M. Brinkmeier, M. Fischer, S. Grau, G. Schäfer, T. Strufe

Methods for Improving Resilience in Communication Networks and P2P Overlays

ABSTRACT

Resilience to failures and deliberate attacks is becoming an essential requirement in most communication networks today. This also applies to P2P Overlays which on the one hand are created on top of communication infrastructures, and therefore are equally affected by failures of the underlying infrastructure, but which on the other hand introduce new possibilities like the creation of arbitrary links within the overlay.

In this article, we present a survey of strategies to improve resilience in communication networks as well as in P2P overlay networks. Furthermore, our intention is to point out differences and similarities in the resilience-enhancing measures for both types of networks.

By revising some basic concepts from graph theory, we show that many concepts for communication networks are based on well-known graph-theoretical problems. Especially, some methods for the construction of protection paths in advance of a failure are based on very hard problems, indeed many of them are in NP and can only be solved heuristically or on certain topologies.

P2P overlay networks evidently benefit from resilience-enhancing strategies in the underlying communication infrastructure, but beyond that, their specific properties pose the need for more sophisticated mechanisms. The dynamic nature of peers requires to take some precautions, like estimating the reliability of peers, redundantly storing information, and provisioning a reliable routing.

1 INTRODUCTION

With the increasing dependence of our modern information society on communication networks and distributed applications, their correct functioning has become essential. However, communication networks as well as overlay structures deployed on top of them may suffer from random failures and/or become target of deliberate attacks. In this context, network resilience – the ability to provide and maintain an acceptable service level in the presence of (random or deliberate) failures – becomes more and more important. A resilient network should be able to cope with a specific amount of failures by remaining completely functional, providing connectivity to all of its parts and providing enough capacity to fulfill its task.

Nevertheless, resilience alone is not sufficient without keeping efficiency in mind, which requires to use all network resources in an efficient manner. Resilience measures usually introduce redundancy into the system, which leads to a decreased efficiency. The more resilient a system should be, the more redundancy has to be applied, and the more the efficiency of the system is decreased. So, there is a trade-off between the two goals and both have to be taken into account when designing a resilient infrastructure.

A networked IT-infrastructure basically consists of connected intermediate nodes, providing a transmission service, and end-nodes, running distributed applications that make use of the transmission service. In this context, errors can be classified into structural failures, like the breakdown of links and nodes, and transmission errors. The latter category is out of the scope of this survey, since measures against transmission errors have already been extensively studied in literature, e.g. simple retransmission methods like ARQ or redundancy introducing codes like FEC [23] or turbo codes [9].

The handling of structural failures requires more complex measures and is the main subject of this survey. Consequences of node or link failures can be packet loss, delayed packets and even partitioning of the network. Packet loss is the result of broken paths, delayed packets are caused by possibly necessary re-routings, and if too many links fail, the network can be divided into two or several isolated parts.

Different applications have different requirements towards network resilience, which can be even contrary to each other. Some applications need a strict timely delivery but may tolerate some loss (e.g. multimedia applications like video streaming), whereas some applications rely on the completeness of delivery without specific timing requirements (e.g. file sharing). So, resilience requirements might be different, depending on the application to be supported. Nevertheless, in order to be able to give a comprehensive overview, this article aims at resilience at a more general level, without a detailed view on specific applications.

In order to achieve resilience, the possibility for detection of failures and their correction is needed. For detection of a failure, some additional effort has to be spend, e.g. periodical heartbeat messages to be sure that a neighbor is alive. After a failure is detected, means for correction have to be provided.

Resilience can be achieved either reactively by ***restoration*** or pro-actively by ***protection*** methods. Restoration, requires a reaction only upon the occurrence of an error. Protection, in contrast, prepares means of correction through additional redundant information before a failure occurs, and often does not even need retransmissions. Therefore, protection usually provides much faster error recovery than restoration, however, requiring more overhead. According to [62], protection and restoration methods usually apply the following steps:

1. *Failure Detection*
2. *Failure Localization (and Isolation)*
3. *Failure Notification*
4. *Recovery (Protection or Restoration)*
5. *Reversion (Normalization)*

Failure Detection is performed by standard mechanisms in routing protocols, as well as *Failure Localization and Isolation*, and *Failure Notification*. More interesting is *Recovery*, which is the main objective of this article. Our survey aims to provide an overview of current recovery approaches to increase resilience in communication networks, as well as in P2P overlay networks. The *Reversion* in the last step, is done by standard mechanisms again and is therefore not treated in detail neither.

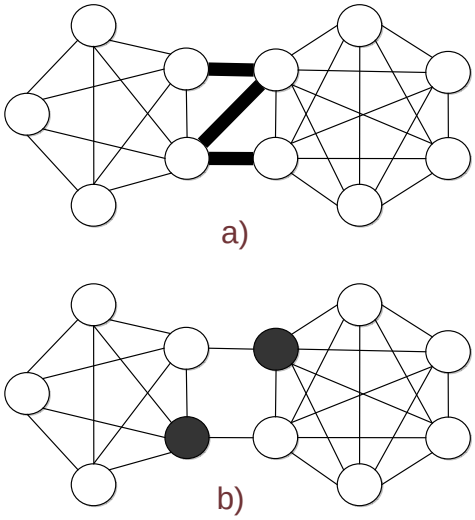

Figure 1 In a) a graph with edge connectivity 3 is given. In b) a graph with vertex connectivity 2 is shown.

Furthermore, we relate the approaches to known concepts of graph theory, and also try to show interactions, similarities and differences between approaches for communication networks and P2P networks.

The rest of this article is organized as follows: in Section 2, background on important graph properties and possible graph classes in communication networks is given. Section 3 presents recovery concepts in infrastructure networks, classified into restoration and protection methods. In Section 4, resilience-enhancing methods for P2P overlay networks are presented and differences to communication networks are identified. Finally, Section 5 summarizes the main findings of the article.

2 GRAPH THEORETICAL BACKGROUND

Any network can be modeled as a (directed) graph G consisting of vertices or nodes V and edges or links E. Edges may be weighted, to either represent communication capacities, or communication costs or delays. This abstract view offers the possibility to study characteristics measuring aspects of resilience, which itself has a very informal and unspecific definition so far. It is also possible to identify certain classes of graphs, showing typical properties in respect to such measures.

Graph theoretic aspects mostly influence the design of protective mechanisms, but can also support restoration.

2.1 Important graph properties

Resilience was defined as the ability to maintain a network service under interference. Since many of these services depend on the reachability of nodes, connectivity measures certainly belong to the most important graph properties.

The ***edge connectivity*** λ and the ***vertex connectivity*** κ are the minimum number of edges (vertices), that need to fail, to separate the graph into at least 2 components and hence are worst-case statistics of resilience. Alternatively interpreted, λ-1 and κ-1 are the number of edges (vertices) which may always be removed, without disconnecting the graph. The edge connectivity equals the size of an (unweighted) minimum cut of the graph and is bounded from above by the minimum degree (i.e. the minimum number of incident edges) of a vertex.

Due to the min-cut-max-flow theorem [35], λ also equals the *minimum number of edge-disjoint paths between any two vertices* in G, and κ is the minimum number of vertex-disjoint paths between any pair of vertices, that are not directly linked by an edge (the latter ones obviously can not be separated by removal of a third vertex). Exactly these edge- or vertex-disjoint paths are a key factor in the protection against structural failures (see Section 3.2).

A graph is called *k-edge-connected* if $\lambda \geq k$, i.e. between every pair of vertices exist at least k edge-disjoint paths. Similar it is called *k-vertex-connected* if $\kappa \geq k$, i.e. between every pair of unconnected vertices there exist at least k vertex-disjoint paths. Edge connectivity augmentation algorithms like [38, 8] can be used to compute the minimum set of additional edges, required to make a graph k-connected. Additionally, the union of k edge-disjoint spanning trees, will result in a k-connected graph, since it is a packing of k paths for every vertex pair. For the augmentation of the vertex connectivity, only algorithms with a running time exponential in the target connectivity are known so far [46].

In the above form, the edge and vertex connectivity are a measure for the resilience of a network against partition, i.e. the minimum number of edges or vertices which have to fail in order to disconnect pairs of vertices. But in many situations, even decreasing the communication capacity between two vertices under a specific value, without disconnecting them completely, is considered a failure. In these cases, it is possible to assign capacities to the edges and the edge-connectivity λ is the minimum sum of capacities of edges crossing a cut (ie. a partition of the vertex set). As in the unweighted case, the min-cut-max-flow theorem ensures, that at least communication capacity c is available between two arbitrary vertices, if the network satisfies $\lambda \geq c$.

Another connectivity related measure is the ***fragmentation*** of a graph. Since, very often the disconnection of a single weakly connected vertex is of minor importance for the whole network, the fragmentation determines a value pair describing the size and relation of its disconnected components. Let $s_1, \ldots, s_c$ be the number of vertices in the c components of the graph, then the value $\text{frag}_1 := \frac{\max_{i=1}^{c} s_i}{\sum_{i=1}^{c} s_i}$ is the relative size of the largest component and the value $\text{frag}_2 := \frac{\sum_{i=1}^{c} s_i - \max_{i=1}^{c} s_i}{c-1}$ represents the average size of the remaining components. Following the discussion above, values near 1 (which is high for frag_1 and low for frag_2) are often seen as advantageous, depending on the type of network service.

If the network services are dependent on short communication paths, especially if delays play a role, a second set of statistics, besides the connectivity metrics, becomes important. Therefore, the degradation behaviour of distance metrics under increasing damage should be studied. Furthermore, these metrics allow the evaluation of scalability of communication within certain topology classes.

The shortest path between two vertices s and t is a set of edges connecting s and t (possibly via intermediate vertices) and having a minimum sum of edge weights. Let the distance $d(s,t)$ be the weight of the shortest s-t-path and the distance between unconnected vertices defined to be infinite. The ***diameter*** of a graph $\text{diam}(G) := max_{s,t \in V} d(s,t)$ then is the length of the longest shortest path between any two vertices. Clearly, the diameter influences the time of information distribution in the whole network.

To get a better view on the whole network, it is also interesting to study the ***average distance*** $\bar{d} := \frac{1}{|V|^2 - |V|} \sum_{s,t \in V} d(s,t)$, i.e. the average length of the shortest path between two vertices of G. However, since this measure jumps to infinity as soon as the graph becomes disconnected, it can be more interesting to look at the ***average connected distance*** $\hat{d}$ regarding only the paths between connected vertices.

When studying overlay networks, we are also interested in the local properties of the graph. The ***clustering coefficient*** $c(v)$ of a vertex v, as introduced by [103], is the fraction of pairs of neighbors of v, which are neighbors themselves. Av-

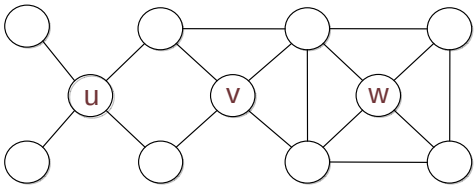

Figure 2 Vertex u has a cluster coefficient of 0, vertex v one of $\frac{1}{3}$ and w has cluster coefficient $\frac{2}{3}$.

eraging this value gives the clustering coefficient of the whole graph $C(G) = \frac{1}{|V|}\sum_{v\in V} c(v)$. One effect of a high clustering coefficient is the tendency to high edge and vertex connectivities. Since many of the neighbors are connected to each other, one has to remove quite a lot of edges to disconnect its neighborhood. Similar, close vertices tend to have a large common neighborhood, leading to a high vertex connectivity.

There is a multitude of further robustness measures for graphs and this section can only present a most important subset. For a thorough study, the reader is referred to [12]. The next section will introduce important graph classes guaranteeing certain limits for the above mentioned properties. As can be seen, clustering and short distances may, but need not, occur together.

2.2 Graph classes of communication networks

Depending on their properties and influenced by underlying building mechanisms, graphs can be categorized into certain classes. Choosing a graph class for the own (overlay) network is one of the most important network design decisions determining resilience and efficiency.

A graph class both simple and easy to construct are the ***random graphs***, where the probability for two vertices to be directly connected to each other is uniform. In many peer-to-peer environments this is often modified by requiring a constant outgoing vertex degree k, such that new vertices receive a list of k other vertices to connect to, chosen uniformly at random. Assuming unit edge-weights, connected random graphs show a low (i.e. logarithmic in the number of vertices) diameter [21] and have a very small clustering coefficient. Due to the expected regularity and the lack of vertices with an outstanding role for the connectivity, they suffer low damage from malicious attacks. More precisely, these attacks have basically the same effect as random node failures: With increasing damage, the network first keeps connected for a long time and then fragments into many small components [5]. The edge- and vertex connectivity of random graphs is with high probability equal to the minimum degree.

Another very popular graph class are ***power-law networks*** [7]. Their properties appear in social and environmental networks as well as the Internet. They are characterized by a power-law distribution of vertex degrees. As Barabasi and Albert have shown, such a property can be reached with the preferential attachment model: Starting with a small clique of initial vertices, new vertices are iteratively added. During this process, each new vertex is connected to k already existing nodes, chosen with a probability proportional to their current vertex degree.

Hence, a small number of nodes have a very high degree, lie on many paths and act as important 'hub' nodes. On the other hand, there are numerous unimportant vertices with low degree. Power-law networks also tend to provide logarithmic diameters. However, the uneven distribution of importance dictates their resilience properties: while being highly robust to random node failure, they heavily suffer from malicious attacks since a systematic removal of the hub nodes leads to a fast fragmentation into many relatively small components and quickly increases the average connected distance [5] (which later rapidly drops, since no more large components exist).

A random vertex removal especially shows good fragmentation behaviour by maintaining one large component for a long time. Another problem, not to be underestimated in peer-to-peer environments with limited resources, are the very high degrees of some nodes.

Finally, a class which is highly relevant for peer-to-peer structures are the ***small-world networks*** [103]. They combine a high clustering coefficient with a logarithmic diameter due to long-range 'shortcut' edges. In that, they offer efficient communication together with a local common neighborhood. This combination supports the ability to make global routing decisions based on local knowledge [49] and enables groups of vertices to locally cooperate, e.g. with tit-for-tat mechanisms. Furthermore, the highly clustered neighborhoods and ability for decentralized routing stimulate local and cooperative restoration mechanisms. The definition of small-world networks includes very different graphs (especially, certain power-law networks too), such that high connectivity and a bounded node degrees have to be enforced by appropriate building mechanisms.

These characteristics are generated by e.g. the Watts-Strogatz model: Beginning with a ring, first connect each vertex to all vertices within distance d on the ring, for some $d \geq \frac{\ln(n)}{2}$. After that, rewire each edge randomly with small, nonzero probability p. If an edge is rewired, one of its vertices is exchanged with a uniformly chosen other vertex. Similar ring structures with shortcuts are often found in structured overlay networks (e.g. Chord [95]). Consequently, these networks show the typical small-world behaviour.

2.3 Routing

The existence of one or more paths between two vertices does not yet control, how a unit of information is eventually forwarded between them. To be able to account for such a concept, we define routing.

In general a ***routing*** on a graph $G = (V,E)$ is a map $R\colon V \times V \to \mathrm{Paths}(G)$, from the set of ordered pairs of vertices to the set $\mathrm{Paths}(G)$ of paths in G, such that $R(u,v)$ is a path from u to v. It is reasonable, to assume that for every vertex v the path $R(v,v)$ has length 0. A ***partial routing*** is a partial map R from a subset of $V \times V$ to $\mathrm{Paths}(G)$. Routings modeled by this definition are static. Dynamic routings can be formalized by several modifications. First, one can introduce time and consider a sequence of routings R_t, one for each point of time. Hence, we have a constant routing at any point of time. This model may be used for adaptive routing algorithms. Secondly, one can assign a probability distribution of the paths from u to v to each pair (u,v), leading to a probabilistic version of routings.

A ***local routing*** assigns a map $R_v : V \setminus v \to \{(v,x) \in E | x \in V\}$ to each vertex v, assigning an edge leaving v to each other vertex. The local routings of all vertices form a path for every vertex pair $(v,w) \in V \times V$ as follows: starting in v, apply the local routing of the current vertex to determine an edge to a next vertex and proceed until either w or an already visited vertex is reached. The set of all those paths is the *routing* of the graph. Of course, this routing may be faulty, since it is not ensured, that the target vertex is reached by this process. To avoid this, often a probabilistic or non-deterministic component is added to local routings.

Assuming mutual reachability of all vertices and the fact, that for every vertex u on the routing path from v to w, the v-u-path is a prefix of the v-w-path, combining all paths starting at v results in a spanning tree of the graph. This spanning tree is called *routing tree of* v. This naturally leads to the notion of a ***tree-based routing*** $R\colon V \to \mathrm{SpanTree}(G)$ onto the set $\mathrm{SpanTree}(G)$ of spanning trees of G.

3 GENERAL RECOVERY AND RESILIENCE-ENHANCING STRATEGIES IN NETWORKING

Resilience in networks is needed to guarantee a proper quality of service in the presence of node and link failures as well as transmission errors. However, the remainder of this article focusses on structural failures, while transmission errors are out of scope of this survey. As already mentioned in the introduction, resilience-enhancing measures can be classified into *Restoration* and *Protection* strategies. In the following Section 3.1, reactive measures for error recovery after a failure has occurred are discussed, and Section 3.2 revises proactive mechanisms which apply effort before a failure occurs.

3.1 Restoration

Restoration mechanisms are reactive and hence applied after failure. For this reason, they offer a higher flexibility to react on failures than protection mechanisms, which are proactive. Furthermore, effort is only caused after a failure has happened, whereas protection causes effort in every case.

Most of the standard routing mechanisms contain restoration methods, in order to be able to react to link- or node failures. Failures are detected and the routing re-converges, excluding the failed element. Nevertheless, this takes some time and at this point data loss may occur. Hence, this approach is not sufficient for sensitive applications that rely on low or bounded delays.

A faster way for restoration can be provided by a rerouting around structural failures until the routing mechanism has reconverged. Therefore, alternative routes are established ad hoc and data is locally rerouted.

The Equal Cost Multipath (ECMP) option, which is used in interior gateway protocols (e.g. OSPF [67]), can be used to distribute traffic via several alternative paths with equal costs to its destination. Similar to ECMP, when several alternative paths are available by the routing itself, a Shortest Path Rerouting (SPR) can be used.

Beyond, the establishment of a new MPLS path [82], after the failure, provides a method for restoration. Yet another approach in packet switched networks is based on IP Fast Reroute [90] and uses “not-via addresses” [60], to exclude a failed element from the data path.

3.2 Protection

In order to protect infrastructure networks against structural failures like node outages or cut links, fallback solutions have to be provided. The straight forward approach to increase the resilience of a transmission infrastructure is to choose a redundant layout by introducing multiple connectivity or multihoming and hence create a network rather than a simple spanning tree over all nodes, thus allowing for a connectivity between nodes in case of link-failures. Further approaches to increase the robustness towards failing nodes or links can consist of redundantly deployed hardware, by safeguarding every router through another one, operating in hot standby, and multiple connectivity between each pair of nodes. The resource demand of these solutions is expensive and they are only feasible for extremely sensitive applications. However, as networks, due to their multiple connectivity, show an inherent redundancy in their connections, virtual fallback solutions can be provided by computing node- or link-disjoint paths between any pair of nodes, thus creating alternative routings. These solutions can be grouped into three classes:

1. alternative global routings
2. locally alternative routings
3. bypass topologies

Alternative global routings are selected in exchange for the initial routing in case of failure [41, 44, 51, 64], so that the path is switched from end-to-end and means for signalling the occurrence of a failure in the network are needed. In an alternative approach [65] the differing paths of the alternative routings are used in parallel. In case of a failure on a path it is avoided and data is sent along remaining paths only. The minimum spanning structures that may be used for alternative global routings are spanning trees. A result of Nash-Williams guarantees, that k edge-disjoint spanning trees can be found in an undirected network of edge-connectivity $2k$ [68].

Locally alternative routings [36, 44, 51, 52] only concern local neighborhoods and are selected in case of the breakdown of a component, in order to circumvent the failure. Basically a locally alternative routing requires the existence of at least two node disjoint paths connecting the two communicating nodes. Hence, a sufficiently high vertex-connectivity is required.

In order to create a detour around failures, the third class of approaches [42] creates an additional routing topology, which can be used to quickly create alternative routes around failed components. In the following examples of this type, usually a spanning or dominating subgraph of connectivity 2 or more is used. Here *spanning* means, that every vertex is member of the subgraph, while a subgraph *dominates* the graph, if every vertex has a distance of at most 1 to the vertices in the dominating subgraph.

Based on the classification given above, in the following, approaches known from various types of networks are presented. Nevertheless, these concepts can be applied to other networks, too.

Linear Automatic Protection Switching Linear Automatic Protection Switching (APS) can be classified as alternative global routing for the protection of a path end-to-end and distributes data redundantly via two bidirectional channels, in order to offer the highest possible protection. The bidirectional channels represent two node- and link-disjoint paths in the network.

Linear APS is part of SONET/SDH [41] and is usually used for high sensitive applications. In packet switched networks a deployment based on MPLS [91, 74] is possible.

Self-Protecting Multi-paths Self-Protecting Multi-paths (SPM) [65] are a method intended to protect paths end-to-end by an alternative global routing. The main idea is to provide k node-disjoint (and hence link-disjoint) paths, whereby the network has to be k-vertex-connected, ie. $\kappa \geq k$ (cmp. Section 2.1). Traffic is distributed via all k paths by a load balancing function like an equal distribution of traffic to all possible paths. In case of a failure, the affected packets are simply shifted to another remaining path.

Since the k node-disjoint paths have to be pre-computed, we run into the Maximum Disjoint Path Problem, which is known to be NP-complete [39, 92]. If the paths are only required to be edge-disjoint, they can be constructed in polynomial time, using standard maximum-flow algorithms.

As stated in [65] a realization of SPM in an IP network can be done by manipulating the link costs in the OSPF-protocol [67] or by using MPLS [91] together with the MPLS Fast Rerouting mechanism [74].

Protection Rings A Protection ring offers alternative global routing protection. The construction of a ring requires to establish a path, which traverses all nodes only once. Thereby, one structural failure can be recovered. For enhanced protection multiple rings can be deployed to counter more than one node or link failure.

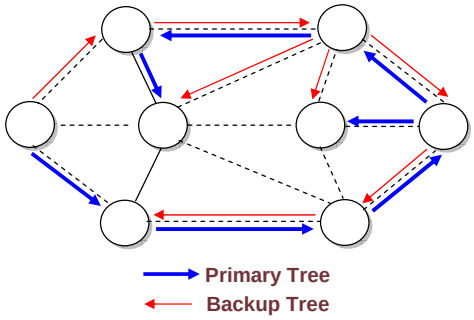

Figure 3 Two trees for the protection against a vertex failure.

In an Unidirectional Path Switched Ring (UPSR) two redundant copies of protected traffic are sent in both directions of the ring, whereas in Bidirectional Line Switched Ring (BLSR) traffic is rerouted in the other direction in the case of a structural failure on the ring.
Such a protection ring is a Hamiltonian cycle, which is one of the classical NP-complete problems [39]. Consequently, this technique may only be applied, if the network has a special structure, like for example an artificially added Hamiltonian cycle.
Examples for the appliance of Protection Rings are circuit switched networks, like SONET/SDH [41]. In [66] three approaches for ring construction in such networks are given. In packet-switched networks a pre-computed ring can be used for protection by a rerouting based on [91, 74, 90, 53].

Redundant Trees Redundant Trees [64] establish an alternative global routing in the network, by the creation of link-disjoint spanning trees. In addition to the main routing tree of all nodes, a second alternative routing tree is established as a backup. They can be applied to every protocol that allows the use of redundancy for the protection from failures and the establishment of tree routing. Even though the approach is based on a centralized algorithm, the authors briefly propose means for a distributed computation.
Tree construction starts at a common source vertex *s* for all trees and tries to include all vertices in the network. In doing so, all constructed trees have to be link-disjoint. Figure 3 shows an example deployment, which allows to tolerate both node and link failures. The thin arrows indicate the primary distribution tree, whereas the fat arrows indicate the backup distribution tree. In case of structural failures on the path of the primary tree, the alternative path in the secondary tree can be chosen for data delivery.
Again, a result of Nash-Williams, guarantees, the existence of k edge-disjoint spanning trees if the network is $2k$-edge-connected [68]. In [83] Roskind and Tarjan described a polynomial time algorithm for the construction of k edge-disjoint spanning trees[1].
A deployment of Redundant Trees for protection in IP networks is described in [22], but can also be done by using the IP Fast Reroute Mechanism [90], by Failure Inferencing-based Fast Rerouting [53] or by MPLS Fast Reroute [74].

Resilient Routing Layers Resilient Routing Layers (RRLs) [44, 51] represent another approach that creates alternative routings. The main idea of RRLs is to find fully connected spanning subgraphs of a given topology, which are labelled *layers*. All different layers in consequence have to contain all nodes and a subset of the links, but need not necessarily be link-disjoint. In addition, there is the restriction that every node v has to be a leaf, ie. connected to only one neighbor, in

[1] In fact they compute k spanning trees with total minimum weight.

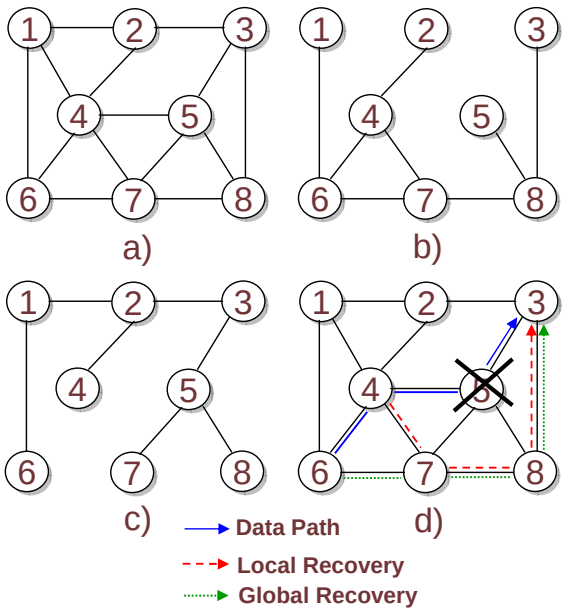

Figure 4 In a) a topology is shown from which two example Resilient Routing Layers b),c) are built. In d) node 5 failed, which can be handled by global (dotted lines) or local recovery (dashed lines) for the original data path from 6 to 3 (solid lines besides topology links).

at least one of the layers. These layers are called *safe* for v. So, v experiences no transit traffic destined for other nodes, except traffic to the connected one.
Under normal circumstances and without the occurrence of an error, ordinary routing across the whole topology (and all links) takes place. As soon as a node failure occurs, the traffic is routed according the safe layer of the failed node. In the presence of link failures, several cases have to be differentiated, whereby we will call the incident node, which formerly received traffic from the broken link, the *downstream node*: If the downstream node is not the traffic destination and the broken link is not its safe link, the safe layer of the downstream node is used. Otherwise, the detecting (upstream) node sends on its own safe layer, if the broken link is not its safe link. If this is additionally the case, it switches the packet to its safe layer as well, however, using another outgoing link, hence, taking advantage of the fact, that its safe layer will not route any traffic back to it.
Note, that the worst-case number of recoverable failures is heavily dependent on the connectivity of the subgraph induced by combining all routing layers. Consequently, although the links of different routing layers do not need to be link-disjoint, such a provision is strongly recommended. Furthermore, for a reasonable application, the underlying graph should have at least a vertex connectivity of two, in order to find enough distinct layers. So, RRLs are no option in sparsely connected topologies.
Figure 4a) shows an example topology in which two different routing layers are established as shown in 4b) and 4c). In Figure 4b) nodes *1,2,3* and *5* are safe, whereas they are transit nodes and therefore not safe in Figure 4c). A failure of one of them in 4c) would disrupt several paths.
In Figure 4d) data is sent from *6* to *3* and traverses the nodes *4* and *5*, as indicated by the solid lines besides the topology links. After the failure of node *5*, global and local recovery methods are possible. In local recovery (dashed lines) no explicit signalisation of the failure to the other nodes is needed. The adjacent node to the failure just switches onto the layer in which node *5* was safe and forwards the traffic. In our example, node *4* switches the traffic to the second layer via node *7*. In global recovery the failure is signalled in the net-

work and the nodes may chose more optimal routes based on this information. In 4d) node *4* signals the failure to *6*, which switches to another layer and forwards the traffic via *7*. A similar approach by the same authors was proposed in [52]. In this publication they describe an improved and faster algorithm for the creation of layers.
RRLs can be created in connectionless IP networks by marking every packet according to the layer of the selected routing, as well as in connection-oriented networks based on MPLS. As already mentioned for the previously described approaches, a deployment in packet switched networks can be done according several mechanisms [91, 74, 90, 53].

Protection Cycles Protection Cycles [36] are basically a Cycle Double Cover (CDC) with additional properties. A CDC is a family of cycles, such that every edge of the graph is in exactly two of them. Furthermore, the cycles of the CDC have to be orientable (OCDC), such that every edge occurs in both possible directions.
Protection Cycles were firstly proposed for optical networks and require a topology with at least two parallel unidirectional working channels, one in each direction, and two parallel backup channels. In case of a failure at a working channel, the reverse backup path is used for rerouting traffic in the reverse direction.
In case of a failure of edge e, the packets destined for forwarding via e, may be redirected along the remainder of the corresponding cycle, containing e in the reverse direction. An edge connectivity of at least two is required, to find a protection cycle for every vertex.
On general 2-edge-connected graphs, the existence of a CDC is far from clear. But it is known that the minimum counterexample for graphs having no CDC has to be of a very specific type. Hence, it is strongly conjectured, that every graph contains a CDC. For details, the interested reader is referred to [36], where a heuristic algorithm for the construction of an OCDC can be found.
Each time a link fails, exactly two protection cycles are involved – one for each direction along the failed link. The failure of an additional link on one of these cycles can not be recovered, since the alternative route is already destroyed. Hence, in the worst case only one link failure can be recovered. But in the best case, every link failure damages two still intact protection cycles. Therefore, if s protection cycles are known, we can handle at most $\lfloor \frac{s}{2} \rfloor$ link failures.
On 2-connected planar graphs, OCDCs are known to exist. In this situation the existence of faces provides a way to construct it.
In an *Eulerian graph* a cycle exists, that traverses every link exactly once. Therewith, it is the counterpart of Hamiltonian rings, which traverse all nodes only once. An Eulerian cycle can be split up in edge-disjoint cycles, resulting in an OCDC, by simply orienting these in both directions. If it exists, an Eulerian cycle can be computed in polynomial time, eg. [37].
As with Protection Rings, the main appliance of Protection Cycles are circuit switched optical networks. Therefore, in [36] also implementation details are given. Furthermore, Protection Cycles can be used to counter node failures, but it is usually out of scope in optical networks.
An appliance to IP networks is possible on the basis of [91, 74, 90, 53] and a suitable topology that corresponds to the one in an optical network according [36].

p-Cycles Pre-configured Cycles (p-Cycles), were firstly proposed in [42] for the protection of optical networks and are similar to Protection Cycles. The authors in [41] showed that p-Cycles have better properties than Protection Cycles in terms of capacity redundancy, application domain and the conceptual basis. Nevertheless, in [45] it is shown that p-Cycles are a more general concept and Protection Cycles can be derived as a limiting case.

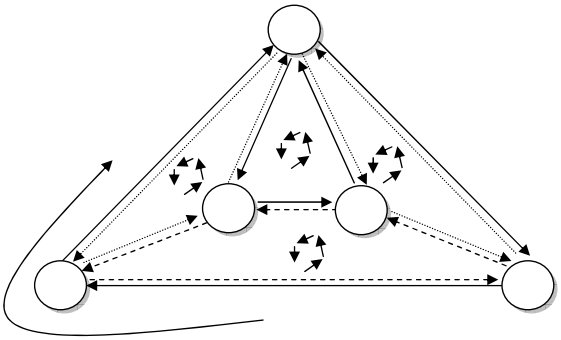

Figure 5 Protection Cycles in a planar graph.

A p-Cycle itself is nothing more than a precomputed logical ring of nodes, protecting against the failure of links incident to two ring nodes (thus not only protecting the links of the ring). In case of a link failure, the adjacent node marks the traffic with its routing distance to the destination and sends it along one intact path on the cycle. The encapsulated packets travel on the p-Cycle until they traverse a node that has a shorter routing distance towards the original destination address (thereby preventing loops). This node decapsulates the traffic and forwards it following the normal routing of the network.
Figure 6a) shows an example. Since p-Cycles have to be computed in advance, they are a protection scheme and can be classified as a bypass topology in which a detour around structural failures is provided by rerouting the traffic on the cycle.
In Figure 6b) a link on the cycle fails and only the remaining direction is left for restoration. In Figure 6c) a straddling link fails respectively. In this case two directions on the ring remain for restoration and it is even possible to split the traffic in both directions.
To protect the whole network, either the p-Cycle should be a Hamiltonian cycle, containing all nodes, or a combination of multiple p-Cycles has to be used. Note, that the computation of a Hamiltonian cycle is an NP-complete problem and therefore, the same applies to the more generalized problem of finding a *minimum* number of cycles, such that for each edge there exists a cycle either containing the edge itself or both end nodes.
For the protection against vertex failures, the concept has to be extended to ***node-encircling p-Cycles*** [93]. As soon as a node fails, an adjacent node routing traffic over it does not know the second-next hop, since the adjacent node has no access to the routing information of the failed node. Furthermore, it does not even know whether the node two hops away is part of the p-Cycle and can be reached via the reverse direction. Hence, to protect from the failure of a node v and to provide restoration for all possible flows, this node v has to be surrounded by a p-Cycle containing all its direct neighbors in the topology, but not v itself.
To protect the whole network, every node is surrounded by one encircling p-Cycle. To ease deployment, it is also possible to apply the weaker concept of region-encircling protection, by encircling all nodes within a certain region. Figure 6d) shows an example. Node *X* is surrounded by a p-Cycle created from adjacent nodes *A,B,C* and non-adjacent node *D* since a ring consisting of only the set of adjacent nodes is not possible.
p-Cycles can be applied in various transmission technologies, like optical networks (WDM, DWDM), networks based

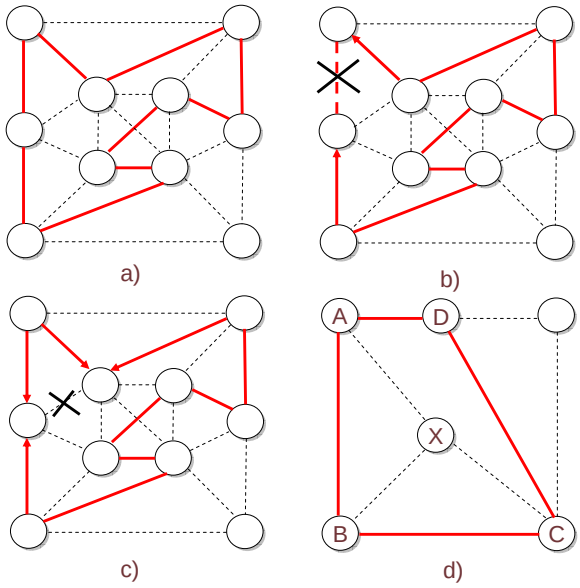

Figure 6 In a) a topology with an example p-Cycle is given. Dashed lines indicate ordinary links and solid lines indicate links used for the p-Cycle. b) shows the failure of an on-cycle link and c) one of a straddling link. In d) another topology with an node-encircling p-Cycle is given in order to protect from a failure of node X.

on virtual connections (e.g. MPLS, GMPLS) or packet-switching [93]. Particularly, in the latter ones, techniques like virtual circuit-like IP tunneling or label switching, e.g. based on MPLS [82], can be applied. p-Cycles can be used in multi-layer networks, like in IP-over-DWDM, and can be even applied in different layers simultaneously. For the creation of p-Cycles, a centralized configuration as well as distributed algorithms [101] are possible.

Table 1 shows a summary of all presented protection schemes, according their class, the underlying graph theoretical problem and the origin of the approach.

4 RESILIENCE IN P2P OVERLAY NETWORKS

Peer-to-Peer is a system architecture that describes a service which is distributed over multiple nodes or processes. While in a client-server architecture the roles are predefined, all participants generally act as both a client and a server in P2P systems. As these participants usually consist of end-hosts, their behaviour, arrival, and departure is not well predictable, and possibly very dynamic.

A common service to all P2P systems is the lookup of information, be it a data resource that is published by another participant or information on existence and addresses of other participants. Here, the communication between peers consists of pairs of requests and replies: a requesting peer takes the role of a client and the requested peer the role of a server. If a requested peer is not able to perform this service, it delegates the request to another peer, thus forwarding the request to a peer that stores the requested information. Additional services, like the registration of information or the distribution of content are commonly implemented on top, in order to create applications.

In order to be able to request information and to route requests in case of the need for delegation, all participants select one or multiple neighbors, thus creating an ***overlay***. Since, due to the dynamic character and the potential size of these overlays, peers can only gather information about a subset of other peers, the main challenge in peer-to-peer systems is to successfully locate information, i.e. to reliably route requests through the overlay to peers that are able to provide the requested information.

Since an overlay network is a network structure built on top of the communication service of an underlying network, special resilience requirements evolve independently from the resilience of both networks seen alone. To study such effects, it is once more helpful to use the terms of graph theory: An *overlay* of a graph $O = (V,E)$ on a communication network $C = (N,L)$ is a pair $M = (M_V, M_E)$ consisting of a map $M_V : V \to N$ of the overlay nodes to nodes of the communication network and a map $M_E : E \to \text{Paths}(C)$, such that $M_E(u,v)$ is a path in C from $M_V(u)$ to $M_V(v)$.

At first view, this definition enforces specific paths on the underlying communication network, and hence a partial routing R_M on C. However, overlays on the application layer usually have no direct influence on the routing of the communication network C. This case is covered, too: assume, that there exists a map $M_V : V \to N$, and a routing R on C. Then we can define the map $M_E : E \to \text{Paths}(C)$, by setting $M_E(u,v) = R(M_V(u), M_V(v))$. Hence, our formal model also covers the case, in which the routing is induced by the underlying communication network C.

A mapping of the overlay, which consists of end-to-end connections onto the underlying communication network is characterized by its congestion and dilation (sometimes also called *path-stretch*): The ***congestion*** K of a graph embedding M is defined as the maximum number of overlay paths traversing an edge of the communication network, i.e. $K(M) := \max_{l \in L} |\{e \in E | l \in M_E(e)\}|$. This notion of congestion corresponds to the notion of congestion in communication networks, in the sense, that an edge with a high congestion, tends to be used by a higher number of packets, than edges of low congestion, implying a high risk of communication congestion.

The ***dilation*** D of an embedding M is the maximum number of edges in any communication path induced by an overlay edge, i.e. $D(M) := \max_{e \in E} |M_E(e)|$.

As the failure of a single communication edge will lead to the breakdown of multiple overlay links, and furthermore, bandwidth exhaustion could appear on the congested edge, high congestion can cause serious problems for the application. Similarly, a high dilation leads to increased communication delays and a higher failure probability of the overlay link, because it is dependent on multiple elements of the underlying network.

Since, usually, the mapping of overlay nodes to communication network nodes cannot be influenced, the only possibility for optimizations lies in considering the underlying communication topology when constructing overlay edges. This includes the addition of edges in the overlay graph O, allowing alternative routings in the underlying network, which may be used to reduce congestion as well as dilation, and may lead to more efficient communication paths.

The main challenge that arises from the decentralized character of peer-to-peer systems is the distribution of the service to end-hosts rather than to dedicated servers and routers. The highly dynamic character of these end-hosts in comparison to dedicated servers requires peer-to-peer systems to take precautions in order to provide a reliable service. These can be classified by their goals into approaches to

- gain an estimation of the reliability of peers
- provide a reliable routing of requests through
  - redundancy in connectivity, information storage or messaging.
  - imposing a structure on the overlay.

4.1 Estimation of reliability

While clients that request a service from a dedicated server will usually trust the entity that provides the resources for the

	Attributes			
Concept	*Class*	*Related Graph Problems*	*Origin*	*References*
Linear APS	global	Maximum Disjoint Paths	optical	[41]
SPM	global	Maximum Disjoint Paths	IP/MPLS	[65]
Protection Rings	global	Hamiltonian Cycle	optical	[41]
Redundant Trees	global	k-Spanning Tree	IP	[64]
RRL	global/local	-	IP	[44, 51]
Protection Cycles	local	Orientable Cycle Double Cover	optical	[36]
p-Cycles	bypass	Hamiltonian Cycle	optical	[42]

Table 1 Comparison of Protection Schemes.

server due to knowledge, which is external to the system, this implicit credibility does not exist in peer-to-peer systems. A similar lack of dedicated resources holds for the forwarding: while in communication networks the roles of routers, as a reliable infrastructure for path selection and forwarding, and end-hosts, which run application level processes, are generally well distinguishable, possibly very unreliable end-hosts have to perform the tasks of routing and forwarding in overlays. Hence, as peers rely on the cooperation of each other, both for delegating requests and performing the information lookup, notions of *reliability* and *trustworthiness* are especially relevant in this environment.
Estimations of the reliability of peers can either be obtained implicitly by analyzing and monitoring basic attributes of peers, or explicitly through the introduction of the notion of a local *image* or distributed *reputation* of peers.

Analyzing Peer's Attributes Depending on the application that is implemented on top of the overlay, a good estimation of the behaviour and the quality of service provided by a peer can be obtained through analysing simple attributes like the time of presence and the available resources. A frequently cited study to model user behavior in filesharing systems [87] has shown that the probability of a peer's departure decreases with increasing time of presence in the overlay. This study has led to a preferential connection to nodes with high uptime in a multitude of systems [63, 58, 14, 98]. Another characteristic shown by the same study is the broad heterogeneity of nodes, which lead to the preference to place information and to select neighbors, based on their resources, thus balancing load for congestion avoidance in the overlay [20, 69, 106, 98]. For implementations of Peer-to-Peer streaming or application layer multicasting (ALM, [81]) this is a natural approach and commonly used to decrease both delays and (random) failure probability [76, 11]. However, it has to be noted that this preferential attachment based on previously obtained information, might also play into the hands of an attacker deliberately choosing nodes to be attacked.

Reputation Systems Increasing the resilience of peer-to-peer systems using reputation mechanisms has been proposed by different research groups. The reputation can be useful for a more exact estimation of the reliability of peers, their credibility and the accuracy of their responses or even for the introduction of punishment of uncooperative or malicious peers (good overviews and discussions on this topic can be found in [86, 15, 54]).
In an early approach to implement a reputation mechanism in peer-to-peer systems, the reputation information is stored in the peer-to-peer system itself [2]. Later approaches save the reputation of peers locally [27, 48, 33, 107, 70, 43].
An essential weakness of reputation systems in decentralized systems is their vulnerability to sybil attacks and collusions [32]: if votes of identities are collected in order to estimate the reputation of a participant, a malicious party can always create multiple identities or form collusions in order to achieve a false, but high reputation. However, certain constraints, design decisions and the characteristic of the service on top of the overlay can increase the robustness of reputation systems towards these threats [31, 84]. Levine et al. [54] define four classes of solutions to the sybil attack: a) by proof of Douceur, the only solution to inhibit sybil attacks is trusted certification using a central certification authority that checks the exact identity of each participant[2]; b) a weak protection is the testing of resources (IP address of identities); c) imposing recurring cost in the form of computational cost, real monetary cost or the demand for time consuming interactions by users can hinder an attacker in creating a high amount of false votes; d) trusted hardware may be used to distinguish between actual hardware devices.

4.2 Reliable Routing

The approaches to provide a reliable routing follow two parallel strategies: the introduction of redundancy and of structure. Due to the lack of global knowledge of both the existence of peers as well as the structure of the overlay, straight forward methods of creating global routings are impossible. Local routings can still be established and the probability of successfully routing messages to requested peers can be increased through the introduction of redundance:

- Redundant connectivity allows for message transmission and the delegation of requests, even if some links or neighboring peers fail.
- Redundant data storage allows for the retrieval of data, even if the primary service providing peer fails or is overloaded.
- Sending redundant requests increases the possibility to successfully find routes to one or different peers that can provide the requested service.

To further increase the routing reliability in a highly dynamic overlay of unknown topology, some structure can be imposed on the overlay, as all peers agree on a common namespace, usually consisting of large random identifiers, and common procedures to create and use the overlay for message delegation.

4.2.1 Introducing Redundancy

Methods that allow to deal with the failure of peers which are expected to forward or serve requests in the overlay, again, are based on redundancy. Due to their dynamic characteristic, they comprise of redundant data storage and redundant requesting on top of creating a redundant layout of the overlay network.

Redundant Connectivity Very similar to the approach of setting up a multiple connectivity in communication networks (cmp. Section 3.2), peers in overlays commonly select a multitude of neighbors. Comparing to the redundant connections in communication networks, the additional links come at a very low cost: as it can be assumed, that all nodes in a

[2] Automatic threshold certification authorities are no solution in this case, as they in general are unable to tell if a request is authentic or the attempt to generate sybil identities. Hence, external checks are necessary.

peer-to-peer system via the underlying communication network can mutually open a connection to each other, the overlay is merely a subset of links of a clique graph that contains all nodes and establishing another link in the overlay does not explicitly imply monetary investment. However, creation and maintenance of links require bandwidth and storage resources, as messaging overhead is needed and the state of the overlay has to be kept. This redundant neighbor selection leads to the possibility to both route requests on short paths and to be able to quickly establish a functional local routing in case of structural failures.

Most peer-to-peer approaches apply this strategy of proactively saving information on possible neighbors for fallback. However, some systems select multiple neighbors only for means of successful routing (e.g. CAN [78] explicitly removes information on nodes, which are no direct neighbors).

Redundant Data Storage A strategy to ensure the location of data is the replication of the lookup service (rather than providing redundant means for the transmission service by introducing multiple connectivity), which leads to a redundant storage of information. This approach is frequently used in conventional client-server based data retrieval services like DNS or webservers, and can also be applied in peer-to-peer systems: In order to be able to retrieve information even after some peers have failed, data is replicated and stored redundantly on different peers.

As a common application on top of the plain routing surrogate of peer-to-peer networks is the storage, retrieval or distribution of data sets, this approach is naturally extended to these applications. Different replication techniques can be distinguished by their data allocation, i.e. the way that peers for the redundant storage are selected.

In some systems [50, 24, 28, 59, 4, 98], the location for the replica is rather random and not explicitly chosen. Hence, data may be implicitly replicated by peers who formerly requested it, by random registration messages or the like. In Freenet [24], for example, data is replicated by a random subset of the peers, that forward it on the path from the replying node back to the requesting peer. The number of replicas of data in consequence depends on its popularity and frequently requested information is replicated with higher probability than information that is rarely requested.

Other approaches implement a deterministic allocation of the replicas, in order to aid their discovery at the time of request delegation. A straight forward strategy for deterministic allocation is the replication on a given number of neighboring nodes in the overlay that is implemented by large number of peer-to-peer substrates [85, 63, 94] and peer-to-peer applications [34, 29, 77]. Since in case of the failure of a service providing peer, the message may still be delegated to one of its previous neighbors, which are now able to provide the requested service. A slight modification of this approach is to select a number of nodes to be responsible for a certain subgroup of services. In PGrid [1] disjoint sets of nodes are selected to provide a fragment of the services, with all nodes of the set replicating the services of each other. A different approach of deterministic allocation is followed by [79, 96], which explicitly choose replicating peers by predetermined mappings. This leads to a higher resilience to correlated failures and the possibility to locate the services on shorter paths through the overlay.

The replication of data again leads to the problem of choosing reliable peers and ensuring its integrity. This is especially relevant for data storage and retrieval applications that are built on top of peer-to-peer substrates, as in their case corruption of data due to errors does not only lead to a temporary unavailability, but a complete break down of the service. Corruption due to adversarial behaviour could even worse lead to the user trusting in data that has been tampered with. Solutions to this problem comprise of reputation systems as presented in 4.1 (wuala[3], [61]), and probabilistic as well as deterministic verification. While probabilistic verification [40, 57, 71, 47, 6] only provides a proof of the integrity of parts of the replicated data, determistic approaches [13, 30, 89, 72] allow for the verification of the complete data at each replicating node.

Redundant Requesting Redundant requesting is the third type of redundancy that helps to increase the reliability of routing in peer-to-peer systems. Message loss and local routing breakdowns due to link or node failures, but also faulty routing due to loops or local optima can cause an unsuccessful delegation of requests. As peer-to-peer systems do not have a notion of a connection between a requesting and a serving peer, these failures can not easily be detected. However, the probability of such incidences can be drastically decreased, by sending multiple requests in parallel, thus reaching the same peer on different paths, or a replication of the requested service.

A straight forward approach to redundant requesting is bounded or global ***flooding*** of requests [25, 98]. Requests are sent and forwarded to all neighboring, except the requesting peer. Using unbounded flooding, every node that is still connected to the network in consequence will receive the message at least once. As unbounded flooding is inefficient and leads to an immense messaging overhead, different improvements have been proposed. Introducing identifiers for the messages and keeping the state of previous searches, and thus avoiding to flood messages multiple times at the same forwarding node, decreases the amount of redundant messages in the overlay. Flooding additionally can be bounded to a preset horizon, by introducing an IP-like time-to-live (TTL) for each message, which then is dropped, after the TTL has expired [98]. Bounding the flooding may decrease the quality of the routing, as it may significantly reduce the fraction of nodes to which the request is forwarded and it hence might not reach a node that is able to deliver the service, as it is behind the search horizon, i.e. external to the flooded subgraph of overlay [105]. Increasing the search horizon in case of unsuccessful searches is a viable countermeasure, which, however, may lead to a high message load in the case that a requested information actually is not available in the overlay.

In order to further decrease the amount of redundant messages other approaches propose to perform ***random walks*** [3, 59, 88, 26] by replicating the message only at the original source of the request and subsequently forwarding it to random neighbors at all requested peers. Thus, too, local optima and faulty routing due to structural failures can be circumvented, while keeping the amount of redundant requests low.

Random walks however lead to higher response times. Furthermore, their success rate heavily depends on the graph on which they are conducted. Usually, a thorough analysis can only be made for specific graph models, like random graphs or some small-world networks (e.g. [49]). In most applications, one has to rely on experimental results.

In systems that replicate services and perform deterministic allocation [79, 96], redundant requests can additionally be ***routed to*** one of the ***replicas***. This not only yields an increased reliability due to the ability to tolerate the failure of the node that originally offered the requested service, but may additionally lead to the possibility to select a replica which can be accessed on a short path through the overlay.

[3] http://www.wua.la/

The strategy of redundant requesting has another flavour with respect to the application: Applications that implement a content distribution scheme, and especially in case of applications that can tolerate a low rate of lost messages (like conferencing, live streaming or video-on-demand services), setting up redundant paths, that comprise of no or only a few common forwarding nodes can significantly decrease the perceived loss due to failures and thus increase the quality of service [18, 96, 97, 55, 99, 73, 80].

4.2.2 Imposing Structure on Overlays

Being able to estimate the reliability of peers and increasing the resilience through different types of redundancy, overlays are at this point perceived as a potentially big graph of unknown topology, spanning a potentially large set of unknown nodes that are connected through unknown links. Imposing structure on the overlays is an orthogonal approach, that aims at providing the possibility to implement a deterministic routing and to achieve a predictable performance of lookups. Overlays can be structured by different policies of neighbor selection on two levels by:

- creating an overlay of a virtual but deterministic topology, and
- optimizing an overlay topology to exhibit desirable characteristics.

Structured Overlays In order to provide means for the implementation of a deterministic routing scheme on the one hand, or in order to be able to achieve highly successful lookups with low delays using stochastic routing schemes on the other hand, different approaches propose to impose different virtual structures on the overlay. These approaches follow one of the three main ideas of a) creating an overlay that resembles a random graph, b) introducing a hierarchy into the previously flat overlay and allocating different roles to nodes, according to their level, or c) making use of the id-space of the nodes and thus implementing a distributed hashtable by organizing all nodes in a predefined overlay structure.

By creating an overlay, which resembles a ***random graph*** and by implementing a replication of information at a preset fraction of completely random nodes, [98] achieves a very high resilience to malfunctions of nodes caused by both failures or attacks, as well as a high reliability due to probabilistic "guarantees".

Another way to reduce the routing complexity and thus achieve a higher reliability of the lookup is to introduce a ***hierarchy*** and allocate roles to the peers accordingly. Hierarchical approaches (fasttrack[4], edonkey[5],[50]) by different means select a subset of all peers as "super nodes" or "super peers", which provide the lookup and routing to the rest of the peers. The super nodes again request and delegate requests by routing in an unstructured network between each other, while the peers of the lower level simply locate one or a set of super nodes and use them in a conventional client-server fashion. This hierarchy leads to a much lower load for the big share of low-level nodes and a much smaller subgraph in which requests are delegated.

Distributed Hashtables (DHT) are the third strategy that imposes a structure on the overlay. A namespace is divided and mapped onto overlay nodes, so that every request is routed towards a node, that has the responsibility for the registration of the area corresponding to the request. The mapping between namespace and nodes is done by a computable function (e.g. a hash function). So, in addition to their underlay address, nodes usually chose an identifier, which can be easily mapped on the identifiers of the provided resources or corresponds to them. In addition, a registration mapping between resources and node identifiers is done, which is based on a distance value. The node, whose node identifier is numerically closest to the resource identifier, stores the resource and all references to replicas of the resource.

The identifiers of nodes are not directly addressable in the network, so the choice of neighbors and the routing have to be adapted towards the namespace. This requires to create a virtual structure above the namespace, that can be a ring [95], a tree [85, 63, 110, 75] or a torus [78]. A routing in these structures requires all nodes to chose their neighbors according the applied virtual structure. Lookups or requests for a specific identifier are routed along the structure towards the responsible node holding the specific registration area in the namespace.

Depending on the chosen structure, upper bounds for the number of hops that are required to route a request can be given. In ring structures the upper bound is at $O(n)$, whereas for balanced d-ary tree structures with $d \geq 2$ the bound is at $O(log\ n)$. For a d-dimensional torus the upper bound lies at $O(d * n^{\frac{1}{d}})$.

Overlay Characteristics In addition, to the chosen structure of the namespace, an overlay can also be categorized by other properties. An important characteristic is the ***locality-awareness*** of the overlay. Overlay nodes can be arbitrarily connected to other overlay nodes, completely independent of the underlying infrastructure network. In this case neighboring nodes in the overlay need not necessarily be neighbors in the underlay and may be located in a high distance from each other, which increases the probability of a failure as the path consists of a high number of comparably unreliable nodes. In a locality-aware overlay the information of neighborhood and routing topology of the underlay is taken into account for the overlay construction. The path stretch, or dilation, is a good metric for the quality of the location awareness of an overlay network. A good overview of P2P substrates that follow this approach is given in [17] and specific approaches can be found in [108, 109, 16, 104]. Including locality information in overlay construction is no direct measure to improve resilience. Nevertheless, it shortens overlay paths, lowers end-to-end latency and consequently decreases the failure probability and the traffic load in the underlying communication networks.

Another property of overlays is the ***power-law*** characteristic as described in Section 2.2. In power-law networks, short overlay paths may be established, since most of the traffic is directed mostly via a few highly connected nodes. The power-law characteristic increases the resilience of a network against random failures, since the probability of the failure of a node with a high number of neighbors is low. However, due to the central role of these highly connected hubs, power-law networks unfortunately are more vulnerable to targeted attacks. Approaches that attempt to create overlays with a power-law character are [3, 88, 96].

Small World, as described in Section 2.2, is a property of networks (e.g. the Internet), which contain short paths between all node pairs. Small-world networks are sparsely connected networks with a huge number of nodes, a low average path stretch and a high clustering coefficient. In many of these networks the degrees are distributed by a power-law, usually causing the low diameter. In the context of overlay networks various approaches exist that attempt to create overlays with small world properties [95, 85, 102, 96]. In addition, these systems can be classified into approaches that focus on the provision of short paths [100, 11, 10, 96], a group of approaches that create inner-node disjoint paths for

[4] http://developer.berlios.de/projects/gift-fasttrack/

[5] http://www.emule-project.net/ , http://www.amule.org/

improved resilience [19, 97, 56, 96], and approaches that attempt to balance the relevance of peers in the network [96] in order to be more resilient against attackers aiming to interrupt the service of highly relevant nodes.

5 SUMMARY

In this article, we surveyed various concepts for improving resilience in communication networks as well as resilience-enhancing strategies and measures in P2P overlay networks. Regarding recovery techniques for communication networks, we classified the current state-of-the-art into alternative global routings, alternative local routings and bypass topologies.

As we have seen, the improvement of the resilience of communication networks gives rise to several algorithmic challenges. Especially the construction of alternative routings for the protection of the communication, often requires the solution of hard problems and most times can only be achieved by heuristics or on special topologies. Furthermore, most of the proposed strategies rely on high edge and/or vertex connectivity. More elaborate metrics, as the fragmentation or the average distances presented in Section 2.1 have not yet been considered in approaches for specific networking or P2P architectures.

P2P overlay networks, realized at application level, require the transmission services of the underlying communication infrastructure, and therefore both benefit from resilience-enhancing measures in the underlying infrastructure. Nevertheless, the specific properties of an P2P overlay network pose the need for more sophisticated resilience-enhancing methods and goes beyond simple rerouting techniques presented in Section 3.

We examined the key properties of P2P overlays that influence resilience without an in-depth view of application specific details, since there is wide variety of different applications employing P2P overlay techniques. A file-sharing application, for example, has other specific demands and puts emphasis on other methods than a multimedia application. One key property, required for all P2P overlays, is the ability to cope with a very dynamic membership, which requires some precautions to provide a reliable service. Measures for improved resilience can be classified in reliability estimations of peers and in the provisioning of a reliable routing. The latter one in turn can be established based on redundancy in connectivity, information storage or messaging and by imposing a structure on the overlay.

REFERENCES

[1] K. Aberer. P-grid: A self-organizing access structure for p2p information systems. In *LNCS: Cooperative Information Systems*, 2001.

[2] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In H. Paques, L. Liu, and D. Grossman, editors, *10th International Conference on Information and Knowledge Management*, pages 310–317. ACM Press, 2001.

[3] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in power-law networks. *Physical Review E*, 64:46135 – 46142, 2001.

[4] A. Adya, W. J. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. R. Douceur, J. Howell, J. R. Lorch, M. Theimer, and R. P. Wattenhofer. Farsite: federated, available, and reliable storage for an incompletely trusted environment. *SIGOPS Oper. Syst. Rev.*, 36:1–14, 2002.

[5] R. Albert, H. Jeong, and A.-L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, July 2000.

[6] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song. Provable data possession at untrusted stores. In *CCS, Proceedings of*, 2007.

[7] A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.

[8] A. A. Benczúr and D. R. Karger. Augmenting undirected edge connectivity in O(n2) time. In *SODA '98: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 500–509, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[9] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. *IEEE International Conference on Communications ICC 93. Geneva*, 2:1064–1070, May 1993.

[10] S. Birrer and F. E. Bustamante. Magellan: performance-based, cooperative multicast. *Web Content Caching and Distribution, 2005. WCW 2005. 10th International Workshop on*, pages 133–143, 12-13 Sept. 2005.

[11] S. Birrer, D. Lu, F. Bustamante, Y. Qiao, and P. Dinda. FatNemo: Building a resilient multi-source multicast fattree. In *9th International Workshop on Web Content Caching and Distribution*, pages 182–196, 2004.

[12] U. Brandes and T. Erlebach, editors. *Network Analysis: Methodological Foundations*, volume 3418 of *Lecture Notes in Computer Science*. Springer, 2005.

[13] G. Caronni and M. Waldvogel. Establishing trust in distributed storage providers. *Third International Conference on Peer-to-Peer Computing, 2003. (P2P 2003). Proceedings.*, pages 128–133, Sept. 2003.

[14] D. Carra and E. W. Biersack. Building a reliable p2p system out of unreliable p2p clients: the case of kad. In *CoNEXT '07: Proceedings of the 2007 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2007. ACM.

[15] R. Cascella. The "value" of reputation in peer-to-peer networks. *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, pages 516–520, Jan. 2008.

[16] M. Castro, P. Druschel, Y. Hu, and A. Rowstron. Topology-aware routing in structured peer-to-peer overlay networks. 2003.

[17] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron. Topology-aware routing in structured peer-to-peer overlay networks. Technical Report MSR-TR-2002-82, Microsoft Research, 2002.

[18] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 298–313, New York, NY, USA, 2003. ACM.

[19] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 298–313, New York, NY, USA, 2003. ACM.

[20] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like p2p systems scalable. In *SIGCOMM, Proceedings of conference on Applications, technologies, architectures, and protocols for computer communications*, 2003.

[21] F. Chung and L. Lu. The average distance in a random graph with given expected degree. *Internet Mathematics*, 1(1):91–114, 2002.

[22] T. Cicic, A. F. Hansen, and O. K. Apeland. Redundant trees for fast ip recovery. In *Broadnets 2007*. IEEE, 2007.

[23] G. C. Clark and J. B. Cain. *Error-Correction Coding for Digital Communications*. Perseus Publishing, 1981.

[24] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *LNCS: Designing Privacy Enhancing Technologies*, 2001.

[25] clip2. The gnutella protocol specification v0.4. `http://rfc-gnutella.sourceforge.net/`, 2002.

[26] B. F. Cooper. An optimal overlay topology for routing peer-to-peer searches. In *LNCS: Middleware 2005*, pages 82 – 101, 2005.

[27] F. Cornelli, E. Damiani, S. D. C. di Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a p2p network. In *World Wide Web, Proceedings of the international conference on*, 2002.

[28] F. M. Cuenca-Acuna, R. P. Martin, and T. D. Nguyen. PlanetP: Using Gossiping and Random Replication to Support Reliable Peer-to-Peer Content Search and Retrieval. Technical Report DCS-TR-494, Department of Computer Science, Rutgers University, 2002.

[29] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with cfs. *SIGOPS Oper. Syst. Rev.*, 35(5), 2001.

[30] Y. Deswarte, J.-J. Quisquater, and A. Saidane. Remote integrity checking. In *IICIS, Proceedings of*, 2004.

[31] J. Dinger and H. Hartenstein. Defending the sybil attack in p2p networks: taxonomy, challenges, and a proposal for self-registration. *The First International Conference on Availability, Reliability and Security. ARES 2006*, pages 8 pp.–, April 2006.

[32] J. R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag.

[33] B. Dragovic, E. Kotsovinos, S. Hand, and P. Pietzuch. Xenotrust: event-based distributed trust management. *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, pages 410–414, Sept. 2003.

[34] P. Druschel and A. Rowstron. Past: a large-scale, persistent peer-to-peer storage utility. *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, pages 75–80, May 2001.

[35] P. Elias, A. Feinstein, and C. Shannon. A note on the maximum flow through a network. *IEEE Transactions on Information Theory*, 2:117–119, December 1956.

[36] G. Ellinas, A. G. Hailemariam, and T. E. Stern. Protection cycles in mesh wdm networks. *Selected Areas in Communications, IEEE Journal on*, 18(10):1924–1937, Oct 2000.

[37] Fleury. Deux problemes de geometrie de situation. *Journal de mathematiques elementaires*, pages 257–261, 1883.

[38] A. Frank. Connectivity augmentation problems in network design. In *Mathematical Programming: State of the Art 1994*, pages 34–63, 1994.

[39] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[40] P. Golle, S. Jarecki, and I. Mironov. Cryptographic primitives enforcing communication and storage complexity. In M. Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2002.

[41] W. Grover. *Mesh-Based Survivable Networks. Options and Strategies for Optical, MPLS, SONET, and ATM Networking*. 2004.

[42] W. Grover and D. Stamatelakis. Cycle-oriented distributed preconfiguration: ring-like speed with mesh-like capacity for self-planning network restoration. *IEEE International Conference on Communications ICC 98*, 1:537–543 vol.1, Jun 1998.

[43] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW: Proceedings of the*, 2004.

[44] A. F. Hansen, A. Kvalbein, T. Cicic, S. Gjessing, and O. Lysne. Resilient routing layers for recovery in packet networks. *International Conference on Dependable Systems and Networks DSN 2005. Proceedings.*, pages 238–247, June-1 July 2005.

[45] H. Huang and J. Copeland. Hamiltonian cycle protection: a novel approach to mesh wdm optical network protection. *IEEE Workshop on High Performance Switching and Routing*, pages 31–35, 2001.

[46] B. Jackson and T. Jordán. Independence free graphs and vertex connectivity augmentation. *J. Comb. Theory, Ser. B*, 94(1):31–77, 2005.

[47] A. Juels and J. Burton S. Kaliski. Pors: proofs of retrievability for large files. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 584–597, New York, NY, USA, 2007. ACM.

[48] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *WWW, Proceedings of the*, 2003.

[49] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000.

[50] T. Klingberg and R. Manfredi. The gnutella protocol specification v0.6. `http://rfc-gnutella.sourceforge.net/`, 2002.

[51] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing, and O. Lysne. Fast recovery from link failures using resilient routing layers. *10th IEEE Symposium on Computers and Communications, ISCC 2005. Proceedings.*, pages 554–560, June 2005.

[52] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing, and O. Lysne. Fast ip network recovery using multiple routing configurations. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, pages 1–11, April 2006.

[53] S. Lee, Y. Yu, S. Nelakuditi, Z.-L. Zhang, and C.-N. Chuah. Proactive vs reactive approaches to failure resilient routing. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1:–186, March 2004.

[54] B. N. Levine, C. Shields, and N. B. Margolin. A survey of solutions to the sybil attack. Technical Report 2006-052, University of Massachusetts Amherst, Amherst, MA, 2006.

[55] J. Liang and K. Nahrstedt. DagStream: Locality Aware and Failure Resilient Peer-to-Peer Streaming. In *Multimedia Computing and Networking*, volume 6071 of *International Society for Optical Engineering proceedings series*, pages 224 – 238, 2006.

[56] J. Liang and K. Nahrstedt. Dagstream: locality aware and failure resilient peer-to-peer streaming. volume 6071. SPIE, 2006.

[57] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard. A cooperative internet backup scheme. In *ATEC: Proceedings of*, 2003.

[58] V. Lo, D. Zhou, Y. Liu, C. GauthierDickey, and J. Li. Scalable supernode selection in peer-to-peer overlay networks. In *Second International Workshop on Hot Topics in Peer-to-Peer Systems*, 2005.

[59] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *16th international conference on Supercomputing*, pages 84–95, 2002.

[60] S. P. M. Shand, S. Bryant. Draft: Ip fast reroute using not-via addresses. February 2008.

[61] P. Maniatis, D. S. H. Rosenthal, M. Roussopoulos, M. Baker, T. Giuli, and Y. Muliadi. Preserving peer replicas by rate-limited sampled voting. In *SOSP, Proceedings of*, 2003.

[62] E. Mannie and D. Papadimitriou. Recovery (protection and restoration) terminology for generalized multi-protocol label switching (gmpls), mar 2006.

[63] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *LNCS: International Workshop on P2P-Systems*, volume 2429, pages 53 – 65. Springer, 2002.

[64] M. Medard, S. G. Finn, R. A. Barry, and R. G. Gallager. Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs. *IEEE/ACM Transactions on Networking*, 7(5):641–652, Oct 1999.

[65] J. M. Michael Menth, Andreas Reifert. Self-protecting multipaths - a simple and resource-efficient protection switching mechanism for mpls networks. *3rd IFIP-TC6 Networking Conference (Networking2004 Athens/Greece)*, 2004.

[66] G. D. Morley and W. D. Grover. Current approaches in the design of ring-based optical networks. *IEEE Canadian Conference on Electrical and Computer Engineering*, 1:220–225, 1999.

[67] J. Moy. Ospf version 2, apr 1998.

[68] C. Nash-Williams. Edge-disjoint spanning trees of finite graphs. *Journal of the Londoin Mathematical Society*, 36:445–450, 1961.

[69] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst, and A. Löser. Super-peer-based routing strategies for rdf-based peer-to-peer networks. In *Proceedings of the World Wide Web Conference*, 2003.

[70] T. Ngan, D. Wallach, and P. Druschel. Incentives-compatible peer-to-peer multicast. In *Proceedings of Economics of Peer-to-Peer Systems*, 2004.

[71] N. Oualha and Y. Roudier. Securing ad hoc storage through probabilistic cooperation assessment. In *WCAN*, 2007.

[72] N. Oualha, M. Önen, and Y. Roudier. A security protocol for self-organizing data storage. In *IFIP-SEC*, 2008.

[73] V. Padmanabhan, H. Wang, and P. Chou. Resilient peer-to-peer streaming. In *11th IEEE International Conference on Network Protocols*, pages 16–27, 2003.

[74] P. Pan, G. Swallow, and A. Atlas. Fast reroute extensions to rsvp-te for lsp tunnels, may 2005.

[75] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 311–320, 1997.

[76] J. Pouwelse, J. Taal, R. Lagendijk, D. Epema, and H. Sips. Real-time video delivery using peer-to-peer bartering networks and multiple description coding. In *Proceedings of the conference on Systems, Man and Cybernetics*, 2004.

[77] V. Ramasubramanian and E. G. Sirer. Beehive: O(1)lookup performance for power-law query distributions in peer-to-peer overlays. In *NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, pages 8–8, Berkeley, CA, USA, 2004. USENIX Association.

[78] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172, 2001.

[79] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172, 2001.

[80] R. Rejaie and A. Ortega. PALS: Peer-to-Peer Adaptive Layered Streaming. In *ACM Network and operating systems support for digital audio and video*, pages 153 – 161, June 2003.

[81] P. Rodriguez, K. W. Ross, and E. W. Biersack. Improving the www: caching or multicast? *Computer Networks and ISDN Systems*, 30:2223 – 2243, 1998.

[82] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture, jan 2001.

[83] J. Roskind and R. Tarjan. A note on finding maximum-cost edge-disjoint spanning trees. *Math. Operations Research*, 10(2):701–708, 1985.

[84] M. Rossberg, T. Strufe, and G. Schäfer. Using Recurring Costs for Reputation Management in Peer-to-Peer Streaming Systems. In *3rd IEEE International Conference on Security and Privacy in Communication Networks (SecureComm)*, 2007.

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

[86] S. Ruohomaa, L. Kutvonen, and E. Koutrouli. Reputation management survey. In *ARES*, 2007.

[87] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Multimedia Computing and Networking*, San Jose, CA, USA, January 2002.

[88] N. Sarshar, P. O. Boykin, and V. P. Roychowdhury. Percolation search in power law networks: making unstructured peer-to-peer networks scalable. In *4th International Conference on Peer-to-Peer Computing*, pages 2–9, 2004.

[89] F. Sebé, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater. Efficient remote data possession checking in critical information infrastructures. *Knowledge and Data Engineering, IEEE Transactions on*, 20:1034 – 1038, 2008.

[90] M. Shand and S. Bryand. Draft: Ip fast reroute framework. Technical report, February 2008.

[91] V. Sharma and F. Hellstrand. Rfc3469: Framework for multi-protocol label switching (mpls)-based recovery, feb 2003.

[92] A. Srinivasan. Improved approximations for edge-disjoint paths, unsplittable flow, and related routing problems. In *FOCS*, pages 416–425, 1997.

[93] D. Stamatelakis and W. D. Grover. Ip layer restoration and network planning based on virtual protection cycles. *IEEE Journal on Selected Areas in Communications*, 18(10):1938–1949, Oct 2000.

[94] M. Steiner, T. En-Najjary, and E. W. Biersack. A global view of kad. In *Proceedings of the IMC*, 2007.

[95] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Aplications. In *ACM Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 149 – 160, September 2001.

[96] T. Strufe. *A Peer-to-Peer-based Approach for the Transmission of Live Multimedia Streams (German: "Ein Peer-to-Peer-basierter Ansatz für die Live-Übertragung multimedialer Daten").* PhD thesis, TU Ilmenau, 2007.

[97] T. Strufe, G. Schäfer, and A. Chang. BCBS: An Efficient Load Balancing Strategy for Cooperative Overlay Live-Streaming. In *IEEE International Congress on Communications (ICC)*, pages 304–309, 2006.

[98] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann. Bubblestorm: resilient, probabilistic, and exhaustive peer-to-peer search. In *SIGCOMM Comput. Commun. Rev.*, 2007.

[99] R. Tian, Q. Zhang, Z. Xiang, Y. Xiong, X. Li, and W. Zhu. Robust and efficient path diversity in application-layer multicast for video streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 15:961– 972, 2005.

[100] D. Tran, K. Hua, and T. Do. Zigzag: an efficient peer-to-peer scheme for media streaming. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 2:1283–1292 vol.2, March-3 April 2003.

[101] D. S. W. D. Grover. Self-organizing closed path configuration of restoration capacity. *Broadband Mesh Transport Networks,Proc. CCBR'98*, 1998.

[102] S. Wang, D. Xuan, and W. Zhao. Analyzing and enhancing the resilience of structured peer-to-peer systems. *Journal of Parallel and Distributed Computing*, 65:207 – 219, 2005.

[103] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[104] R. Wouhaybi and A. Campbell. Phenix: supporting resilient low-diameter peer-to-peer topologies. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 1:–119, March 2004.

[105] B. Yang and H. Garcia-Molina. Improving search in peer-to-peer networks. *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 5–14, 2002.

[106] B. Yang and H. Garcia-Molina. Designing a super-peer network. In *Data Engineering, Proceedings of the international conference on*, 2003.

[107] B. Yu, M. P. Singh, and K. Sycara. Developing trust in large-scale peer-to-peer systems. In *Multi-Agent Security and Survivability*, 2004.

[108] Y. Yu, S. Lee, and Z.-L. Zhang. Leopard: A locality aware peer-to-peer system with no hot spot. *R. Boutaba et al. (Eds.): NETWORKING 2005, LNCS 3462, pp. 27-39*, 2005.

[109] X. Y. Zhang, Q. Zhang, Z. Zhang, G. Song, and W. Zhu. A construction of locality-aware overlay network: moverlay and its performance. *IEEE Journal on Selected Areas in Communications*, 22(1):18–28, Jan. 2004.

[110] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, UC Berkeley, April 2001.