

[OS00] A. Orda and A. Sprintson. QoS Routing: The Precomputation Perspective. In *Proceedings of INFOCOM*, 2000.

[p2p06] P2PSim King Data Set (http://pdos.csail.mit.edu/p2psim/kingdata/), 2006.

[p2p08] P2PSim Project (http://pdos.csail.mit.edu/p2psim/), 2008.

[PCW+03] M. Pias, J. Crowcroft, S. Wilbur, T. Harris, and S. Bhatti. Lighthouses for Scalable Distributed Location. In *Proceedings of the International Workshop on Peer-to-Peer Systems*, 2003.

[PF02] S. Ponnekanti and A. Fox. SWORD: A Developer Toolkit for Web Service Composition. In *Proceedings of 11th World Wide Web Conference (WWW2002 Web Engineering Track)*, 2002.

[PMM93] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. RFC 1546 (Informational), November 1993.

[RAC+02] B. Raman, S. Agarwal, Y. Chen, M. Caesar, W. Cui, P. Johansson, K. Lai, K. Lavian, S. Machiraju, Z. Mao, G. Porter, T. Roscoe, and Mukund. The SAHARA Model for Service Composition Across Multiple Providers. In *Proceedings of International Conference on Pervasive Computing Zurich Switzerland*, 2002.

[Rat02] Sylvia Paul Ratnasamy. *A scalable content-addressable network*. PhD thesis, University of California, Berkeley, 2002. Chair-Scott Shenker and Chair-Ion Stoica.

[RFH+01] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content Addressable Network. In *Proceedings of ACM SIGCOMM*, 2001.

[RHKS02] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologicallyaware overlay construction and server selection. In *Proceedings of IEEE INFOCOM*, 2002.

[Rip01] M. Ripeanu. Peer-to-Peer Architecture Case Study: Gnutella Network. Technical report, University of Chicago, 2001.

[RK03] B. Raman and R. Katz. Load Balancing and Stability Issues in Algorithms for Service Composition. In *Proceedings of IEEE INFOCOM*, 2003.

[RS00] D. Raz and Y. Shavitt. Optimal Partition of QoS requirements with Discrete Cost Functions. In *Proceedings of IEEE INFOCOM*, 2000.

[RS07] A. Riedl and D. Schupke. Routing optimization in IP networks utilizing additive and concave link metrics. *IEEE/ACM Transaction in Networking*, 15(5):1136–1148, 2007.

[RT99] E. Royer and C. Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks. In *Proceedings of IEEE Personal Communications*, 1999.

[SA05] S. Seetharaman and M. Ammar. Overlay-friendly Native Network: A Contradiction in Terms? In *Proceedings of ACM HotNets-IV*, 2005.

[SAZ+02] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet indirection infrastructure. In *Proceedings of ACM SIGCOMM*, 2002.

[SRC84] J. Saltzer, D. Reed, and D. Clark. End-To-End Arguments in System Design. *ACM Transactions on Computer Systems*, 2(4):277–288, November 1984.

[SSAP06] E. Shi, I. Stoica, D. Andersen, and A. Perrig. OverDoSe: A Generic DDoS Protection Service Using an Overlay Network.

Technical Report CMU-CS-06-114, Carnegie Mellon University Computer Science Department, 2006.

[SSBK04] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz. OverQoS: An Overlay based Architecture for Enhancing Internet QoS. In *Proceedings of USENIX*, 2004.

[ST03] Y. Shavitt and T. Tankel. Big-Bang Simulation for Embedding Network Distances in Euclidean Space. In *Proceedings of IEEE INFOCOM*, San Francisco, California, 2003.

[TC03] L. Tang and M. Crovella. Virtual Landmarks for the Internet. In *Proceedings of the ACM IMC*, Miami Beach, FL, 2003.

[Tou01] J. Touch. Dynamic Internet overlay deployment and management using the X-bone. *Comput. Networks*, 36(2-3):117–135, 2001.

[ttm08] Test Traffic Measurement project (http://www.ripe.net/ttm), 2008.

[vlc08] VideoLAN Project (http://www.videolan.org), 2008.

[VMF02] M. Velasco, P. Marti, and J.M. Fuertes. Peer-to-peer communication for virtual industrial devices . In *Proceedings ofFactory Communication Systems 2002. 4th IEEE International Workshop on* , 2002.

[Wan08] F. Wang. Development and evaluation of a P2P service delivery platform for value added multimedia transport. Master's thesis, (Diplomathesis) TU Berlin, 2008.

[WC96] Z. Wang and J. Crowcroft. Quality-of-Service Routing for Supporting Multimedia Applications. *IEEE Journal of Selected Areas in Communications*, 14(7):1228–1234, 1996.

[WLL04] M. Wang, B. Li, and Z. Li. sFlow: Towards Resource-Efficient and Agile Service Federation in Service Overlay Networks. In

Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04), 2004.

[Wro97] J. Wroclawski. The Use of RSVP with IETF Integrated Services. RFC 2210 (Proposed Standard), September 1997.

[wsd08] Web Services Description Language (WSDL) Version 2.0 (http://www.w3.org/TR/wsdl20/), 2008.

[WSS05] B. Wong, A. Slivkins, and E. Sirer. Meridian: A Lightweight Network Location Service Without Virtual Coordinates. In *Proceedings of ACM SIGCOMM*, Philadelphia, PA, 2005.

[WVK+01] B. Whetten, L. Vicisano, R. Kermode, M. Handley, S. Floyd, and M. Luby. Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer. RFC 3048 (Informational), January 2001.

[XN02] D. Xu and K. Nahrstedt. Finding service paths in a media service proxy network. In *Proceedings of the ACM/SPIE Conference on Multimedia Computing and Networking*, 2002.

[XSLB05] Z. Xu, P. Sharma, S.-J. Lee, and S. Banerjee. Netvigator: Scalable Network Proximity Estimation. Technical Report HPL-2004-28R1, HP Laboratories, 2005.

[YST+05] S. Yamaoka, T. Sun, M. Tamai, K. Yasumoto, N. Shibata, and M. Ito. ResourceAware Service Composition for Video Multicast to Heterogeneous Mobile Users. In *Proceedings of 1st International Workshop on Multimedia Service Composition*, 2005.

[ZCB96a] E. W. Zegura, K. Calvert, and S. Bhattacharjee. How to Model an Internetwork. In *Proceedings of IEEE INFOCOM*, 1996.

[ZCB96b] Ellen W. Zegura, Kenneth L. Calvert, and Samrat Bhattacharjee. How to Model an Internetwork. In *Proceedings of IEEE INFOCOM*, 1996.

[ZLPG05] H. Zheng, E. Lua, M. Pias, and T. Griffin. Internet Routing Policies and Round-Trip-Times. In *Proceedings Passive and Active Measurement Conference (PAM)*, 2005.

ISBN 3-937201-05-X
ISSN 1868-2634 (print)
ISSN 1868-2642 (electronic)