Survey of Research towards Robust Peer-to-Peer Networks: Search Methods

JOHN RISSON AND TIM MOORS

University of New South Wales

The pace of research on peer-to-peer (P2P) networking in the last five years warrants a critical survey. P2P has the makings of a disruptive technology - it can aggregate enormous storage and processing resources while minimizing entry and scaling costs. The key to realizing this potential in applications other than file sharing is robustness.

The entire body of P2P research can be divided into four groups - search, storage, security and applications. Although we briefly elaborate on this overall taxonomy, the main focus of this survey is P2P search methods. We summarize and compare P2P indexing methods, from distributed hash tables to systems for keyword lookup, information retrieval and data management. We canvass the early work to optimize various types of queries over P2P indexes. Throughout, we emphasize robustness considerations and identify open research issues. Our ambition here is to baseline research on P2P search methods and assist efforts toward more dependable, adaptable P2P systems.

RISSON, J., MOORS, T. Survey of research towards robust peer-to-peer networks: search methods, *Technical Report UNSW-EE-P2P-1-1, University of New South Wales, Sydney, Australia*, (September 2004), *awaiting publication.*

Authors addresses: J.Risson and T.Moors, Dept. of Electrical Engineering, University of New South Wales, Sydney, New South Wales, 2052, Australia, e-mail: jr@tuffit.com, t.moors@unsw.edu.au

Table of Contents

1. Introduction

Peer-to-peer (P2P) networks are those that exhibit three characteristics: self-organization, symmetric communication and distributed control (Roussopoulos, Baker et al. 2004). A self-organizing P2P network "automatically adapts to the arrival, departure and failure of nodes" (Rowstron and Druschel 2001a). Communication is symmetric in that peers act as both clients and servers. It has no centralized directory or control point. USENET servers or BGP peers have these traits (Yeager and Bhattacharjee 2003) but the emphasis here is on the flurry of research since 2000. Leading examples include Gnutella (Klingberg and Manfredi 2002), Freenet (Clarke 1999), Pastry (Rowstron and Druschel 2001a), Tapestry (Zhao, Kubiatowicz et al. 2001), Chord (Stoica, Morris et al. 2001), the Content Addressable Network (CAN) (Ratnasamy, Francis et al. 2001), pSearch (Tang, Xu et al. 2002b) and Edutella (Nejdl, Decker et al. 2003). Some have suggested that peers are inherently unreliable (Aberer and Hauswirth 2001). Others have assumed well-connected, stable peers (Zhou and van Renesse 2004).

This critical survey of P2P academic literature is warranted, given the intensity of recent research. At the time of writing, one research database lists over 5,800 P2P publications ("Citeseer Scientific Literature Digital Library" 2004). One vendor surveyed P2P products and deployments (Milojicic, Kalogeraki et al. 2002). There is also a tutorial survey of leading P2P systems (Aberer and Hauswirth 2002). DePaoli and Mariani recently reviewed the dependability of some early P2P systems at a high level (DePaoli and Mariani 2004). The need for a critical survey was flagged in the peer-to-peer research group of the Internet Research Task Force (IRTF) (Yeager 2003).

The entire body of P2P research literature can be divided along four lines: search, security, storage and applications. The search and security classifications have been used by Daswani et al (Daswani, Garcia-Molina et al. 2003). In Figure 1, we elaborate on this taxonomy and give a representative list of papers. The list is by no means comprehensive. In this survey, we concentrate on search methods, considering semantic and semantic-free ways of indexing information in a P2P network. We also canvass early work to optimize various query types over P2P indexes. Storage, security and application aspects are deferred.

Our initial objective for this paper was to answer the question, "How robust are P2P search networks?" A good proportion of the paper first addresses the supporting question, "How do P2P search networks work?" This foundation is important given the pace and breadth of research. Where it is available, we emphasize work on robustness and flush out promising lines of investigation.

P2P is potentially a disruptive technology with numerous applications, but this potential will not be realized unless it is demonstrated to be robust. A massively distributed search technique may yield numerous practical benefits for applications (Balakrishnan, Kaashoek et al. 2003). A P2P system has potential to be more dependable than architectures relying on a small number of centralized servers. It has potential to evolve better from small configurations - the capital outlays for high performance servers can be reduced and spread over time if a P2P assembly of general purpose nodes is used. A similar argument motivated the deployment of distributed databases – one thousand, off-the-shelf PC processors are more powerful and much less expensive than a large mainframe computer (Kossmann 2000). Storage and processing can be aggregated to achieve massive scale. Wasteful partitioning between servers or clusters can be avoided. As Gedik and Liu put it, if P2P is to find its way into applications other than simple file sharing, then reliability needs to be addressed (Gedik and Liu 2003b).

1.1. Related Disciplines

Peer-to-peer research draws upon numerous distributed systems disciplines. Networking researchers will recognize familiar issues of naming, routing and congestion control. P2P designs need to address routing and security issues across network region boundaries (Sollins 2003). Networking research has traditionally been host-centric. The web's Universal Resource Identifiers are naturally tied to specific hosts, making object mobility a challenge (Walfish, Balakrishnan et al. 2004).

Taxonomy	Selected References
Search	(Yang and Garcia-Molina 2002b; Yang and Garcia-Molina 2002a; Yang and Garcia-Molina 2002c; Balakrishnan, Kaashoek et al. 2003; Bawa, Sun et al. 2003; Cooper and Garcia-Molina 2003b; Daswani, Garcia-Molina et al. 2003; Gummadi, Gummadi et al. 2003; Hellerstein 2003; Huebsch, Hellerstein et al. 2003; Shi, Guangwen et al. 2004)
Semantic-Free Indexes *Plaxton Trees* *Rings* *Tori* *Butterflies* *de Bruijn Graphs* *Skip Graphs*	(Devine 1993; Litwin, Niemat et al. 1993; Litwin, Neimat et al. 1996; Karger, Lehman et al. 1997; Plaxton, Rajaraman et al. 1997; Rowstron and Druschel 2001a; Stoica, Morris et al. 2001; Zhao, Kubiatowicz et al. 2001; Harvey, Dunagan et al. 2002; Li and Plaxton 2002; Malkhi, Naor et al. 2002; Maymounkov and Mazieres 2002; Ratnasamy, Shenker et al. 2002; Zhao, Duan et al. 2002; Aberer, Cudre-Mauroux et al. 2003; Aspnes and Shah 2003; Cates 2003; Gupta, Birman et al. 2003; Harvey, Jones et al. 2003a; Kaashoek and Karger 2003; Loguinov, Kumar et al. 2003; Rhea, Roscoe et al. 2003; Stoica, Morris et al. 2003; Ganesan, Krishna et al. 2004; van Renesse and Bozdog 2004; Zhao, Huang et al. 2004)
Semantic Indexes *Keyword Lookup* *Peer Information Retrieval* *Peer Data Management*	(Clarke, Sandberg et al. 2001; Jovanovic 2001; Babaoglu, Meling et al. 2002; Kalogaraki, Gunopulos et al. 2002; Klingberg and Manfredi 2002; Lv, Cao et al. 2002; Lv, Ratnasamy et al. 2002; Ripeanu, Iamnitchi et al. 2002; Schlosser, Sintek et al. 2002; Sunaga, Takemoto et al. 2002; Bawa, Manku et al. 2003; Chawathe, Ratnasamy et al. 2003; Joseph and Hoshiai 2003; Nejdl, Siberski et al. 2003; Tatarinov, Mork et al. 2003; Yang and Garcia-Molina 2003; Zhang, Shi et al. 2003; Cai and Frank 2004; Loo, Huebsch et al. 2004b; Tempich, Staab et al. 2004)
Search *Range Queries* *Multi-Attribute Queries* *Join Queries* *Aggregation Queries* *Continuous Queries* *Recursive Queries* *Adaptive Queries*	(Litwin, Neimat et al. 1994; Litwin and Neimat 1996; Avnur and Hellerstein 2000; Aberer 2002b; Andrzejak and Xu 2002; Harren, Hellerstein et al. 2002; Hodes, Czerwinski et al. 2002; Aberer, Datta et al. 2003; Albrecht, Arnold et al. 2003; Aspnes and Shah 2003; Bhagwan, Varghese et al. 2003; Cai, Frank et al. 2003; Daswani, Garcia-Molina et al. 2003; Gedik and Liu 2003b; Gedik and Liu 2003a; Gupta, Agrawal et al. 2003; Harvey, Jones et al. 2003a; Hellerstein 2003; Huebsch, Hellerstein et al. 2003; Montresor, Jelasity et al. 2003; Ratnasamy, Francis et al. 2003; Triantafillou and Pitoura 2003; Tsangou, Ndiaye et al. 2003; van Renesse, Birman et al. 2003; Zhang, Shi et al. 2003; Albrecht, Arnold et al. 2004; Aspnes, Kirsch et al. 2004; Bawa, Gionis et al. 2004; Bharambe, Agrawal et al. 2004; Ganesan, Bawa et al. 2004; Jelasity, Kowalczyk et al. 2004; Loo, Huebsch et al. 2004a; Ramabhadran, Ratnasamy et al. 2004; Schmidt and Parashar 2004; Tanin, Harwood et al. 2004; van Renesse and Bozdog 2004; Yalagandula and Dahlin 2004)
Storage	
Consistency & Replication *Eventual consistency* *Trade-offs…*	(Lomet 1996; Cohen and Shenker 2002b; Cohen and Shenker 2002a; Weatherspoon and Kubiatowicz 2002b; Geels and Kubiatowicz 2003; On, Schmitt et al. 2003; Gopalakrishnan, Silaghi et al. 2004b) (Kubiatowicz, Bindel et al. 2000; Rhea, Wells et al. 2001; Rowstron and Druschel 2001b; Adya, Wattenhofer et al. 2002; Lin, Lian et al. 2004)
Distribution *Epidemics, Bloom Filters*	(Bloom 1970; Bailey 1975; Demers, Greene et al. 1987; Gupta, Birman et al. 2001; Hodes, Czerwinski et al. 2002; Mohan and Kalogaraki 2002; Weatherspoon and Kubiatowicz 2002a; Aberer, Cudre-Mauroux et al. 2003; Birman 2003; Costa, Migliavacca et al. 2003; Eugster, Guerraoiu et al. 2003; Ganesh, Kermarrec et al. 2003; Gupta, Birman et al. 2003; Koloniari, Petrakis et al. 2003; Koloniari and Pitoura 2003; van Renesse, Birman et al. 2003; Vogels, Renesse et al. 2003; Voulgaris and van Steen 2003; Birman and Gupta 2004; Costa, Migliavacca et al. 2004; Eugster, Guerraoiu et al. 2004; Gupta 2004; Koloniari and Pitoura 2004)
Fault Tolerance *Erasure Coding* *Byzantine Agreement …*	(Byers, Considine et al. 2002; Rodrigues, Liskov et al. 2002; Weatherspoon and Kubiatowicz 2002b; Weatherspoon, Moscovitz et al. 2002; Castro, Rodrigues et al. 2003; Cates 2003; Maymounkov and Mazieres 2003; Naor and Wieder 2003b; Plank, Atchley et al. 2003; Krohn, Freedman et al. 2004)
Locality	(Gummadi, Saroui et al. 2002; Hildrum, Kubiatowicz et al. 2002; Keleher, Bhattacharjee et al. 2002; Li and Plaxton 2002; Ng and Zhang 2002; Zhao, Duan et al. 2002; Freedman and Mazieres 2003; Gummadi, Gummadi et al. 2003; Harvey, Jones et al. 2003b; Massoulie, Kermarrec et al. 2003; Pias, Crowcroft et al. 2003; Ruhl 2003; Sollins 2003; Awerbuch and Scheideler 2004a; Cox, Dabek et al. 2004; Dabek, Cox et al. 2004; Dabek, Li et al. 2004; Fessant, Handurukande et al. 2004; Hildrum, Krauthgamer et al. 2004; Karger and Ruhl 2004a; Liu, Liu et al. 2004; Lua, Crowcroft et al. 2004; Mislove and Druschel 2004; Zhang, Zhang et al. 2004)
Load Balancing	(Aberer, Datta et al. 2003; Adler, Halperin et al. 2003; Baquero and Lopes 2003; Byers, Considine et al. 2003; Kaashoek and Karger 2003; Rao, Lakshminarayanan et al. 2003; Ruhl 2003; Aspnes, Kirsch et al. 2004; Castro, Lee et al. 2004; Gao and Steenkiste 2004; Gopalakrishnan, Silaghi et al. 2004a; Karger and Ruhl 2004b; Karger and Ruhl 2004c; Manku 2004; Stavrou, Rubenstein et al. 2004; Wang, Zhang et al. 2004)
Security	
Character *Identity* *Reputation and Trust* *Incentives*	(Damiani, di Vimercati et al. 2002; Blaze, Feigenbaum et al. 2003; Buragohain, Agrawal et al. 2003; Caronni and Waldvogel 2003; Schneidman and Parkes 2003; Anagnostakis and Greenwald 2004; Feldman, Lai et al. 2004; Marti, Ganesan et al. 2004; Papaioannou and Stamoulis 2004; Selcuk, Uzun et al. 2004; Sieka, Kshemkalyani et al. 2004)
Goals *Availability* *Authenticity* *Anonymity* *Access Control* *Fair Trading*	(Clarke, Sandberg et al. 2001; Daswani and Garcia-Molina 2002; Fiat and Saia 2002; Freedman and Morris 2002; Hazel and Wiley 2002; Serjantov 2002; Sit and Morris 2002; Bawa, Sun et al. 2003; Cox and Noble 2003; Daswani, Garcia-Molina et al. 2003; Ngan, Wallach et al. 2003; Ramaswamy and Liu 2003; Singh and Liu 2003; Surridge and Upstill 2003; Berket, Essiari et al. 2004; Feldman, Papadimitriou et al. 2004; Josephson, Sirer et al. 2004; O'Donnel and Vaikuntanathan 2004)
Applications	(Li 2002; Considine, Walfish et al. 2003; Karp, Ratnasamy et al. 2004; Roussopoulos, Baker et al. 2004)
Memory *File Systems* *Web* *Content Delivery Networks* *Directories* *Service Discovery* *Publish / Subscribe ..*	(Dabek, Kaashoek et al. 2001; Gold and Tidhar 2001; Kan and Faybishenko 2001; Annexstein, Berman et al. 2002; Kangasharju, Ross et al. 2002; Muthitacharoen, Morris et al. 2002b; Muthitacharoen, Morris et al. 2002a; Saroiu, Gummadi et al. 2002b; Mislove, Post et al. 2003; Fessant, Handurukande et al. 2004) (Iyer, Rowstron et al. 2002; Bawa, Bayardo et al. 2003; Li, Loo et al. 2003; Freedman, Freudenthal et al. 2004; Loo, Krishnamurthy et al. 2004) (Cuenca-Acuna, Peery et al. 2002; Junginger and Lee 2004; van Renesse and Bozdog 2004) (Balazinska, Balakrishnan et al. 2002; Chander, Dawson et al. 2002; Cox, Muthitacharoen et al. 2002; Hodes, Czerwinski et al. 2002; Iamnitchi 2003; Awerbuch and Scheideler 2004b; Walfish, Balakrishnan et al. 2004)
Intelligence *GRID* *Security…*	(Hoschek 2002; Iamnitchi, Foster et al. 2002; Foster and Iamnitchi 2003; Lo, Zappala et al. 2004) (Ajmani, Clarke et al. 2002; Aberer, Datta et al. 2004)
Communication *Multicasting* *Streaming Media* *Mobility* *Sensors …*	(Zhuang, Zhao et al. 2001; Castro, Druschel et al. 2002; Halepovic and Deters 2002; Lienhart, Holliman et al. 2002; Ratnasamy, Karp et al. 2002; Stoica, Adkins et al. 2002; Castro, Druschel et al. 2003; Demirbas and Ferhatosmanoglu 2003; Hefeeda, Habib et al. 2003; Ratnasamy, Karp et al. 2003; Sasabe, Wakamiya et al. 2003; van Renesse, Birman et al. 2003; Voulgaris and van Steen 2003; Zhang and Hu 2003; Hellerstein and Wang 2004; Hsieh and Sivakumar 2004; Nicolosi and Mazieres 2004; Padmanabhan, Wang et al. 2004; Sit, Dabek et al. 2004; Tran, Hua et al. 2004; Wawrzoniak, Peterson et al. 2004; Zhou and van Renesse 2004)

Figure 1 Classification of P2P Research Literature

P2P work is data-centric (Shenker 2003). P2P systems for dynamic object location and routing have borrowed heavily from the distributed systems corpus. Some have used replication, erasure codes and Byzantine agreement (Rhea, Wells et al. 2001). Others have used epidemics for durable peer group communication (Gupta, Birman et al. 2003).

Similarly, P2P research is set to benefit from database research (Gribble, Halevy et al. 2001). Database researchers will recognize the need to reapply Codd's principle of physical data independence, that is, to decouple data indexes from the applications that use the data (Hellerstein 2003). It was the invention of appropriate indexing mechanisms and query optimizations that enabled data independence. Database indexes like B+ trees have an analog in P2P's distributed hash tables (DHTs). Wide-area, P2P query optimization is a ripe, but challenging, area for innovation.

More flexible distribution of objects comes with increased security risks. There are opportunities for security researchers to deliver new methods for availability, file authenticity, anonymity and access control (Daswani, Garcia-Molina et al. 2003). Proactive and reactive mechanisms are needed to deal with large numbers of autonomous, distributed peers. To build robust systems from cooperating but self-interested peers, issues of identity, reputation, trust and incentives need to be tackled.

Possibly the largest portion of P2P research has majored on basic routing structures (Balakrishnan, Kaashoek et al. 2003), where research on algorithms comes to the fore. Should the overlay be "structured" or "unstructured"? Are the two approaches competing or complementary? Comparisons of the "structured" approaches – hypercubes, rings, toroids, butterflies, de Bruijn and skip graphs – have weighed the amount of routing state per peer and the number of links per peer against overlay hop-counts. While "unstructured" overlays initially used blind flooding and random walks, overheads usually trigger some structure, for example super-peers and clusters.

P2P applications rely on cooperation between these disciplines. Applications have included file sharing, directories, content delivery networks, email, distributed computation, publish-subscribe middleware, multicasting, and distributed authentication. Which applications will be suited to which structures? Are there adaptable mechanisms which can decouple applications from the underlying data structures? What are the criteria for selection of applications amenable to a P2P design (Roussopoulos, Baker et al. 2004)?

Robustness is emphasized throughout the survey. We are particularly interested in two aspects. The first, dependability, was a leading design goal for the original Internet (Clark 1988). It deserves the same status in P2P. The measures of dependability are well established: reliability, a measure of the mean-time-to-failure (MTTF); availability, a measure of both the MTTF and the mean-time-to-repair (MTTR)[1]; maintainability; and safety (Laprie 1995). The second aspect is the ability to accommodate variation in outcome, which one could call adaptability. Its measures have yet to be defined. In the context of the Internet, it was only recently acknowledged as a first class requirement (Clark, Wroclawski et al. 2002). In P2P, it means planning for the tussles over resources and identity. It means handling different kinds of queries and accomodating changeable application requirements with minimal intervention. It means "organic scaling" (Huebsch, Hellerstein et al. 2003), whereby the system grows gracefully, without a priori data center costs or architectural breakpoints.

In the following section, we discuss one notable omission from the taxonomy of P2P networking in Figure 1 - routing.

1.2. Structured and Unstructured Routing

P2P routing algorithms have been classified as "structured" or "unstructured". Early instantiations of Gnutella were unstructured – keyword queries were flooded widely (Clip2 2000). Napster ("Napster") had decentralized content and a centralized index, so only partially satisfies the distributed control criteria for P2P systems. Early structured algorithms included Plaxton, Rajaraman and Richa (Plaxton, Rajaraman et al. 1997), Pastry (Rowstron and Druschel 2001a), Tapestry (Zhao, Huang et al. 2004), Chord (Stoica, Morris et al. 2001) and the Content Addressable Network (Ratnasamy, Francis et al. 2001). Mishchke and Stiller recently classified P2P systems by the presence or absence of structure in routing tables and network topology (Mishchke and Stiller 2004).

Some have cast unstructured and structured algorithms as competing alternatives. Unstructured approaches have been called "first generation", implicitly inferior to the "second generation" structured algorithms (Rowstron and Druschel 2001a; Zhao, Huang et al. 2004). When generic key lookups are required, these structured, key-based routing schemes can guarantee location of a target within a bounded number of hops (Hellerstein 2003). The broadcasting unstructured approaches, however, may have large routing costs, or fail to find available content (Huebsch, Hellerstein et al. 2003). Despite the apparent advantages of structured P2P, several research groups are still pursuing unstructured P2P. There have been two main criticisms of structured systems (Chawathe, Ratnasamy et al. 2003). Firstly, highly transient peers are not well supported by DHTs. One objection to this is that many applications use peers in well-connected parts of the

[1] Traditionally, availability = MTTF / (MTTF + MTTR)

network. Others also opined that it is possible to maintain a robust DHT at relatively low cost (Li and Sollins 2003). Several structured P2P papers have presented some defense of performance during "churn", when peers are continually arriving and departing. For example, the Tapestry authors analysed the impact of churn in a network of 1000 nodes (Zhao, Huang et al. 2004). Very few papers have quantitatively *compared* the resilience of structured systems. Loguinov, Kumar et al claimed that there were only two such works (Gummadi, Gummadi et al. 2003; Loguinov, Kumar et al. 2003). Transient peers motivate the question of correctness: "when many joins and leaves happen concurrently, it is not clear whether the neighbor tables will remain in a 'good' state" (Li and Plaxton 2002).

The second main criticism of structured systems is that they don't support keyword searches and complex queries as well as unstructured systems. Given the current file-sharing deployments, keyword searches seem more important than exact-match key searches in the short term. Paraphrased, "most queries are for hay, not needles" (Chawathe, Ratnasamy et al. 2003).

More recently, some have justifiably seen unstructured and structured proposals as complementary, not competing (Castro, Costa et al. 2004). Their starting point was the observation that unstructured flooding or random walks are inefficient for data that is not highly replicated across the P2P network. Structured graphs can find keys efficiently, irrespective of replication. Castro et al proposed Structella, a hybrid of Gnutella built on top of Pastry (Castro, Costa et al. 2004). Another design used structured search for rare items and unstructured search for massively replicated items (Loo, Huebsch et al. 2004b).

However, the "structured versus unstructured routing" taxonomy is becoming less useful, for two reasons, Firstly, most "unstructured" proposals have evolved and incorporated structure. Consider the classic "unstructured" system, Gnutella (Klingberg and Manfredi 2002). For scalability, its peers are either ultrapeers or leaf nodes. This hierarchy is augmented with a query routing protocol whereby ultrapeers receive a hashed summary of the resource names available at leaf-nodes. Between ultrapeers, simple query broadcast is still used, though methods to reduce the query load here have been considered (Singla and Rohrs 2002). Secondly, there are emerging schema-based P2P designs (Nejdl, Siberski et al. 2003), with super-node hierarchies and structure within documents. These are quite distinct from the structured DHT proposals.

Given that most, if not all, P2P designs today assume some structure, a more instructive taxonomy would describe the structure. In this survey, we use a database taxonomy in lieu of the networking taxonomy, as suggested by Hellerstein, Cooper and Garcia-Molina (Cooper and Garcia-Molina 2003a; Hellerstein 2003). The structure is determined by the type of index. Queries feature in lieu of routing. The DHT algorithms implement a "semantic-free index" (Walfish, Balakrishnan et al. 2004). They are oblivious of whether keys represent document titles, meta-data, or text. Gnutella-like and schema-based proposals have a "semantic index".

1.3. Indexing and Searching

Index engineering is at the heart of P2P search methods. It captures a broad range of P2P issues, as demonstrated by the Search/Index Links (SIL) model (Cooper and Garcia-Molina 2003a). As Manber put it, "the most important of the tools for information retrieval is the index—a collection of terms with pointers to places where information about documents can be found" (Manber 1999). Sen and Wang noted that a "P2P network" usually consists of connections between hosts for application-layer signaling, rather than for the data transfer itself (Sen and Wang 2004). Similarly, we concentrate on the "signaled" indexes and queries.

Our focus here is the dependability and adaptability of the search network. Static dependability is a measure of how well queries route around failures in a network that is normally fault-free. Dynamic dependability gives an indication of query success when nodes and data are continually joining and leaving the P2P system. An adaptable index accommodates change in the data and query distribution. It enables data independence, in that it facilitates changes to the data layout without requiring changes to the applications that use the data (Hellerstein 2003). An adaptable P2P system can support rich queries for a wide range of applications. Some applications benefit from simple, semantic-free key lookups (Balakrishnan, Shenker et al. 2003). Others require more complex, SQL-like queries to find documents with multiple keywords, or to aggregate or join query results from distributed relations (Huebsch, Hellerstein et al. 2003).

2. Index Types

A P2P index can be *local*, *centralized* or *distributed*. With a local index, a peer only keeps the references to its own data, and does not receive references for data at other nodes. The very early Gnutella design epitomized the local index (Section 2.1). In a centralized index, a single server keeps references to data on many peers. The classic example is Napster (Section 2.2). With distributed indexes, pointers towards the target reside at several nodes. One very early example is Freenet (Section 2.3). Distributed indexes are used in most P2P designs nowadays – they dominate this survey.

P2P indexes can also be classified as *non-forwarding* and *forwarding*. When queries are guided by a non-forwarding index, they jump to the node containing the target data in a single hop. There have been semantic and semantic-free one-hop schemes (Rodrigues, Liskov et al. 2002; Gupta, Liskov et al. 2003; Yang, Vinograd et al. 2004). Where scalability to a massive number of peers is required, these schemes have been extended to two-hops (Mizrak, Cheng et al. 2003; Gupta, Liskov et al. 2004). More common are the forwarding P2Ps where the number of hops varies with the total number of peers, often logarithmically. The related tradeoffs between routing state, lookup latency, update bandwidth and peer churn are critical to total system dependability.

2.1. Local Index

P2Ps with a purely local data index are becoming rare. In such designs, peers flood queries widely and only index their own content. They enable rich queries – the search is not limited to a simple key lookup. However, they also generate a large volume of query traffic with no guarantee that a match will be found, even if it does exist on the network. For example, to find potential peers on the early instantiations of Gnutella, 'ping' messages were broadcast over the P2P network and the 'pong' responses were used to build the node index. Then small 'query' messages, each with a list of keywords, are broadcast to peers which respond with matching filenames (Klingberg and Manfredi 2002).

There have been numerous attempts to improve the scalability of local-index P2P networks. Gnutella uses fixed time-to-live (TTL) rings, where the query's TTL is set less than 7-10 hops (Klingberg and Manfredi 2002). Small TTLs reduce the network traffic and the load on peers, but also reduce the chances of a successful query hit. One paper reported, perhaps a little too bluntly, that the fixed "TTL-based mechanism does not work" (Lv, Cao et al. 2002) To address this TTL selection problem, they proposed an expanding ring, known elsewhere as iterative deepening (Yang and Garcia-Molina 2002a). It uses successively larger TTL counters until there is a match. The flooding, ring and expanding ring methods all increase network load with duplicated query messages. A random walk, whereby an unduplicated query wanders about the network, does indeed reduce the network load but massively increases the search latency. One solution is to replicate the query k times at each peer. Called random k-walkers, this technique can be coupled with TTL limits, or periodic checks with the query originator, to cap the query load (Lv, Cao et al. 2002). Adamic, Lukose et al. suggested that the random walk searches be directed to nodes with higher degree, that is, with larger numbers of inter-peer connections (Adamic, Lukose et al. 2001). They assumed that higher-degree peers are also capable of higher query throughputs. However without some balancing design rule, such peers would be swamped with the entire P2P signaling traffic. In addition to the above approaches, there is the 'directed breadth-first' algorithm (Yang and Garcia-Molina 2002a). It forwards queries within a subset of peers selected according to heuristics on previous performance, like the number of successful query results. Another algorithm, called probabilistic flooding, has been modeled using percolation theory (Banaei-Kashani and Shahabi 2003).

Several measurement studies have investigated locally indexed P2Ps. Jovanovic noted Gnutella's power law behaviour (Jovanovic 2001). Sen and Wang compared the performance of Gnutella, Fasttrack ("KaZaa Media Desktop, Sharman Networks Ltd" 2001) and Direct Connect (Sen and Wang 2002; Sen and Wang 2004) ("Direct Connect")[2]. At the time, only Gnutella used local data indexes. All three schemes now use distributed data indexes, with hierarchy in the form of Ultrapeers (Gnutella), Super-Nodes (FastTrack) and Hubs (Direct Connect). It was found that a very small percentage of peers have a very high degree and that the total system dependability is at the mercy of such peers. While peer up-time and bandwidth were heavy-tailed, they did not fit well with the Zipf distribution. Fortunately for Internet Service Providers, measures aggregated by IP prefix and Autonomous System (AS) were more stable than for individual IP addresses. A study of University of Washington traffic found that Gnutella and Kazaa together contributed 43% of the university's total TCP traffic (Saroiu, Gummadi et al. 2002a). They also reported a heavy-tailed distribution, with 600 external peers (out of 281,026) delivering 26% of Kazaa bytes to internal peers. Furthermore, objects retrieved from the P2P network were typically three orders of magnitude larger than web objects – 300 objects contributed to almost half of the total outbound Kazaa bandwidth. Others reported Gnutella's topology mismatch, whereby only 2-5% of P2P connections link peers in the same AS, despite over 40% of peers being in the top 10 ASes (Ripeanu, Iamnitchi et al. 2002). Together these studies underscore the significance of multimedia sharing applications. They motivate interesting caching and locality solutions to the topology mismatch problem.

These same studies bear out one main dependability lesson: total system dependability may be sensitive to the dependability of high degree peers. The designers of Scamp translated this observation to the design heuristic, "have the degree of each node be of nearly equal size" (Massoulie, Kermarrec et al. 2003). They analyzed a system of N peers, with mean degree c.log(N), where link failures occur independently with probability ϵ. If $\delta>0$ is fixed and $c>(1+\delta)/(-\log(\varepsilon))$ then the probability of graph disconnection goes to zero as $N\to\infty$. Otherwise, if $c<(1-\delta)/(-\log(\varepsilon))$ then the probability of disconnection goes to one as $N\to\infty$. They presented a localizer, which finds approximate minima to a global function of peer degree and arbitrary link costs using only local information. The Scamp overlay construction algorithms could support any of the flooding and walking routing schemes above, or other epidemic and multicasting schemes for that matter. Resilience to high churn rates was identified for future study.

[2] Bearshare and Limewire clients use Gnutella. KaZaa and Grokster clients use FastTrack. When Sen and Wang wrote their 2002 paper, Morpheus also used FastTrack.

2.2. Central Index

Centralized schemes like Napster ("Napster" 1999) are significant because they were the first to demonstrate the P2P scalability that comes from separating the data index from the data itself. Ultimately 36 million Napster users lost their service not because of technical failure, but because the single administration was vulnerable to the legal challenges of record companies (Loo 2003).

There has since been little research on P2P systems with central data indexes. Such systems have also been called 'hybrid' since the index is centralized but the data is distributed. Yang and Garcia-Molina devised a four-way classification of hybrid systems (Yang and Garcia-Molina 2001): *unchained* servers, where users whose index is on one server do not see other servers' indexes; *chained* servers, where the server that receives a query forward it to a list of servers if it does not own the index itself; *full replication*, where all centralized servers keep a complete index of all available metadata; and *hashing*, where keywords are hashed to the server where the associated inverted list is kept. The unchained architecture was used by Napster, but it has the disadvantage that users do not see all indexed data in the system. Strictly speaking, the other three options illustrate the distributed data index, not the central index. The chained architecture was recommended as the optimum for the music-swapping application at the time. The methods by which clients update the central index were classified as *batch* or *incremental*, with the optimum determined by the query-to-login ratio. Measurements were derived from a clone of Napster called OpenNap (Scholl 2001). Another study of live Napster data reported wide variation in the availability of peers, a general unwillingness to share files (20-40% of peers share few or no files), and a common understatement of available bandwidth so as to discourage other peers from sharing one's link (Saroiu, Gummadi et al. 2002b).

Influenced by Napster's early demise, the P2P research community may have prematurely turned its back on centralized architectures. Chawathe, Ratnasamy et al. opined that Google and Yahoo demonstrate the viability of a centralized index. They argued that "the real barriers to Napster-like designs are not technical but legal and financial" (Chawathe, Ratnasamy et al. 2003). Even this view may be a little too harsh on the centralized architectures – it implies that they always have an upfront capital hurdle that is steeper than for distributed architectures. The closer one looks at scalable 'centralized' architectures, the less the distinction with 'distributed' architectures seems to matter. For example, it is clear that Google's designers consider Google a distributed, not centralized, file system (Ghemawat, Gobioff et al. 2003). Google demonstrates the scale and performance possible on commodity hardware, but still has a centralized master that is critical to the operation of each Google cluster. Time may prove that the value of emerging P2P networks, regardless of the centralized-versus-distributed classification, is that they smooth the capital outlays and remove the single points of failure across the spectra of scale and geographic distribution.

2.3. Distributed Index

An important early P2P proposal for a distributed index was Freenet (Clarke 1999; Clarke, Sandberg et al. 2001; Clarke, Miller et al. 2002). While its primary emphasis was the anonymity of peers, it did introduce a novel indexing scheme. Files are identified by low-level "content-hash" keys and by "secure signed-subspace" keys which ensure that only a file owner can write to a file while anyone can read from it. To find a file, the requesting peer first checks its local table for the node with keys closest to the target. When that node receives the query, it too checks for either a match or another node with keys close to the target. Eventually, the query either finds the target or exceeds time-to-live (TTL) limits. The query response traverses the successful query path in reverse, positing a new routing table entry (the requested key and the data holder) at each peer. The insert message similarly steps towards the target node, updating routing table entries as it goes, and finally stores the file there. Whereas early versions of Gnutella used breadth-first flooding, Freenet uses a more economic depth-first search (Mache, Gilbert et al. 2002).

An initial assessment has been done of Freenet's robustness. It was shown that in a network of 1000 nodes, the median query path length stayed under 20 hops for a failure of 30% of nodes. While the Freenet designers considered this as evidence that the system is "surprisingly robust against quite large failures" (Clarke, Sandberg et al. 2001), the same datapoint may well be outside meaningful operating bounds. How many applications are useful when the first quartile of queries have path lengths of several hundred hops in a network of only 1000 nodes, per Figure 4 of (Clarke, Sandberg et al. 2001)? To date, there has been no analysis of Freenet's dynamic robustness. For example, how does it perform when nodes are continually arriving and departing?

There have been both criticisms and extensions of the early Freenet work. Gnutella proponents acknowledged the merit in Freenet's avoidance of query broadcasting (Rohrs 2002). However, they are critical on two counts: the exact file name is needed to construct a query; and exactly one match is returned for each query. P2P designs using DHTs, per Section 3, share similar characteristics – a precise query yields a precise response. The similarity is not surprising since Freenet also uses a hash function to generate keys. However, the query routing used in the DHTs has firmer theoretical foundations. Another difference with DHTs is that Freenet will take time, when a new node joins the network, to build an index that facilitates efficient query routing. By the inventor's own admission, this is damaging for a user's first impressions (Clarke 2003). It was proposed to download a copy of routing tables from seed nodes at startup, even though the new node might be far from the seed node. Freenet's slow startup motivated Mache, Gilbert et al. to amend the overlay after failed requests and to place additional index entries on successful requests – they claim almost an order of magnitude reduction in average query path length (Mache, Gilbert et al. 2002). Clarke also highlighted the lack of locality or bandwidth information available for efficient query routing decisions (Clarke 2003). He proposed that each node gather response times, connection times and proportion of successful requests for each entry in the query routing table. When searching for a key that is not in its own routing table, it was proposed to estimate response times from the routing metrics for the nearest known keys and consequently choose the node that can retrieve the data fastest. The response time heuristic assumed that nodes close in the key space have similar response times. This assumption stemmed from early deployment observations that Freenet peers seemed to specialize in parts of the keyspace – it has not been justified analytically. Kronfol drew attention to Freenet's inability to do keyword searches (Kronfol 2002). He suggested that peers cache lists of weighted keywords in order to route queries to documents, using Term Frequency Inverse Document Frequency (TFIDF) measures and inverted indexes (Section 4.2.1). With these methods, a peer can route queries for simple keyword lists or more complicated conjunctions and disjunctions of keywords. Robustness analysis and simulation of Kronfol's proposal remains open.

The vast majority of P2P proposals in following sections rely on a distributed index.

3. Semantic-Free Index

Many of today's distributed network indexes are *semantic*. The semantic index is human-readable. For example, it might associate information with other keywords, a document, a database key or even an administrative domain. It makes it easy to associate objects with particular network providers, companies or organizations, as evidenced in the Domain Name System (DNS). However, it can also trigger legal tussles and frustrate content replication and migration (Walfish, Balakrishnan et al. 2004).

Distributed Hash Tables (DHTs) have been proposed to provide *semantic-free*, data-centric references. DHTs enable one to find an object's persistent key in a very large, changing set of hosts. They are typically designed for (Hellerstein 2003):

a) low degree. If each node keeps only a small number of transport connections to other nodes, the impact of high node arrival and departure rates is contained;

b) low diameter. The hops and delay introduced by the extra indirection are minimized;

c) greedy routing. Nodes independently calculate a short path to the target. At each hop, the query moves closer to the target; and

d) robustness. A path to the target can be found even when links or nodes fail.

To understand the origins of recent DHTs, one needs to look to three contributions from the 1990s. The first two, Plaxton Trees (Plaxton, Rajaraman et al. 1997) and Consistent Hashing (Karger, Lehman et al. 1997), were published within one month of each other. The third, the Scalable Distributed Data Structure (SDDS) (Litwin, Niemat et al. 1993), was curiously ignored in significant structured P2P designs despite having some similar goals (Rowstron and Druschel 2001a; Stoica, Morris et al. 2001; Zhao, Kubiatowicz et al. 2001). It has been briefly referenced in other P2P papers (Gribble, Brewer et al. 2000; Aberer 2002a; Malkhi, Naor et al. 2002; Honicky and Miller 2003; Manku 2003).

The Plaxton Tree is the most recent of the three. It influenced the designs of Pastry (Rowstron and Druschel 2001a), Tapestry (Zhao, Kubiatowicz et al. 2001), Chord (Stoica, Morris et al. 2001) and Oceanstore (Kubiatowicz, Bindel et al. 2000). The value of the Plaxton Tree is that it can locate objects using fixed-length routing tables (Zhao, Kubiatowicz et al. 2001). Objects and nodes are assigned a semantic-free address, for example a 160 bit key. Every node is effectively the root of a spanning tree. A message routes toward an object by matching longer address suffixes, until it encounters either the object's root node or another node with a 'nearby' copy. It can route around link and node failure by matching nodes with a related suffix. The scheme has several disadvantages (Zhao, Kubiatowicz et al. 2001): global knowledge is needed to construct the overlay; an object's root node is a single point of failure; nodes cannot be inserted and deleted; there is no mechanism for queries to avoid congestion hot spots.

Karger et al. introduced Consistent Hashing in the context of the web caching problem (Karger, Lehman et al. 1997). Web servers could conceivably use standard hashing to place objects across a network of caches. Clients could use the approach to find the objects. For normal hashing, most object references would be moved when caches are added or deleted. On the other hand, Consistent Hashing is "smooth" – when caches are added or deleted, the minimum number of object references move so as to maintain load balancing. Consistent Hashing also ensures that the total number of caches responsible for a particular object is limited. Whereas Litwin's Linear Hashing (LH*) scheme requires 'buckets' to be added one at a time in sequence, Consistent Hashing allows them to be added in any order (Karger, Lehman et al. 1997). There is an open Consistent Hashing problem pertaining to the fraction of items moved when a node is inserted (Karger and Ruhl 2004b). Extended Consistent Hashing was recently proposed to randomize queries over the spread of caches to significantly reduce the load variance (Lei and Grama 2004). Interestingly, Karger (Karger, Lehman et al. 1997) referred to an older DHT algorithm "using a novel autonomous location discovery algorithm that learns the buckets' locations instead of using a centralized directory" (Devine 1993).

In turn, Devine's primary point of reference was Litwin's work[3] on SDDSs and the associated LH* algorithm (Litwin, Niemat et al. 1993). An SDDS satisfies three design requirements: files grow to new servers only when existing servers are well loaded; there is no centralized directory; the basic operations like insert, search and split never require atomic updates to multiple clients. Honicky and Miller suggested the first requirement could be considered a limitation since expansion to new servers is not under administrative control (Honicky and Miller 2003). Litwin recently noted numerous similarities and differences between LH* and Chord (Litwin 2004a). He found that both implement key search. Although LH* refers to clients and servers, nodes can operate as peers in both. Chord 'splits' nodes when a new node is inserted, while LH* schedules 'splits' to avoid overload. Chord requests travel O(logN) hops, while LH* client requests need at most two hops to find the target. Chord stores a small number of 'fingers' at each node. LH* servers store N/2 to N addresses while LH* clients store 1 to N addresses. This tradeoff between hop count and the size of the index affects system robustness, and bears striking similarity to recent one- and two-hop P2P schemes in Section 2. The arrival and departure of LH* clients does not disrupt LH* server metadata at all. Given the size of the index, the arrival and departure of LH* servers is likely to cause more churn than that of Chord nodes. Unlike Chord, LH* has a single point of failure, the split coordinator. It can be replicated. Alternatively it can be removed in later LH* variants, though details have not been progressed for lack of practical need (Litwin 2004a). Given that the SDDS work has continued in parallel with recent structured P2P efforts (Litwin 2004), a quantitative comparison of system dependability, scalability, and configuration effort would be worthwhile.

In the same vein, comprehensive comparisons across the range of recent structured P2P proposals are needed. The multiplicity of proposals and design choices has frustrated such efforts. The following paragraphs summarize the very few comparisons. It appears unlikely that the P2P community will settle on a small number of designs in the near term. Comparisons could fall into three categories: *fault-free performance*, a pre-failure measure; *static dependability* as a measure of performance before recovery mechanisms take over; and *dynamic dependability* for the most likely case in massive networks where there is continual failure and recovery.

For fault-free performance, Christin and Chuang developed a model for latency, service, routing and maintenance costs, and compared results for De Bruijn graphs, D-dimensional tori like CAN, Plaxton trees and Chord rings (Christin and Chuang 2003; Christin and Chuang 2004). Contrary to P2P mantra, they suggested that well-connected, central peers may minimize total system costs. They did not identify clear winners amongst the structured P2P geometries - several improvements were suggested to first make the cost models realistic. Rhea et al. presented an empirical performance comparison of Tapestry and Chord (Rhea, Roscoe et al. 2003). They deferred robustness measures, like the cost of nodes continually arriving and departing, since these are more difficult than performance measures. For large files, they found little difference in lookup latencies. For small files, Tapestry had lower lookup latency than Chord – it was conjectured that Tapestry's recursive lookups are faster than Chord's interative lookups. It was also found that Tapestry's algorithm to find nearest replicas gives it an advantage over Chord, but still falls short of the ideal.

For static dependability, a comparison of the tree, hypercube, butterfly, ring, XOR and hybrid geometries prompted Gummadi et al. to ask "Why not the ring?" – their preliminary results suggested that it offers the best flexibility and resilience (Gummadi, Gummadi et al. 2003). Loguinov et al. compared Chord, CAN and de Bruijn graphs, finding that de Bruijn graphs offer the optimal diameter and connectivity (Loguinov, Kumar et al. 2003). They opined that the classical failure analyses, for example the probability that a particular node becomes disconnected, yield no major differences between the resilience of these three graphs. Using bisection width (the minimum edge count between two equal partitions) and path overlap (the likelihood that backup paths will encounter the same failed nodes or links as the primary path), they argued for the superior resilience of the de Bruijn graph. Hsiao and King compared Chord, Tapestry, Pastry and Tornado in terms of adaptivity, the number of alternatives available for the next hop (Hsiao and King 2003).

Li et al. pointed that such fault-free and static dependability comparisons are biased in favour of algorithms with more state at each node (Li, Stribling et al. 2004). Their dynamic dependability comparison reported lookup latency (msec) against average live bandwidth (bytes/node/sec) for Chord, Kelips, Tapesty and Kadlemia. The bandwidth metric was used on the premise that the bandwidth is more constraining than memory or processing capacity. It incorporated costs of maintaining state during churn. The latency/bandwidth tradeoff was found to be sensitive to the specific design parameters of each algorithm, so the optimal curve for each was compared using a discrete event packet simulator (Gil, Kaashoek et al. 2003). While all four schemes had comparable performance for high

[3] Both Litwin and Devine were at UC-Berkeley in 1993.

bandwidth, Chord was found to have lower latency at low bandwidths. The lookup rate and node departure rate were quite slow, exponentially distributed with respective means of ten minutes and one hour. The authors plan to isolate the impact of specific design choices in each algorithm and simulate other workloads.

In addition to these performance and resilience considerations, formal development rigour may help P2P robustness. Some designers claim provable correctness (Stoica, Morris et al. 2001; Maymounkov and Mazieres 2002; Hildrum, Kubiatowicz et al. 2003). Others expressed concern that "when many joins and leaves happen concurrently, it is not clear whether the neighbor tables will remain in a 'good' state" (Li and Plaxton 2002). While acknowledging that guaranteeing consistency in a failure prone network is impossible, Lynch, Malkhi et al. sketched amendments to the Chord algorithm to guarantee atomicity (Lynch, Malkhi et al. 2002). More recently, Gilbert, Lynch et al. gave a new algorithm for atomic read/write memory in a churning distributed network, suggesting it to be a good match for P2P (Gilbert, Lynch et al. 2004). Lynch and Stoica show in an enhancement to Chord that lookups are provably correct when there is a limited rate of joins and failures (Lynch and Stoica 2004). A good starting point for a formal P2P development would be the numerous informal API specifications (Awerbuch and Scheideler 2003; Dabek, Zhao et al. 2003; Huebsch, Hellerstein et al. 2003). Such work could be informed by other efforts to formally specify routing invariants (Feamster and Balakrishnan 2003; Ahlgren, Brunner et al. 2004), replication invariants (Gedik and Liu 2003b) and liveness/safety invariants in distributed publish/subscribe systems (Muhl 2002).

In this section, we introduce the main geometries for simple key lookup and survey their primary robustness mechanisms.

3.1. Plaxton Trees

Work began in March 2000 on a structured, fault-tolerant, wide-area Dynamic Object Location and Routing (DOLR) system called Tapestry (Zhao, Kubiatowicz et al. 2001; Hildrum, Kubiatowicz et al. 2002). While DHTs fix replica locations, a DOLR API enables applications to control object placement (Zhao, Huang et al. 2004). Tapestry's basic location and routing scheme follows Plaxton, Rajaraman and Richa (PRR) (Plaxton, Rajaraman et al. 1997), but it remedies PRR's robustness shortcomings described previously. Whereas each object has one root node in PRR, Tapestry uses several to avoid a single points of failure. Unlike PRR, it allows nodes to be inserted and deleted. Whereas PRR required a total ordering of nodes, Tapestry uses 'surrogate routing' to *incrementally* choose root nodes. The PRR algorithm does not address congestion, but Tapestry can put object copies close to nodes generating high query loads. PRR nodes only know of the nearest replica, whereas Tapestry nodes enable selection from a set of replicas (for example to retrieve the most up to date). To detect routing faults, Tapestry uses TCP timeouts and UDP heartbeats for detection, sequential secondary neighbours for rerouting, and a 'second chance' window so that recovery can occur without the overhead of a full node insertion. Tapestry's dependability has been measured on a testbed of about 100 machines and on simulations of about 1000 nodes. Successful routing rates and maintenance bandwidths were measured during instantaneous failures and ongoing churn (Zhao, Huang et al. 2004).

Pastry, like Tapestry, uses Plaxton-like prefix routing (Rowstron and Druschel 2001a). As in Tapestry, Pastry nodes maintain O(log N) neighbours and route to a target in O(log N) hops. Pastry differs from Tapestry only in the method by which it handles network locality and replication (Rowstron and Druschel 2001a). Each Pastry node maintains a 'leaf set' and a 'routing table'. The leaf set contains *l*/2 node IDs on either side of the local node ID in the node ID space. The routing table, in row *r* column *c*, points to the node ID with the same *r*-digit prefix as the local node, but with an r+1 digit of *c*. A Pastry node periodically probes leaf set and routing table nodes, with periodicity of T_{ls} and T_{rt} and a timeout T_{out}. Mahajan, Castry et al. analysed the reliability versus maintenance cost tradeoffs in terms of the parameters *l*, $T_{ls,}$ $T_{rt,}$ and T_{out} (Mahajan, Castro et al. 2003). They concluded that earlier concerns about excessive maintenance cost in a churning P2P network were unfounded, but suggested followup work for a wider range of reliability targets, maintenance costs and probe periods. Rhea Geels et al. concluded that existing DHTs fail at high churn rates (Rhea, Geels et al. 2003). Building on a Pastry implementation from Rice University, they found that most lookups fail to complete when there is excessive churn. They conjectured that short-lived nodes often leave the network with lookups that have not yet timed out, but no evidence was provided to confirm the theory. They identified three design issues that affect DHT performance under churn: reactive versus periodic recovery of peers; lookup timeouts; and choice of nearby neighbours. Since reactive recovery was found to add traffic to already congested links, the authors used periodic recovery in their design. For lookup timeouts, they advocated an exponentially weighted moving average of each neighbour's response time, over alternative fixed timeout or 'virtual coordinate' schemes. For selection of nearby neighbours, they found that 'global sampling' was more effective than simply sampling a 'neighbour's neighbours' or 'inverse neighbours'. Castro, Costa et al. have refuted the suggestion that DHTs cannot cope with high churn rates (Castro, Costa et al. 2003). By implementing methods for continuous detection and repair, their MSPastry implementation achieved shorter routing paths and a maintenance overhead of less than half a message per second per node.

There have been more recent proposals based on these early Plaxton-like schemes. Maymounkov and Mazieres described their system, Kadlemia, as being most like Pastry in the way that it repeatedly find nodes about half way to the target node (Maymounkov and Mazieres 2002). Kadlemia uses a bit-wise exclusive or (XOR) metric[4] for the 'distance' between 160 bit node identifiers. Each node keeps a list of contact nodes for each section of the node space that is between 2^i and 2^{i+1} from itself ($0 \leq i < 160$). Longer-lived nodes are deliberately given preference on this list – it has been found in Gnutella that the longer a node has been active, the more likely it is to remain active. Like Kadlemia, Willow uses the XOR metric (van Renesse and Bozdog 2004). It implements a Tree Maintenance Protocol to 'zipper' together broken segments of a tree. Where other schemes use DHT routing to inefficiently add new peers, Willow can merge disjoint or broken trees in O(log N) parallel operations.

3.2. Rings

Chord is the prototypical DHT ring, so we first sketch its operation. Chord maps nodes and keys to an identifier ring (Stoica, Morris et al. 2001; Stoica, Morris et al. 2003). Chord supports one main operation: find a node with the given key. It uses Consistent Hashing (Section 3) to minimize disruption of keys when nodes join and leave the network. However, Chord peers need only track O(log N) other peers, not all peers as in the original consistent hashing proposal (Karger, Lehman et al. 1997). It enables concurrent node insertions and deletions, improving on PRR. Compared to Pastry, it has a simpler join protocol. Each Chord peer tracks its predecessor, a list of successors and a finger table. Using the finger table, each hop is at least half the remaining distance around the ring to the target node, giving an average[5] lookup hop count of $(½)\log_2 N$. Each Chord node runs a periodic stabilization routine that updates predecessor and successor pointers to cater for newly added nodes. All successors of a given node need to fail for the ring to fail. Although a node departure could be treated the same as a failure, a departing Chord node first notifies the predecessor and successors, so as to improve performance.

In their definitive paper, Chord's inventors critiqued its dependability under churn (Stoica, Morris et al. 2003). They provided some proofs on the behaviour of the Chord network when nodes in a stable network fail, stressing that such proofs are inadequate in the general case of a perpetually churning network. An earlier paper had posed the question, "For lookups to be successful during churn, how regularly do the Chord stabilization routines need to run?" (Liben-Nowell, Balakrishnan et al. 2002). Stoica, Morris et al. modeled a range of node join/departure rates and stabilization periods for a Chord network of 1000 nodes. They measured the number of timeouts (caused by a finger pointing to a departed node) and lookup failures (caused by nodes that temporarily point to the wrong successor during churn). They also modelled the 'lookup stretch', the ratio of the Chord lookup time to optimal lookup time on the underlying network. They demonstrated the latency advantage of recursive lookups over iterative lookups, but there remains room for delay reduction. For further work, the authors proposed to improve resilience to network partitions, using a small set of known nodes or 'remembered' random nodes. To reduce the number of messages per lookup, they suggested an increase in the size of each step around the ring, accomplished via a larger number of fingers at each node. Much of the paper assumed independent, equally likely node failures. Analysis of correlated node failures, caused by massive site or backbone failures, will be more important in some deployments. The paper did not attempt to recommend a fixed optimal stabilization rate. Liben-Nowell, Balakrishnan et al. had suggested that optimum stabilization rate might evolve according to measurements of peers' behaviour (Liben-Nowell, Balakrishnan et al. 2002) – such a mechanism has yet to be devised.

Alima, El-Ansary et al. considered the communication costs of Chord's stabilization routines, referred to as 'active correction', to be excessive (Alima, El-Ansary et al. 2003). Two other robustness issues also motivated their Distributed K-ary Search (DKS) design, which is similar to Chord. Firstly, the total system should evolve for an optimum balance between the number of peers, the lookup hopcount and the size of the routing table. Secondly, lookups should be reliable – P2P algorithms should be able to guarantee a successful lookup for key/value pairs that have been inserted into the system. A similar lookup correctness issue was raised elsewhere by one of Chord's authors, "Is it possible to augment the data structure[6] to work even when nodes (and their associated finger lists) just disappear?" (Karger and Ruhl 2002) Alima, El-Ansary et al. asserted that P2Ps using active correction, like Chord, Pastry and Tapesty, are unable to give such a guarantee. They propose an alternate 'correction-on-use' scheme, whereby expired routing entries are corrected by information piggybacking lookups and insertions. A prerequisite is that lookup and insertion rates are significantly higher than node arrival, departure and failure rates. Correct lookups are guaranteed in the presence of simultaneous node arrivals or up to f concurrent node departures, where f is configurable.

3.3. Tori

Ratnasamy, Francis et al. developed the Content-Addressable Network (CAN), another early DHT

[4] To be more precise, Maymounkov and Mazieres make comparison with Pastry's first routing phase, saying that Pastry's second phase uses numeric difference.

[5] For r successors, the average hop count is more accurately expressed as $(½)\log_2 N - (½)\log_2(r) + 1$

[6] The question was posed in the context of a nearest neighbour search algorithm, a proposed Chord extension.

widely referenced alongside Tapestry, Pastry and Chord (Ratnasamy, Francis et al. 2001; Ratnasamy 2002). It is arranged as a virtual *d*-dimensional Cartesian coordinate space on a d-torus. Each node is responsible for a zone in this coordinate space. The designers used a heuristic thought to be important for large, churning P2P networks: keep the number of neighbours independent of system size. Consequently, its design differs significantly from Pastry, Tapestry and Chord. Whereas they have O(logN) neighbours per node and O(logN) hops per lookup, CAN has O(d) neighbours and O($dn^{1/d}$) hop-count. When CAN's system-wide parameter d is set to log(N), CAN converges to their profile. If the number of nodes grows, a major rearrangement of the CAN network may be required (Ruhl 2003). The CAN designers considered building on PRR, but opted for the simple, low-state-per-node CAN algorithm instead. They had reasoned that a PRR-based design would not perform well under churn, given node departures and arrivals would affect a logarithmic number of nodes (Ratnasamy, Francis et al. 2001).

There have been preliminary assessments of CAN's resilience. When a node leaves the CAN in an orderly fashion, it passes its own Virtual ID (VID), its neighbours' VIDs and IP addresses, and its key/value pairs to a takeover node. If a node leaves abruptly, its neighbours send recovery messages towards the designated takeover node. CAN ensures the recovery messages reach the takeover node, even if nodes die simultaneously, by maintaining a VID chain with Chord's stabilization algorithm. Some initial 'proof of concept' resilience simulations were run using the Network Simulator (ns) (McCanne and Floyd) for up to a few hundred nodes. Average hopcounts and lookup failure probabilities were plotted against the total number of nodes, for various node failure rates (Ratnasamy, Francis et al. 2001). The CAN team documented several open research questions pertaining to state/hopcount tradeoffs, resilience, load, locality and heterogeneous peers (Ratnasamy 2002; Ratnasamy, Shenker et al. 2002).

3.4. Butterflies

Viceroy approximates a butterfly network (Malkhi, Naor et al. 2002). It generally has constant degree[7] like CAN. Like Chord, Tapesty and Pastry, it has logarithmic diameter. It improves on these systems, inasmuch as its diameter is better than CAN and its degree is better than Chord, Tapestry and Pastry. As with most DHTs, it utilizes Consistent Hashing. When a peer joins the Viceroy network, it takes a random but permanent 'identity' and selects its 'level' within the network. Each peer maintains general ring pointers ('predecessor' and 'successor'), level ring pointers ('nextonlevel' and 'prevonlevel') and butterfly pointers ('left', 'right' and 'up'). When a peer departs, it normally passes its key pairs to a successor, and notifies other peers to find a replacement peer.

The Viceroy paper scoped out the issue of robustness. It explicitly assumed that peers do not fail (Malkhi, Naor et al. 2002). It assumed that join and leave operations do not overlap, so as to avoid the complication of concurrency mechanisms like locking. Kaashoek and Karger were somewhat critical of Viceroy's complexity (Kaashoek and Karger 2003). They also pointed to its fault tolerance blindspot. Li and Plaxton suggested that such constant-degree algorithms deserve further consideration (Li and Plaxton 2002). They offered several pros and cons. The limited degree may increase the risk of a network partition, or inhibit use of local neighbours (for the simple reason that there are less of them). On the other hand, it may be easier to reason about the correctness of fixed-degree networks. One of the Viceroy authors has since proposed constant-degree peers in a two-tier, locality-aware DHT (Abraham, Malkhi et al. 2004) – the lower degree maintained by each lower-tier peer purportedly improves network adaptability. Another Viceroy author has since explored an alternative bounded-degree graph for P2P, namely the de Bruijn graph (Naor and Wieder 2003a).

3.5. de Bruijn Graphs

De Bruijn graphs have had numerous refinements since their inception (de Bruijn 1946; Mao 2003). Schlumberger was the first to use them for networking (Schlumberger 1974). Two research teams independently devised the 'generalized' de Bruijn graph that accommodates a flexible number of nodes in the system (Reddy, Pradhan et al. 1980; Imase and Itoh 1981). Rowley and Bose studied fault-tolerant rings overlaid on the de Bruijn graph (Rowley and Bose 1993). Lee, Liu et al. devised a two-level de Bruijn hierarchy, whereby clusters of local nodes are interconnected by a second-tier ring (Lee, Liu et al. 2000).

Many of the algorithms discussed previously are 'greedy' in that each time a query is forwarded, it moves closer to the destination. Unfortunately, greedy algorithms are generally suboptimal – for a given degree, the routing distance is longer than necessary (Naor and Wieder 2004). Unlike these earlier P2P designs, de Bruijn graphs of degree k achieve an asymptotically optimal diameter $\log_k n$, where n is the number of nodes in the system and k can be varied to improve resilience. If there are O(log(n)) neighbours per node, the de Bruijn hop count is O(log n/log log n). To illustrate de Bruijn's practical advantage, consider a network with one million nodes of degree 20: Chord has a diameter of 20, while de Bruijn has a diameter of 5 (Loguinov, Kumar et al. 2003). In 2003, there were a quick succession of de Bruijn proposals – D2B (Fraigniaud and Gauron 2003), Koorde (Kaashoek and Karger 2003), Distance Halving (Naor and Wieder 2003a; Naor and Wieder 2003b) and the Optimal

[7] Viceroy's expected degree is a constant. However, its high probability bound is O(log n). For a very small number of nodes, degree is Ω(log n). Kaashoek, F. and D. Karger (2003). "Koorde: A Simple Degree-optimal Hash Table." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Diameter Routing Infrastructre (ODRI) (Loguinov, Kumar et al. 2003).

Fraigniaud and Gauron began the D2B design by laying out an informal problem statement: keys should be evenly distributed; lookup latency should be small; traffic load should be evenly distributed; updates of routing tables and redistribution of key should be fast when nodes join or leave the network. They defined a node's "congestion" to be the probability that a lookup will traverse it. Apart from its optimal de Bruijn diameter, they highlighted D2B's merits: a constant expected update time when nodes (O(log n) w.h.p[8].); the expected node congestion is O((logn)/n) (O(($\log^2$n)/n) w.h.p.) (Fraigniaud and Gauron 2003). D2B's resilience was discussed only in passing.

Koorde extends Chord to attain the optimal de Bruijn degree/diameter tradeoff above (Kaashoek and Karger 2003). Unlike D2B, Koorde does not constrain the selection of node identifiers. Also unlike D2B, it caters for concurrent joins, by extension of Chord's functionality. Kaashoek and Karger investigated Koorde's resilience to a rather harsh failure scenario: "in order for a network to stay connected when all nodes fail with probability of ½, some nodes must have degree Ω(log n)" (Kaashoek and Karger 2003). They sketched a mechanism to increase Koorde's degree for this more stringent fault tolerance, losing de Bruijn's constant degree advantage. Similarly, to achieve a constant-factor load balance, Koorde would have to sacrifice its degree optimality. They suggested that the ability to trade the degree, and hence the maintenance overhead, against the expected hop count may be important for churning systems. They also identified an open problem: find a load-balanced, degree optimal DHT. Datta, Girdzijauskas et al. argued recently that Koorde makes an unrealistic, simplifying assumption of uniform key distribution (Datta, Girdzijauskas et al. 2004). They showed that for arbitrary key distributions, de Bruijn graphs fail to meet the dual goals of load balancing and search efficiency. They posed the question, "(Is there) a constant routing table sized DHT which meets the conflicting goals of storage load balancing and search efficiency for an arbitrary and changing key distribution?"

Distance Halving was also inspired by de Bruijn (Naor and Wieder 2003a) and shares its optimal diameter. Naor and Wieder argued for a two-step "continuous-discrete" approach for its design. The correctness of its algorithms is proven in a continuous setting. The algorithms are then mapped to a discrete space. The source x and target y are points on the continuous interval [0,1). Data items are hashed to this same interval. σ is a string which determines how messages leave any point on the ring: if bit *t* of the string is 0, the left leg is taken; if it is 1, the right leg is taken. σ increases by one bit each hop, giving a sequence by which to step around the ring. A lookup has two phases. In the first, the lookup message containing the source, target and the random string hops toward the midpoint of the source and target. On each hop, the distance between σ_t(x) and σ_t(y) is halved, by virtue of the specific 'left' and 'right' functions. In the second phase, the message steps 'backward' from the midpoint to the target, removing the last bit in σ_t at each hop. 'Join' and 'leave' algorithms were outlined but there was no consideration of recovery times or message load on churn. Using the Distance Halving properties, the authors devised a caching scheme to relieve congestion in a large P2P network. They have also modified the algorithm to be more robust in the presence of random faults (Naor and Wieder 2003b).

Solid comparisons of DHT resilience are scarce, but Loguinov, Kumar et al. give just that in their ODRI paper (Loguinov, Kumar et al. 2003). They compare Chord, CAN and de Bruijn in terms of routing performance, graph expansion and clustering. At the outset, they give the optimal *diameter* (the *maximum* hopcount between any two nodes in the graph) and *average hopcount* for graphs of fixed degree. De Bruijn graphs converge to both optima, and outperform Chord and CAN on both counts. These optima impact both delay and aggregate lookup load. They present two clustering measures (edge expansion and node expansion) which are interesting for resilience. Unfortunately, after decades of de Bruijn research, they have no exact solution. De Bruijn was shown to be superior against a simplified node expansion metric – for any destination, there are a large number of non-overlapping paths. If there is a single node failure, very few of the alternative paths to a destination are affected.

3.6. Skip Graphs

Skip Graphs have been pursued by two research camps (Aspnes and Shah 2003; Harvey, Jones et al. 2003a). They augment the earlier Skip Lists (Pugh 1989; Pugh 1990). Unlike earlier balanced trees, the Skip List is probabilistic – its insert and delete operations do not require tree rearrangements and so are faster by a constant factor. The Skip List consists of layers of ordered linked lists. All nodes participate in the bottom layer 0 list. Some of these nodes participate in the layer 1 list with some fixed probability. A subset of layer 1 nodes participate in the layer 2 list, and so on. A lookup can proceed quickly through the list by traversing the sparse upper layers until it is close to, or at, the target. Unfortunately, nodes in the upper layers of a Skip List are potential hot spots and single points of failure. Unlike Skip Lists, Skip Graphs provide multiple lists at each level for redundancy, and every node participates in one of the lists at each level.

[8] W.h.p. With high probability $1 - n^{-\varepsilon}$

Each node in a Skip Graph has $\Theta(\log n)$ neighbours on average, like some of the preceding DHTs. The Skip Graph's primary edge over the DHTs is its support for prefix and proximity search. DHTs hash objects to a random point in the graph. Consequently, they give no guarantees over where the data is stored. Nor do they guarantee that the path to the data will stay within the one administration as far as possible (Harvey, Jones et al. 2003a). Skip graphs, on the other hand, provide for location-sensitive name searches. For example, to find the document *docname* on the node *user.company.com*, the Skip Graph might step through its ordered lists for the prefix *com.company.user* (Harvey, Jones et al. 2003a). Alternatively, to find an object with a numeric identifier, an algorithm might search the lowest layer of the Skip Graph for the first digit, the next layer for the next digit, in the same vein until all digits are resolved. Being ordered, Skip Graphs also facilitate range searches. In each of these examples, the Skip Graph can be arranged such that the path to the target, as far as possible, stays within an administrative boundary. If one administration is detached from the rest of the Skip Graph, routing can continue within each of the partitions. Mechanisms have been devised to merge disconnected segments (Harvey, Jones et al. 2003b), though at this stage, segments are remerged one at a time. A parallel merge algorithm has been flagged for future work.

The advantages of Skip Graphs come at a cost. To be able to provide range queries and data placement flexibility, Skip Graph nodes require many more pointers than their DHT counterparts. An increased number of pointers implies increased maintenance traffic. Another shortcoming of at least one of the early proposals was that no algorithm was given to assign keys to machines. Consequently, there are no guarantees on system-wide load balancing or on the distance between adjacent keys (Aspnes, Kirsch et al. 2004). Aspnes, Kirsch et al. have recently devised a scheme to reduce the inter-machine pointer count from $O(m \log m)$, where m is the number of data elements, to $O(n \log n)$, where n is the number of nodes (Aspnes, Kirsch et al. 2004). They proposed a two-layer scheme – one layer for the Skip Graph itself and the second 'bucket layer'. Each machine is responsible for a number of buckets and each bucket elects a representative key. Nodes locally adjust their load. They accept additional keys if they are below their threshold or disperse keys to nearby nodes if they are above threshold. There appear to be numerous open issues: simulations have been done but analysis is outstanding; mechanisms are required to handle the arrival and departure of nodes; there were only brief hints as to how to handle nodes with different capacities.

4. Semantic Index

Semantic indexes capture object relationships. While the semantic-free methods (DHTs) have firmer theoretic foundations and guarantee that a key can be found if it exists, they do not on their own capture the relationships between the document name and its content or metadata. Semantic P2P designs do. However, since their design is often driven by heuristics, they may not guarantee that scarce items will be found.

So what might the semantically indexed P2Ps add to an already crowded field of distributed information architectures? At one extreme there are the distributed relational database management systems (RDBMSs), with their strong consistency guarantees (Gribble, Brewer et al. 2000). They provide strong data independence, the flexibility of SQL queries and strong transactional semantics – Atomicity, Consistency, Isolation and Durability (ACID) (Gray 1981). They guarantee that the query response is complete – all matching results are returned. The price is performance. They scale to perhaps 1000 nodes, as evidenced in Mariposa (Stonebraker, Aoki et al. 1996; Loo, Hellerstein et al. 2004), or require query caching front ends to constrain the load (Gribble, Brewer et al. 2000). Database research has "arguably been cornered into into traditional, high-end, transactional applications" (Harren, Hellerstein et al. 2002). Then there are distributed file systems, like the Network File System (NFS) or the Serverless Network File Systems (xFS), with little data independence, low-level file retrieval interfaces and varied consistency (Gribble, Brewer et al. 2000). Today's eclectic mix of Content Distribution Networks (CDNs) generally deload primary servers by redirecting web requests to a nearby replica. Some intercept the HTTP requests at the DNS level and then using consistent hashing to find a replica (Hellerstein 2003). Since this same consistent hashing was a forerunner to the DHT approaches above, CDNs are generally constrained to the same simple key lookups.

The opportunity for semantically indexed P2Ps, then, is to provide:

a) *moderate* levels of data independence, consistency and query flexibility, and

b) *probabilistically* complete query responses, across

c) *very large* numbers of low-cost, geographically distributed, dynamic nodes.

4.1. Keyword Lookup

Perhaps the most widely referenced P2P system for simple keyword match is Gnutella (Klingberg and Manfredi 2002). Gnutella queries contain a string of keywords. Gnutella peers answer when they have files[9] whose names contain all the keywords. As discussed in Section 2.1, early versions of Gnutella did not forward the document index. Queries were flooded and peers searched their own local indexes for filename matches. An early review highlighted numerous areas for improvement (Ripeanu, Iamnitchi et al. 2002). It was estimated that the query traffic alone from 50,000 early-

[9] The Gnutella 0.6 specification only provides semantics for finding plain files, but hints that Gnutella could store other resources, like cryptographic keys or meta-information.

generation Gnutella nodes would amount to 1.7% of the total U.S. internet backbone traffic at December 2000 levels. It was speculated that high degree Gnutella nodes would impede dependability. An unnecessarily high percentage of Gnutella traffic crossed Autonomous System (AS) boundaries – a locality mechanism may have found suitable nearby peers.

Fortunately, there have since been numerous enhancements within the Gnutella Developer Forum. At the time of writing, it has been reported that Gnutella has almost 350,000 unique hosts, of which nearly 90,000 accept incoming connections ("Limewire Host Count" 2004). One of the main improvements is that an index of filename keywords, called the Query Routing Table (QRT), can now be forwarded from 'leaf peers' to its 'ultrapeers' (Singla and Rohrs 2002). Ultrapeers can then ensure that the leaves only receive queries for which they have a match, dramatically reducing the query traffic at the leaves. Ultrapeers can have connections to many leaf nodes (~10-100) and a small number of other ultrapeers (<10) (Singla and Rohrs 2002). Originally, a leaf node's QRT was not forwarded by the parent ultrapeer to other ultrapeers. More recently, there has been a proposal to distributed aggregated QRTs amongst ultrapeers (Fisk 2003b). To further limit traffic, QRTs are compressed by hashing, according to the Query Routing Protocol (QRP) specification (Rohrs 2002). This same specification claims QRP may reduce Gnutella traffic by orders of magnitude, but cautions that simulation is required before mass deployment. A known shortcoming of QRP was that the extent of query propagation was independent of the popularity of the search terms. The Dynamic Query Protocol addressed this (Fisk 2003a). It required leaf nodes to send single queries to high-degree ultrapeers which adjust the queries' time-to-live (TTL) bounds according to the number of received query results. An earlier proposal, called the Gnutella UDP Extension for Scalable Searches (GUESS) (Daswani and Fisk 2002), similarly aimed to reduce the number of queries for widely distributed files. GUESS reuses the *non-forwarding* idea (Section 2). A GUESS peer repeatedly queries single ultrapeers with a TTL of 1, with a small timeout on each query to limit load. It chooses the number of iterations and selects ultrapeers so as to satisfy its search needs. At the time of writing, GUESS has pending status. For adaptability, a small number of experimental Gnutella nodes have implemented XML schemas for richer queries (Thadani 2001b; Thadani 2001a). None of the above Gnutella proposals explicitly assess robustness.

The broader research community has recently been leveraging aspects of the Gnutella design. Lv, Ratnasamy et al. exposed one assumption implicit in some of the early DHT work – that designs "such as Gnutella are inherently not scalable, and therefore should be abandoned" (Lv, Ratnasamy et al. 2002). They argued that by making better use of the more powerful peers, Gnutella's scalability issues could be alleviated. Instead of its flooding mechanism, they used random walks. Their preliminary design to bias random walks towards high capacity nodes did not go as far as the ultrapeer proposals in that the indexes did not move to the high capacity nodes. Chawathe, Ratnasamy et al. chose to extend the Gnutella design with their Gia system, in response to the perceived shortcomings of DHTs in Section 1.2 (Chawathe, Ratnasamy et al. 2003). Compared to the early Gnutella designs, they incorporated several novel features. They devise a topology adaptation algorithm so that most peers are attached to high-degree peers. They use a random walk search algorithm, in lieu of flooding, and bias the query load towards higher-degree peers. For 'one-hop replication', they require all nodes keep pointers to content on adjacent peers. To implement a receiver-controlled token-based flow control, a peer must have a token from its neighbouring peer before it sends a query to it. The combination of these features, they claim, provides a scalability improvement of three to five orders of magnitude over Gnutella "while retaining significant robustness". Castro, Costa and Rowstron argued that if Gnutella were built on top of structured overlay, then both the query and overlay maintenance traffic could be reduced (Castro, Costa et al. 2004). The main robustness metrics they used were the 'collapse point' query rate (the per node query rate at which the successful query rate falls below 90%) and the average hop-count immediately prior to collapse. Their comparison with Gnutella did not take into account the Gnutella enhancements above – this was left as future work. So too was any discussion of frequent node arrivals and departures. Yang, Vinograd et al. explore various policies for peer selection in the GUESS protocol, since the issue is left open in the original proposal (Yang, Vinograd et al. 2004). For example, the peer initiating the query could choose peers that have been "most recently used" or that have the "most files shared". Various policy pitfalls are identified. For example, good peers could be overloaded, victims of their own success. Alternatively, malicious peers could encourage the querying peer to try inactive peers. They conclude that a "most results" policy gives the best balance of robustness and efficiency. Like Castro, Costa and Rowstron, they concentrated on the static network scenario. Cholvi, Felber et al. very briefly describe how similar "least recently used" and "most often used" heuristics can be used by a peer to select peer 'acquaintances' (Cholvi, Felber et al. 2004). They were motivated by the congestion associated with Gnutella's TTL-limited flooding. Recognizing that the busiest peers can quickly become overloaded central hubs for the entire network, they limit the number of acquaintances for any given peer to 25. They sketch a mechanism to decrement a query's TTL multiple times when it traverses "interested peers". In summary, these Gnutella-related investigations are characterized by a bias for high degree peers and very short directed query paths, a disdain for flooding, and concern about excessive load on the 'better' peers. Generally, the robustness analysis for dynamic networks (content updates and node arrivals/departures) remains open.

One aspect of P2P keyword search systems has received particular attention: should the index be partitioned by document or by keyword? The issue affects scalability. To be partitioned by document, each node has a local index of documents for which it is responsible. Gnutella is a prime example. Queries are generally flooded in systems partitioned by document. On the other hand, a peer may assume responsibility for a set of keywords. The peer uses an inverted list to find a matching document, either locally or at another peer. If the query contains several keywords, inverted lists may need to be retrieved from several different peers to find the intersection (Shi, Guangwen et al. 2004). The initial assessment by Li, Loo et al. was that the partition-by-document approach was superior (Li, Loo et al. 2003). For one scenario of a full-text web search, they estimated the communications costs to be about six times higher than the feasible budget. However, wanting to exploit prior work on inverted list intersection, they studied the partition-by-keyword strategy. They proposed several optimizations which put the communication costs for a partition-by-keyword system within an order of magnitude of feasibility. There had been a couple of prior papers that suggested partitioned-by-keyword designs incorporate DHTs to map keywords to peers (Gnawali 2002; Reynolds and Vahdat 2003). In Gnawali's Keyword-set Search System (KSS), the index is partitioned by *sets* of keywords. Terpstra, Behnel et al. point out that by keeping keyword pairs or triples, the number of lists per document in KSS is squared or tripled (Terpstra, Behnel et al. 2004). Shi, Guangwen et al. interpreted the approximations of Li, Loo et al. to mean that neither approach is feasible on its own (Shi, Guangwen et al. 2004). Their Multi-Level Partitioning (MLP) scheme incorporates both partitioning approaches. They arrange nodes into a group hierarchy, with all nodes in the single 'level 0' group, and with the same nodes sub-divided into k logical subgroups on 'level 1'. The subgroups are again divided, level by level, until level l. The inverted index is partitioned by document between groups and by keyword within groups. MLP avoids the query flooding normally associated with systems partitioned by document, since a small number of nodes in each group process the query. It reduces the bandwidth overheads associated with inverted list intersection in systems partitioned solely by keyword, since groups can calculate the intersection independently over the documents for which they are responsible. MLP was overlaid on SkipNet, per Section 3.6 (Harvey, Jones et al. 2003a). Some initial analyses of communications costs and query latencies were provided.

Much of the research above addresses *partial* keyword search. Daswani et al. highlighted the open problem of efficient, *comprehensive* keyword search (Daswani, Garcia-Molina et al. 2003). How can exhaustive searches be achieved without flooding queries to every peer in the network? Terpstra, Behnel et al. couched the keyword search problem in rendezvous terms: dynamic keyword queries need to 'meet' with static document lists (Terpstra, Behnel et al. 2004). Their Bitzipper scheme is partitioned by document. They improved on full flooding by putting document metadata on $2\sqrt{n}$ nodes and forwarding queries through only $6\sqrt{n}$ nodes. They reported that Bitzipper nodes need only 1/166th of the bandwidth of full-flooding Gnutella nodes for an exhaustive search. An initial comparison of query load was given. There was little consideration of either static or dynamic resilience, that is, of nodes failing, of documents continually changing, or of nodes continually joining and leaving the network.

4.2. Peer Information Retrieval

The field of Information Retrieval (IR) has matured considerably since its inception in the 1950s (Singhal 2001). A taxonomy for IR models has been formalized (Baeza-Yates and Ribeiro-Neto 1999). It consists of four elements: a representation of documents in a collection; a representation of user queries; a framework describing relationships between document representations and queries; and a ranking function that quantifies an ordering amongst documents for a particular query. Three main issues motivate current IR research – information relevance, query response time, and user interaction with IR systems. The dominant IR trends for searching large text collections are also threefold (Baeza-Yates and Ribeiro-Neto 1999). The size of collections is increasing dramatically. More complicated search mechanisms are being found to exploit document structure, to accommodate heterogeneous document collections, and to deal with document errors. Compression is in favour – it may be quicker to search compact text or retrieve it from external devices. In a *distributed* IR system, queries processing has four parts. Firstly, particular collections are targeted for the search. Secondly, queries are sent to the targeted collections. Queries are then evaluated at the individual collections. Finally results from the collections are collated.

So how do P2P networks differ from distributed IR systems? Bawa, Manku et al. presented four differences (Bawa, Manku et al. 2003). They suggested that a P2P network is typically larger, with tens or hundreds of thousands of nodes. It is usually more dynamic, with node lifetimes measured in hours. They suggested that a P2P network is usually homogeneous, with a common resource description language. It lacks the centralized "mediators" found in many IR systems, that assume responsibility for selecting collections, for rewriting queries, and for merging ranked results. These distinctions are generally aligned with the peer characteristics in Section 1. One might add that P2P nodes display more symmetry – peers are often both information consumers and producers. Daswani, Garcia-Molina et al. pointed out that, while there are IR techniques for ranked keyword search at moderate scale, research is required so that ranking mechanisms are efficient at the larger scale targeted by P2P designs (Daswani, Garcia-Molina et al. 2003). Joseph and Hoshiai surveyed several P2P systems using metadata techniques from the IR toolkit (Joseph and Hoshiai 2003). They described an assortment of IR techniques and P2P systems, including various metadata formats, retrieval models, bloom filters, DHTs and trust issues.

In the ensuing paragraphs, we survey P2P work that has incorporated information retrieval models, particularly the Vector Model and the Latent Semantic Indexing Model. We omit the P2P work based on Bayesian models. Some have pointed to such work (Joseph and Hoshiai 2003), but it made no explicit mention of the model (Cohen, Fiat et al. 2003). One early paper on P2P content-based image retrieval also leveraged the Bayesian model (Muller and Henrich 2003). For the former two models, we briefly describe the design, then try to highlight robustness aspects. On robustness, we are again stymied for lack of prior work. Indeed, a search across all proceedings of the Annual ACM Conference on Research and Development in Information Retrieval for the words "reliable", "available", "dependable" or "adaptable" did not return any results at the time of writing. In contrast, a standard text on distributed database management systems (Ozsu and Valduriez 1999) contains a whole chapter on reliability. IR research concentrates on performance measures. Common performance measures include recall, the fraction of the relevant documents which has been retrieved, and precision, the fraction of the retrieved documents which is relevant (Baeza-Yates and Ribeiro-Neto 1999). Ideally, an IR system would have high recall and high precision. Unfortunately techniques favouring one often disadvantage the other (Singhal 2001).

4.2.1. Vector Model

The vector model (Salton, Wong et al. 1975) represents both documents and queries as term vectors, where a term could be a word or a phrase. If a document or query has a term, the weight of the corresponding dimension of the vector is non-zero. The similarity of the document and query vectors gives an indication of how well a document matches a particular query.

The weighting calculation is critical across the retrieval models. Amongst the numerous proposals for the probabilistic and vector models, there are some commonly recurring weighting factors (Singhal 2001). One is term frequency. The more a term is repeated in a document, the more important the term is. Another is inverse document frequency. Terms common to many documents give less information about the content of a document. Then there is document length. Larger documents can bias term frequencies, so weightings are sometimes normalized against document length. The expression "TFIDF weighting" refers to the collection of weighting calculations that incorporate term frequency and inverse document frequency, not just to one. Two weighting calculations have been particularly dominant – Okapi (Robertson, Walker et al. 1999) and pivoted normalization (Singhal, Choi et al. 1999). A distributed version of Google's Pagerank algorithm has also been devised for a P2P environment (Sankaralingam, Sethumadhavan et al. 2003). It allows incremental, ongoing Pagerank calculations while documents are inserted and deleted.

A couple of early P2P systems leveraged the vector model. Building on the vector model, PlanetP divided the ranking problem into two steps (Cuenca-Acuna, Peery et al. 2002). In the first, peers are ranked for the probability that they have matching documents. In the second, higher priority peers are contacted and the matching documents are ranked. An Inverse Peer Frequency, analogous to the Inverse Document Frequency, is used to rank relevant peers. To further constrain the query traffic, PlanetP contacts only the first group of m peers to retrieve a relevant set of documents. In this way, it repeatedly contacts groups of m peers until the top k document rankings are stable. While the PlanetP designers first quantified recall and precision, they also considered reliability. Each PlanetP peer has a global index with a list of all other peers, their IP addresses, and their Bloom filters. This large volume of shared information needs to be maintained. Klampanos and Jose saw this as PlanetP's primary shortcoming (Klampanos and Jose 2004). Each Bloom filter summarized the set of terms in the local index of each peer. The time to propagate changes, be they new documents or peer arrivals/departures, was studied by simulation for up to 1000 peers. The reported propagation times were in the hundreds of seconds. Design workarounds were required for PlanetP to be viable across slower dial-up modem connections. For future work, the authors were considering some sort of hierarchy to scale to larger numbers of peers.

A second early system using the vector model is the Fault-tolerant, Adaptive, Scalable Distributed search engine (Kronfol 2002), which extended the Freenet design (Section 2.3) for richer queries. The original Freenet design could find a document based on a globally unique identifier. Kronfol's design added the ability to search, for example, for documents about "apples AND oranges NOT bananas". It uses a TFIDF weighting scheme to build a document's term vector. Each peer calculates that similarity of the query vector and local documents and forwards the query to the best downstream peer. Once the best downstream peer returns a result, the second-best peer is tried, and so on. Simulations with 1000 nodes gave an indication of the query path lengths in various situations - when routing queries in a network with constant rates of node and document insertion, when bootstrapping the network in a "worst-case" ring topology, or when failing randomly and specifically selected peers. Kronfol claimed excellent average-case performance – less than 20 hops to retrieve the same top *n* results as a centralized search engine. There were, however, numerous cases where the worst-case path length was several hundred hops in a network of only 1000 nodes.

In parallel, there have been some P2P designs based on the vector model from the University of Rochester – pSearch[10] (Tang, Xu et al. 2002b; Tang, Xu et al. 2003) and eSearch (Tang and Dwarkadas 2004). The early pSearch paper suggested a couple of retrieval models, one of which was the Vector Space Model, to search only the nodes likely to have matching documents. To obtain approximate global statistics for the TFIDF calculation, a spanning tree was constructed across a subset of the peers. For the *m* top terms, the term-to-document index was inserted into a Content-Addressable Network (Ratnasamy 2002). A variant which mapped terms to document *clusters* was also suggested. eSearch is a hybrid of the partition-by-document and partition-by-term approaches seen in the previous section. eSearch nodes are primarily partitioned by term. Each is responsible for the inverted lists for some top terms. For each document in the inverted list, the node stores the complete term list. To reduce the size of the index, the complete term lists for a document are only kept on nodes that are responsible for top terms in the document. eSearch uses the Okapi term weighting to select top terms. It relies on the Chord DHT (Stoica, Morris et al. 2003) to associate terms with nodes storing the inverted lists. It also uses automatic query expansion. This takes the significant terms from the top document matches and automatically adds them to the user's query to find additional relevant documents. The eSearch performance was quantified in terms of search precision, the number of retrieved documents, and various load-balancing metrics. Compared to the more common proposals for partitioning by keywords, eSearch consumed 6.8 times the storage space to achieve faster search times.

4.2.2. Latent Semantic Indexing

Another retrieval model used in P2P proposals is Latent Semantic Indexing (LSI) (Furnas, Deerwester et al. 1988). Its key idea is to map both the document and query vectors to a concept space with lower dimensions. The starting point is a $t*N$ weighting matrix, where t is the total number of indexed terms, N is the total number of documents, and the matrix elements could be TFIDF rankings. Using singular value decomposition, this matrix is reduced to a smaller number of dimensions, while retaining the more significant term-to-document mappings. Baeza-Yates and Ribeiro-Neto suggested that LSI's value is a novel theoretic framework, but that its practical performance advantage for real document collections had yet to be proven (Baeza-Yates and Ribeiro-Neto 1999). pSearch incorporated LSI (Tang, Xu et al. 2002b). By placing the indices for semantically similar documents close in the network, Tang, Xu et al. touted significant bandwidth savings relative to the early full-flooding variant of Gnutella (Tang, Xu et al. 2003). They plotted the number of nodes visited by a query. The also explored the tradeoff with accuracy, the percentage match between the documents returned by the the distributed pSearch algorithm and those from a centralized LSI baseline. In a more recent update to the pSearch work, Tang, Dwarkadas et al. summarized LSI's shortcomings (Tang, Dwarkadas et al. 2004). Firstly, for large document collections, its retrieval quality is inherently inferior to Okapi. Secondly, singular value decomposition consumes excessive memory and computation time. Consequently, the authors used Okapi for searching while retaining LSI for indexing. With Okapi, they selected the next node to be searched and selected documents on searched nodes. With LSI, they ensured that similar documents are clustered near each other, thereby optimizing the network search costs. When retrieving a small number of top documents, the precision of LSI+Okapi approached that of Okapi. However, if retrieving a large number of documents, the LSI+Okapi precision is inferior. The authors want to improve this in future work.

4.3. Peer Data Management

Thus far, we have reviewed P2P systems indexed by keys (Section 3) and by keywords (Sections 4.1 and 4.2). Motivated by a rich database heritage, there is another growing class called Peer Data Management Systems (PDMSs):*"In a PDMS, every peer is associated with a schema that represents the peer's domain of interest, and semantic relationships between peers are provided locally between pairs (or small sets) of peers. By traversing semantic paths of mappings, a query over one peer can obtain relevant data from any reachable peer in the network. Semantic paths are traversed by reformulating queries at a peer into queries on its neighbors"* (Tatarinov and Halevy 2004).

While there are some similarities between PDMSs and distributed databases, there are significant differences. Distributed databases have not dealt with the scale and dynamism of P2P networks where, because of their sheer numbers, nodes are expected to leave (fail) and return (recover) frequently (Crespo and Garcia-Molina 2002). In these conditions, the traditional data placement problem – locating data for the lowest query workload within resource constraints – is frustrated (Gribble, Halevy et al. 2001). Schema translation issues are also well-known, but the number of schemas and the lack of a common mediating schema is new (Tatarinov, Mork et al. 2003). Distributed databases are expected to give complete answers to queries, but P2P offers only probabilistically complete responses – peers might only have partial information or might even be disconnected. In distributed databases, the content location is generally known. In P2P, queries route via other peers to find the location (Ng, Ooi et al. 2003).

[10] The pSearch design had earlier been proposed under the name PeerSearch Tang, C., Z. Xu and M. Mahalingam (2002a). "PeerSearch: Efficient information retrieval in peer-to-peer networks." Proceedings of HotNets-I, ACM SIGCOMM.

System	Information Integration	Network Topology	Markup / Schema Language	Query Language
Edutella, Semantic Overlay Clusters, Hypercup (Nejdl, Wolf et al. 2002; Schlosser, Sintek et al. 2002; Loser, Naumann et al. 2003; Nejdl, Siberski et al. 2003)	Distributed query processing and mediation	Super-peer hypercubes, Clustering based on ontologies, queries or rules	Resource Description Framework (RDF) and Schema (RDFS) (Brickley and Guha 2000)	Edutella's RDF Query Exchange Language (QEL) (Nejdl, Decker et al. 2003)
Local Relational Model (Bernstein, Giunchiglia et al. 2002)	Domain relations and coordination formula		eXtensible Markup Language (XML)	
Piazza (Halevy, Ives et al. 2003a; Tatarinov, Mork et al. 2003)	Peer descriptions and storage descriptions		XML or RDF; XML Schema or OWL ontologies (McGuinness and van Harmelen 2004)3}	Based on XQuery (Boag, Chamberlin et al. 2003)
Chatty Web (Aberer, Cudre-Maroux et al. 2004)	Local agreement by semantic gossiping	Gnutella-like ping/pong discovery	XML	Generic selection, projection and mapping operators, XQuery
Hyperion (Arenas, Kantere et al. 2003)	Instantiate the Local Relational Model: Data coordination; mapping tables; distributed active rules			
PeerDB (Ng, Ooi et al. 2003; Ooi, Shu et al. 2003)	Phase 1: relation matching agents Phase 2: data retrieval agents	Short hops to peers that most recently provided answers		SQL
Xpeer (Sartiani, Manghi et al. 2004)	(Key goal is zero administration - only applicable to situations where schema mapping is not required)	Self-organizing network of peers and super-peers	XML	XQuery FLWOR (for, let, where, order by, return) expressions (Boag, Chamberlin et al. 2003)
DBGlobe (Milo, Abiteboul et al. 2003; Pitoura, Abiteboul et al. 2003)		Three-layer hierarchy of Primary Mobile Objects, Cell Administration Servers, and Community Administration Servers; Bloom filters to summarize local/nearby services and documents	XML Schema extensions	XQuery extensions
RDFPeers, Mult-Attribute Addressable Network (MAAN) (Cai, Frank et al. 2003; Cai and Frank 2004)		Stores <subject, predicate,object> triples on a Chord extension; Shuns super-peers; Uses 'successor probing' for load balancing	RDFS	Native queries (atomic triple patterns, disjunctive and range queries, and conjunctive multi-predicate queries) or RDQL (Miller, Seaborne et al. 2002)
NeuroGrid (Joseph 2002; Joseph and Hoshiai 2003)		Each node has a 'knowledge base' (keyword to node associations) and a Globally Unique Identifier table to identify queries; Gnutella-like mechanisms to prevent loops and indefinite propagation; Learns from prior user interactions	RDF	
Mutant Query Plans (Papadimos, Maier et al. 2003)		Peers take one or more roles – base server, index server, meta-index server, category server; Queries routed via distributed hierarchic catalogs	XML	Mutant query plans carry query operators and data; they accumulate partial results before returning to the initiating client with the complete result

Figure 2 Peer Data Management Systems

Nejdl, Siberski et al. suggested four building blocks for PDMSs: a schema language to describe the stored data; a query language to retrieve data; a network topology over which queries are routed; and techniques to integrate information from disparate peers (Nejdl, Siberski et al. 2003). In Figure 2, several PDMSs are classified along these lines. A blank table entry implies that a particular building block was not a major emphasis. Open standards for schemata and query languages can make for greater choice among P2P implementations. Rich, structured queries can support a wide range of applications. Information integration is about flexible interworking between different schemata and organizations.

In the following paragraphs, we briefly review some of the proposals that addressed *information integration*, the problem that arises when peers use heterogeneous schemata. We step through the first six rows of Figure 2. The Edutella designers saw two facets to information integration (Nejdl, Siberski et al. 2003). One is distributed query processing, where the type of query and the extent of clustering can determine the optimum location to combine query results. The other is mediation. They explored the use of explicit mediation peers and rule-based mediation in super-peers. Nejdl, Wolf et al. sketched the mapping from Edutella's RDF-based Query Exchange Language (QEL) to numerous other query languages – RQL (an RDF query language), TRIPLE (an RDF query and transformation language based on Horn logic), SQL, dbXML (an open source native XML database), and AmosQL (an object-relational query language) (Nejdl, Wolf et al. 2002). Edutella has a two-pronged approach to mediation. When queries can be answered completely by one peer, it uses a simple mediation model. Each peer implements a simple wrapping function locally, to and from Edutella's common data model. When query results are to be retrieved from several peers, there are the more complicated 'integrating' mediators or hubs. They translate a query to several sub-queries, which are forwarded to heterogeneous peers. They then collate the responses and send the query result to the originator.

Bernstein, Giunchiglia et al. proposed the Local Relational Model to address similar issues (Bernstein, Giunchiglia et al. 2002). They couched the model in terms of peers, acquaintances (a set of associated peers), coordination formulas (the semantic dependencies between a peer and its acquaintances), and domain relations (data translation rules between a peer and its acquaintances). Their overarching objective was to establish and evolve acquaintances between dynamic peers without a global registry or schema, and without significant effort by a database administrator. They made a start on the formalization of the Local Relational Model and instantiate it in terms of four modules – the user interface, the query manager, the update manager and the wrapper. Several items for future research were identified: a protocol to establish acquaintances and exchange peer names, schemas and privileges; semi-automated maintenance of coordination formula and domain relations; new query optimizations and constraints on query propagation; and data advertisement to spawn interest groups.

The designers of Piazza also avoided a global schema (Halevy, Ives et al. 2003a; Halevy, Ives et al. 2003b; Tatarinov, Mork et al. 2003; Tatarinov and Halevy 2004). They contend that the conventional data integration architecture, with a slowly evolving, global, mediated schema, "requires centralized administration and schema design and does not scale to large numbers of small-scale collaborations" (Halevy, Ives et al. 2003a). Piazza has two kinds of schema mappings: "peer descriptions" map between each peer's "view of the world" and are used to route queries; "storage descriptions" map a peer's "view of the world" to the specific data at the peer. The Piazza researchers were also concerned about interworking between the widely used eXtensible Markup Language (XML) and the emerging Resource Description Framework (RDF), with its richer mechanisms for capturing the classes, properties and relationships between objects. They devised a language based on XQuery (Boag, Chamberlin et al. 2003) that maps between XML and RDF nodes. They designed an algorithm for answering queries through chained directional mappings. Although this problem of answering queries using views is well understood for relational data, a solution was required for nested XML data. Sartiani et al. pointed out that some human effort is required to setup schema mappings (Sartiani, Manghi et al. 2004). For future work, the authors are considering caching and replication mechanisms for higher reliability and performance.

The Chatty Web (Aberer, Cudre-Maroux et al. 2004) is complementary to Piazza. Whereas Piazza assumes pairwise schema mappings, the objective of the Chatty Web is to improve the quality of the mappings over time. Aberer, Cudre-Maroux et al. advocated two measures of schema mapping quality. *Syntactic similarity* indicates the extent of information lost from queries when attributes of one schema do not exist in another. *Semantic similarity* indicates how well peers agree on the meaning of translations, at both the schema and data levels. Each query is augmented with these measures, which are updated as the query propagates through the network. In this way, peers agree globally by interacting locally.

Hyperion (Arenas, Kantere et al. 2003) was inspired by the local relational model (Bernstein, Giunchiglia et al. 2002). Whereas the Chatty Web work addressed the quality of schema mappings between peers, Hyperion proposes specific mechanisms to ensure consistency across peers. Each peer has an *acquaintance manager* that constructs mapping tables and expressions when peers meet. Each has a *query manager* that either resolves a query locally or routes it to other peers using the mapping tables. The *rule manager* uses active event-condition-action rules to achieve consistency between acquainted peers. Initially, the mapping tables were built manually, but the Hyperion designers foreshadowed semi-automated construction.

PeerDB (Ng, Ooi et al. 2003) has the qualities seen in the other PDMSs: there is no global schema; peers can join and leave at any time; peers route queries towards a target; query results may be incomplete if peers are not active. In addition to these, PeerDB adds the use of mobile agents. Ng, Ooi et al. claim that agent technology makes it easier to summarize and filter query results (Ng, Ooi et al. 2003). PeerDB also ensures that the "best" peers are topologically close. The PeerDB network relies on a small number of *location independent global names lookup servers* to track the IP address and status of every peer on the network. Queries have two phases. Firstly *relation matching agents* use information retrieval measures like precision and recall to find promising peers. Secondly *data retrieval agents* translate and submit SQL queries to those peers, which send results to the master agent originating the query. Experiments were conducted on a test bed of 32 nodes to measure query response times. PeerDB was reportedly the first PDMS implementation (Ng, Ooi et al. 2002; Arenas, Kantere et al. 2003).

5. Search

Database research suggests *directions* for P2P research. Hellerstein observed that, while work on fast P2P indexes is well underway, P2P search optimization remains a promising topic for future research (Hellerstein 2003). Kossman reviewed the state of the art of distributed query processing, highlighting areas for future research: simulation and query optimization for networks of ten thousands of servers and millions of client; non-relational data types like XML, text and images; and partial query responses since on the Internet "failure is the rule rather than the exception" (Kossmann 2000). A primary motivation for the P2P system, PIER, was to scale from the largest database systems of a few hundred nodes to an Internet environment in which there are over 160 million nodes (Huebsch, Hellerstein et al. 2003). Litwin and Sahri have also considered ways to combine distributed hashing, more specifically the Scalable Distributed Data Structures, with SQL databases, claiming to be first to implement scalable distributed database partitioning (Litwin and Sahri 2004). Motivated by the lack of transparent distribution in current distributed databases, they measure query execution times for Microsoft SQL servers aggregated by means of an SDDS layer. One of their starting assumptions was that it is too challenging to change the SQL query optimizer.

Database research also suggests the *approach* to P2P research. Researchers of database query optimization were divided between those looking for optimal solutions in special cases and those using heuristics to answer all queries (Jarke and Koch 1984). Gribble et al. cast query optimization in terms of the data placement problem, which is to "distribute data and work so the full query workload is answered with lowest cost under the existing bandwidth and resource constraints" (Gribble, Halevy et al. 2001). They pointed out that even the static version of this problem is NP-complete in P2P networks. Consequently, research on massive, dynamic P2P networks will likely progress using both strategies of early database research - heuristics and special-case optimizations.

If P2P networks are going to be adaptable, if they are to support a wide range of applications, then they need to accommodate many search types (Harren, Hellerstein et al. 2002). Up to this point, we have reviewed queries for keys (Section 3) and keywords (Sections 4.1and 4.2). Unfortunately, a major shortcoming of the DHTs in Sections 3.1 to 3.5 is that they primarily support exact-match, single-key queries. Skip Graphs support range and prefix queries, but not aggregation queries. We have also seen early proposals to support query languages (Section 4.3). Here we probe below the language syntax to identify the open research issues associated with more expressive P2P queries (Daswani, Garcia-Molina et al. 2003). Triantafillou and Pitoura observed the disparate P2P designs for different types of queries and so outlined a unifying framework (Triantafillou and Pitoura 2003). To classify queries, they considered the number of relations (single or multiple), the number of attributes (single or multiple) and the type of query operator. They described numerous operators: equality, range, join and "special functions". The latter referred to aggregation (like sum, count, average, minimum and maximum), grouping and ordering. The following sections approximately fit their taxonomy - range queries, multi-attribute queries, join queries and aggregation queries. There has been some initial P2P work on other query types - continuous queries (Gedik and Liu 2003b; Gedik and Liu 2003a; Huebsch, Hellerstein et al. 2003), recursive queries (Huebsch, Hellerstein et al. 2003; Loo, Huebsch et al. 2004a) and adaptive queries (Avnur and Hellerstein 2000; Hellerstein 2003). For these, we defer to the primary references.

5.1. Range Queries

The support of efficient range predicates in P2P networks was identified as an important open research issue by Huebsch et al. (Huebsch, Hellerstein et al. 2003). Range partitioning has been important in parallel databases to improve performance, so that a transaction commonly needs data from only one disk or node (Huebsch, Hellerstein et al. 2003). One type of range search, longest prefix match, is important because of its prevalence in routing schemes for voice and data networks alike. In other applications, users may pose broad, inexact queries, even though they require only a small number of responses. Consequently techniques to locate similar ranges are also important (Gupta, Agrawal et al. 2003). Various proposals for ranges searches over P2P networks are summarized in Figure 3. Since the Scalable Distributed Data Structure (SDDS) has been an important influence on contemporary Distributed Hash Tables (DHTs) (Devine 1993; Litwin, Neimat et al. 1996; Karger, Lehman et al. 1997), we also include ongoing work on SDDS range searches.

The papers on P2P range search can be divided into those that rely on an underlying DHT (the first four entries in (Figure 3) and those that do not (the subsequent three entries). Bharambe, Agrawal et al. argued that DHTs are inherently ill-suited to range queries (Bharambe, Agrawal et al. 2004). The very feature that makes for their good load balancing properties, randomized hash functions, works against range queries. One possible solution would be to hash ranges, but this can require a priori partitioning. If the partitions are too large, partitions risk overload. If they are too small, there may be too many hops.

Technique ***(Underlying DHT*** [1]***)***	**Reference**
Peer-to-Peer (P2P)	
Locality Sensitive Hashing *(Chord)*	(Gupta, Agrawal et al. 2003)
Prefix Hash Trees *(unspecified DHT)*	(Ratnasamy, Francis et al. 2003; Ramabhadran, Ratnasamy et al. 2004)
Space Filling Curves *(CAN)*	(Andrzejak and Xu 2002)
Space Filling Curves *(Chord)*	(Schmidt and Parashar 2004)
Quadtrees *(Chord)*	(Tanin, Harwood et al. 2004)
Skip Graphs	(Aspnes and Shah 2003; Harvey, Jones et al. 2003a; Aspnes, Kirsch et al. 2004; Ganesan, Bawa et al. 2004)
Mercury	(Bharambe, Agrawal et al. 2004)
P-Grid	(Aberer 2002b; Aberer, Datta et al. 2003)
Scalable Distributed Data Structures (SDDS)	
RP*	(Litwin, Neimat et al. 1994; Tsangou, Ndiaye et al. 2003)

Note 1. Although several of the authors based their work on one particular DHT, it may be possible to port their work to others.

Figure 3 Solutions for Range Queries on P2P and SDDS indexes.

Despite these potential shortcomings, there have been several range query proposals based on DHTs. If hashing ranges to nodes, it is entirely possible that overlapping ranges map to different nodes. Gupta, Agrawal et al. rely on locality sensitive hashing to ensure that, with high probability, similar ranges are mapped to the same node (Gupta, Agrawal et al. 2003). They propose one particular family of locality sensitive hash functions, called min-wise independent permutations. The number of partitions per node and the path length were plotted against the total numbers of peers in the system. For a network with 1000 nodes, the hop-count distribution was very similar to that of the exact-matching Chord scheme. Was it load-balanced? For the same network with 50,000 partitions, there were over two orders of magnitude variation in the number of partitions at each node (first and ninety-ninth percentiles). The Prefix Hash Tree is a trie in which prefixes are hashed onto any DHT. The preliminary analysis suggests efficient doubly logarithmic lookup, balanced load and fault resilience (Ratnasamy, Francis et al. 2003; Ramabhadran, Ratnasamy et al. 2004). Andrzejak and Xu were perhaps the first to propose a mapping from ranges to DHTs (Andrzejak and Xu 2002). They use one particular Space Filling Curve, the Hilbert curve, over a Content Addressable Network (CAN) construction (Section 3.3). They maintain two properties: nearby ranges map to nearby CAN zones; if a range is split into two sub-ranges, then the zones of the sub-ranges partition the zone of the primary range. They plot path length and load proxy measures (the total number of messages and nodes visited) for three algorithms to propagate range queries: brute force; controlled flooding and directed controlled flooding. Schmidt and Parashar also advocated Space Filling Curves to achieve range queries over a DHT (Schmidt and Parashar 2004). However they point out that, while Andrzejak and Xu use an inverse Space Filling Curve to map a one-dimensional space to d-dimensional zones, they map a d-dimensional space back to a one-dimensional index. Such a construction gives the ability to search across multiple attributes (Section 5.2). Tanin, Harwood et al. suggested quadtrees over Chord (Tanin, Harwood et al. 2004), and gave preliminary simulation results for query response times.

Because DHTs are naturally constrained to exact-match, single-key queries, researchers have considered other P2P indexes for range searches. Several were based on Skip Graphs (Aspnes and Shah 2003; Harvey, Jones et al. 2003a) which, unlike the DHTs, do not necessitate randomizing hash functions and are therefore capable of range searches. Unfortunately, they are not load balanced (Ganesan, Bawa et al. 2004). For example, in SkipNet (Harvey, Dunagan et al. 2002), hashing was added to balance the load - the Skip Graph could support range searches or load balancing, but not both. One solution for load-balancing relies on an increased number of 'virtual' servers (Rao, Lakshminarayanan et al. 2003) but, in their search for a system that can both search for ranges and balance loads, Bharambe, Agrawal et al. rejected the idea (Bharambe, Agrawal et al. 2004). The virtual servers work assumed load imbalance stems from hashing, that is, by skewed data insertions and deletions. In some situations, the imbalance is triggered by a skewed query load. In such circumstances, additional virtual servers can increase the number of routing hops and increase the number of pointers that a Skip Graph needs to maintain. Ganesan, Bawa et al. devised an alternate method to balance load (Ganesan, Bawa et al. 2004). They proposed two Skip Graphs, one to index the data itself and the other to track load at each node in the system. Each node is able to determine the load on its neighbours and the most (least) loaded nodes in the system. They devise two algorithms: NBRADJUST balances load on neighbouring nodes; using REORDER, empty nodes can take over some of the tuples on heavily loaded nodes. Their simulations focus on skewed storage load, rather than on skewed query loads, but they surmise that the same approach could be used for the latter.

Other proposals for range queries avoid both the DHT and the Skip Graph. Bharambe, Agrawal et al. distinguish their Mercury design by its support for multi-attribute range queries and its explicit load balancing (Bharambe, Agrawal et al. 2004). In Mercury, nodes are grouped into routing hubs, each of which is responsible for various query attributes. While it does not use hashing, Mercury is loosely similar to the DHT approaches: nodes within hubs are arranged into rings, like Chord (Stoica, Morris et al. 2003); for efficient routing within hubs, k long-distance links are used, like Symphony (Manku, Bawa et al. 2003). Range lookups require $O(\log^2 n / k)$ hops. Random sampling is used to estimate the average load on nodes and to find the parts of the overlay that are lightly loaded. Whereas Symphony assumed that nodes are responsible for ranges of approximately equal size, Mercury's random sampling can determine the location of the start of the range, even for non-uniform ranges (Bharambe, Agrawal et al. 2004). P-Grid (Aberer, Cudre-Mauroux et al. 2003) does provide for range queries, by virtue of the key ordering in its tree structures. Ganesan, Bawa et al. critiqued its capabilities (Ganesan, Bawa et al. 2004): P-Grid assumes fixed-capacity nodes; there was no formal characterization of imbalance ratios or balancing costs; every P-Grid periodically contacts other nodes for load information.

The work on Scalable Distributed Data Structures (SDDSs) has progressed in parallel with P2P work and has addressed range queries. Like the DHTs above, the early SDDS Linear Hashing (LH*) schemes were not order-preserving (Litwin, Niemat et al. 1993). To facilitate range queries, Litwin, Niemat et al. devised a Range Paritititioning variant, RP* (Litwin, Neimat et al. 1994). There are options to dispense with the index, to add indexes to clients and to add them to servers. In the variant without an index, every query is issued via multicasting. The other variants also use some multicasting. The initial RP* paper suggested scalability to thousands of sites, but a more recent RP* simulation was capped at 140 servers (Tsangou, Ndiaye et al. 2003). In that work, Tsangou, Ndiaye et al. investigated TCP and UDP mechanisms by which servers could return range query results to clients. The primary metrics were search and response times. Amongst the commercial parallel database management systems, they reported that the largest seems only to scale to 32 servers (SQL Server 2000). For future work, they planned to explore aggregation of query results, rather than establishing a connection between the client and every single server with a response.

All in all, it seems there are numerous open research questions on P2P range queries. How realistic is the maintenance of global load statistics considering the scale and dynamism of P2P networks? Simulations at larger scales are required. Proposals should take into account both the storage load (insert and delete messages) and the query load (lookup messages). Simplifying assumptions need to be attacked. For example, how well do the above solutions work in networks with heterogeneous nodes, where the maximum message loads and index sizes are node-dependent?

5.2. Multi-Attribute Queries

There has been some work on multi-attribute P2P queries. As late as September 2003, it was suggested that there has not been an efficient solution (Triantafillou and Pitoura 2003).

Again, an early significant work on multi-attribute queries over aggregated commodity nodes germinated amongst SDDSs. k-RP* (Litwin and Neimat 1996) uses the multi-dimensional binary search tree (or k-d tree where k indicates the number of dimensions of the search index) (Bentley 1975). It builds on the RP* work from the previous section and inherits their capabilities for range search and partial match. Like the other SDDSs, k-RP* indexes can fit into RAM for very fast lookup. For future work, Litwin and Neimat suggested a) a formal analysis of the range search termination algorithm and the k-d paging algorithm, b) a comparison with other multi-attribute data structures (quad-trees and R-trees) and c) exploration of query processing, concurrency control and transaction management for k-RP* files, and (Litwin and Neimat 1996). On the latter point, others have considered transactions to be inconsequential to the core problem of supporting more complex queries in P2P networks (Harren, Hellerstein et al. 2002).

In architecting their secure wide-area Service Discovery Service (SDS), Hodes, Czerwinski et al. considered three possible designs for multi-criteria search – Centralization, Mapping and Flooding (Hodes, Czerwinski et al. 2002). These correlate to the index classifications of Section 2 – Central, Distributed and Local. They discounted the centralized, Napster-like index for its risk of a single point of failure. They considered the hash-based mappings of Section 3 but concluded that it would not be possible to adequately partition data. A document satisfying many criteria would be wastefully stored in many partitions. They rejected full flooding for its lack of scalability. Instead, they devised a query filtering technique, reminiscent of Gnutella's query routing protocol (Section 4.1). Nodes push proactive summaries of their data rather than waiting for a query. Summaries are aggregated and stored throughout a server hierarchy, to guide subsequent queries. Some initial prototype measurements were provided for total load on the system, but not for load distribution. They put several issues forward for future work. The indexing needs to flexible to change according to query and storage workloads. A mesh topology might improve on their hierarchic topology since query misses would not propagate to root servers. The choice is analogous to BGP meshes and DNS trees. They also wanted to develop a solution for mobile clients.

More recently, Cai, Frank et al. devised the Multi-Attribute Addressable Network (MAAN) (Cai, Frank et al. 2003). They built on Chord to provide both multi-attribute and range queries, claiming to be the first to service both query types in a structured P2P system. Each MAAN node has O(log n) neighbours. MAAN multi-attribute range queries require $O(\log N + N \times s_{\min})$ hops, where $s_{\min}$ is the minimum range selectivity across all attributes. Selectivity is the ratio of the query range to the entire identifier range. The paper assumed that a locality preserving hash function would ensure balanced load. Per Section 5.1, the arguments by Bharambe, Agrawal et al. have highlighted the shortcomings of this assumption (Bharambe, Agrawal et al. 2004). MAAN required that the schema must be fixed and known in advance – adaptable schemas were recommended for subsequent attention. The authors also acknowledged that there is a selectivity breakpoint at which full flooding becomes more efficient than their scheme. This begs for a query resolution algorithm that adapts to the profile of queries. Cai and Frank followed up with RDFPeers (Cai and Frank 2004)(Section4.3, Figure 2). They differentiate their work from other RDF proposals by a) guaranteeing to find query results if they exist and b) removing the requirement of prior definition of a fixed schema. They hashed <subject, predicate, object> triples onto the MAAN and reported routing hop metrics for their implementation. Load imbalance across nodes was reduced to less than one order of magnitude, but the specific measure was number of triples stored per node – skewed query loads were not considered. They plan to improve load balancing with the virtual servers of Section 5.1 (Rao, Lakshminarayanan et al. 2003).

5.3. Join Queries

Two research teams have done some initial work on P2P join operations. Harren, Hellerstein et al. initially described a three-layer architecture – storage, DHT and query processing. The implemented the join operation by modifying an existing Content Addressable Network (CAN) simulator, reporting "significant hot-spots in all dimensions: storage, processing and routing" (Harren, Hellerstein et al. 2002). They progressed their design more recently in the context of PIER, a distributed query engine based on CAN (Huebsch, Hellerstein et al. 2003; Chun, Stoica et al. 2004). They implemented two equi-join algorithms. In their design, a key is constructed from the "namespace" and the "resource ID". There is a namespace for each relation and the resource ID is the primary key for base tuples in that relation. Queries are multicast to all nodes in the two namespaces (relations) to be joined. Their first algorithm is a DHT version of the symmetric hash join. Each node in the two namespaces finds the relevant tuples and hashes them to a new query namespace. The resource ID in the new namespace is the concatenation of join attributes. In the second algorithm, called "fetch matches", one of the relations is already hashed on the join attributes. Each node in the second namespace finds tuples matching the query and retrieves the corresponding tuples from the the first relation. They leveraged two other techniques, namely the symmetric semi-join rewrite and the Bloom filter rewrite, to reduce the high bandwidth overheads of the symmetric hash join. For an overlay of 10,000 nodes, they simulated the delay to retrieve tuples and the aggregate network bandwidth for these four schemes. The initial prototype was on a cluster of 64 PCs, but it has more recently been expanded to PlanetLab.

Triantafillou and Pitoura considered multicasting to large numbers of peers to be inefficient (Triantafillou and Pitoura 2003). They therefore allocated a limited number of special peers, called range guards. The domain of the join attributes was divided, one partition per range guard. Join queries were sent only to range guards, where the query was executed. Efficient selection of range guards and a quantitive evaluation of their proposal were left for future work.

5.4. Aggregation Queries

Aggregation of information from large groups of peers is important and challenging. One example of an aggregation function is Count. Daswani, Garcia-Molina et al. identified the need for further research on more expressive aggregation functions like Sum, Max and Median (Daswani, Garcia-Molina et al. 2003). Such functions play a part in system management, data sharing and caching, IP address aggregation, DNS resolution, multicast tree formation and load balancing (Yalagandula and Dahlin 2004). They support key distributed systems functions like leader election, voting, resource location, object placement and error recovery (van Renesse 2003). As late as September 2003, it was reported that it remains unclear how to efficiently aggregate statistics from large DHT networks (Huebsch, Hellerstein et al. 2003). The dynamism of such networks frustrates the problem further.

Much of the research on P2P aggregation has been influenced by Astrolabe, a P2P system for large-scale distributed information management from Cornell University (van Renesse, Birman et al. 2003). Cone differentiated itself from Astrolabe by its emphasis on range queries and heap functions (Bhagwan, Varghese et al. 2003). The Distributed Approximative System Information Service was designed for incorporation into DHTs whereas Astrolabe is standalone (Albrecht, Arnold et al. 2004). Like Astrolabe, the work by Montresor, Jelasity et al. used epidemic propagation of information but dispensed with Astrolabe-like hierarchy (Montresor, Jelasity et al. 2003). Similarly, Newscast has less structure and is more light-weight than Astrolabe (Jelasity, Kowalczyk et al. 2004). Astrolabe was one of two primary motivations for the Scalable Distributed Information Management System (Yalagandula and Dahlin 2004). The other was the DHT. In terms of its hierarchic architecture, Zhang, Shi et al. considered their Self-Organized Metadata Overlay (SOMO) to be most like Astrolabe (Zhang, Shi et al. 2003). The main difference is that SOMO builds on DHTs. Bawa, Gionis et al. saw the need to qualify the query semantics of best-effort aggregation algorithms, such as that used in Astrolabe (Bawa, Gionis et al. 2004). They attach a correctness condition, called single site validity, to qualify aggregation results returned from a large network in which the likelihood of a single node failure is high. Willow combines aggregation, multicasting, DHT lookups and publish/subscribe capabilities in one algorithm (van Renesse and Bozdog

2004). One of its authors, Robbert van Renesse, was a co-author of the Astrolabe paper. Willow supports a wider range of queries than Astrolabe, with faster responses. It uses Plaxton routing similar to that of Kadlemia (Maymounkov and Mazieres 2002). It also provides a tree repair algorithm by which disjoint or broken trees can be joined in O(log N) time. The contributions of each of these aggregation proposals is summarized in Figure 4. Also shown are two specific aggregation algorithms. The first addressed size estimation for large groups (Psaltoulis, Kostoulas et al. 2004). Another Astrolabe author, Ken Birman, is one of its designers. P2P traces were used in simulations to explore performance for both static and dynamic groups. In the second, Cao and Wang devised a "top k" aggregation algorithm that reduces the network traffic by one to two orders of magnitude when compared to existing algorithms (Cao and Wang 2004). In future work, they will be investigating the various "top k" design choices in P2P networks.

Learning from his Astrolabe experience, van Renesse identified numerous aggregation issues worthy of future research (van Renesse 2003). Networks that aggregate information lazily are *weakly consistent*, so applications need to be able to balance consistency against performance or scalability. Applications need to be able to manage the timeliness or *staleness* of aggregation results. Load balancing, for example, risks instability if estimates of network or processor load are old and inaccurate. Several of the above proposals use epidemics which are scalable and robust, but have *high latency*. Applications need to be able to select *appropriate hierarchy*. Some of the above proposals eschewed hierarchy. Astrolabe's hierarchy was manually configured. Willow uses the hierarchy inherent in Plaxton routing to aid aggregation. Whereas Willow uses a single tree, SOMO and SDIMS both use one tree per key (van Renesse and Bozdog 2004). Such an approach frustrates multi-attribute aggregation (Yalagandula and Dahlin 2004). In Astrolabe, the size of the aggregation's *attribute set was limited* to ensure scalability of the epidemic protocols. Techniques like compression are required to improve the number and scope of simultaneous aggregations. From their SDIMS implementation, Yalagandula and Dahlin also outline open lines of research that lie close to the dependability and adaptability concerns of this survey (Yalagandula and Dahlin 2004). They found reconfigurations at the aggregation layer to be costly – techniques are required to reduce the cost and frequency of such reconfigurations. Applications need to be able to tune the algorithms according to the read/write profiles of the data to be aggregated.

6. Conclusions

Research on peer-to-peer networks can be divided into four categories – search, storage, security and applications. This critical survey has focused on search methods. While P2P networks have been classified by the existence of an index (structured or unstructured) or the location of the index (local, centralized and

Proposal	Reference	Contribution
P2P Aggregation Overlays		
Astrolabe	(van Renesse, Birman et al. 2003)	Aggregation of large scale system state using: hierarchic zones for scaling; dynamic installation of SQL aggregation queries; epidemics; certificates
Cone	(Bhagwan, Varghese et al. 2003)	Augments DHTs with a trie which has an aggregation operator (max, min, sum…) at the root of every subtree
Distributed Approximative System Information Service (DASIS)	(Albrecht, Arnold et al. 2003; Albrecht, Arnold et al. 2004)	Aggregation for load-balanced join
(Epidemic aggregation)	(Montresor, Jelasity et al. 2003)	Aggregation that is robust to communication and node failures
Newscast	(Jelasity, Kowalczyk et al. 2004)	News correspondents epidemically receive news from local agent and send news updates to small random set of peers
Scalable Distributed Information Management System (SDIMS)	(Yalagandula and Dahlin 2004)	Aggregation functions installed at DHT nodes for particular aggregation types. Flexible API to control read/write propagation. Administrative autonomy. Lazy and on-demand reaggregation.
Self-Organized Metadata Overlay (SOMO)	(Zhang, Shi et al. 2003)	Arbitrary data structures on DHTs
WildFire	(Bawa, Gionis et al. 2004)	Single-site validity, price of validity
Willow	(van Renesse and Bozdog 2004)	Aggregation, DHT routing, multicast, and publish/subscribe in one simple, location-aware protocol; Tree repair
Aggregation Algorithms for P2P		
Hops Sampling and Interval Density	(Psaltoulis, Kostoulas et al. 2004)	Estimation of number and density of processes in dynamic groups of peers. Uses epidemics.
Three Phase Uniform Threshold (TPUT) Algorithm	(Cao and Wang 2004)	Find the k objects with the highest aggregate values. Improves on efficiency of existing algorithms by orders of magnitude. Potential for P2P application.

Figure 4 Proposals for P2P Aggregation Queries

distributed), this survey has shown that most have evolved to have some structure, whether it is indexes at superpeers or indexes defined by DHT algorithms. As for location, the distributed index is most common. The survey has characterized indexes as semantic and semantic-free. It has also critiqued P2P work on major query types. While much of it addresses work from 2000 or later, we have traced important building blocks from the 1990s.

The initial motivation in this survey was to answer the question, "How robust are P2P search networks?" The question is key to the deployment of P2P technology. Balakrishnan, Kaashoek et al. argued that the P2P architecture is appealing: the startup and growth barriers are low; they can aggregate enormous storage and processing resources; "the decentralized and distributed nature of P2P systems gives them the *potential* to be robust to faults or intentional attacks" (Balakrishnan, Kaashoek et al. 2003). If P2P is to be a disruptive technology in applications other than simple file sharing, then robustness needs to be practically verified (Gedik and Liu 2003b). We have taken robustness to include dependability and adaptability.

Since the open issues collated in this survey are numerous, we conclude instead with general observations and suggestions for future P2P research:

1) Complete the companion P2P surveys for storage, security and applications. Critical surveys on the remaining three categories are left as a challenge to anyone prepared to grapple with the volume and pace of P2P research. A rough outline has been suggested in Figure 1, along with references. The need for such surveys was highlighted within the peer-to-peer research group of the Internet Research Task Force (IRTF) (Yeager 2003).

2) P2P indexes are maturing. P2P queries are embryonic. On indexes, there is consensus building in favour of the algorithmic families that are logarithmic in degree and diameter. The remains some dispute as to whether fixed degree constructs like the de Bruijn graph will yield practical advantage. Work on more expressive queries over P2P indexes started to gain momentum in 2003, but remains fraught with efficiency and load issues. If P2P networks are to be adaptable, they need to handle multiple types of queries, albeit with relaxed semantics.

3) Isolate individual mechanisms and evaluate their robustness. There is limited value in comparing robustness across broad algorithmic families (like rings versus de Bruijn graphs), when it is highly sensitive to underlying mechanisms. Adhoc examples include reactive versus periodic recovery of peers, lookup timeouts, choice of nearby neighbours, TCP timeouts and UDP heartbeats for detection, 'surrogate routing', bias toward higher-degree peers, one-hop replication and receiver-controlled token-based flow control. A more systematic approach would be instructive.

4) Build on the full body of related research. P2P research cuts across numerous research disciplines – databases, networking, operating systems, information retrieval, security, algorithms and distributed systems more generally. Some of the more interesting solutions will come at the seams. There is value in monitoring and leveraging related research streams. One example is the recent cross-pollination between P2P and Scalable Distributed Data Structures.

5) Build consensus on robustness metrics and their acceptable ranges. This paper has teased out numerous measures that impinge on robustness, for example, the median query path length for a failure of x% of nodes, bisection width, path overlap, the number of alternatives available for the next hop, lookup latency, average live bandwidth (bytes/node/sec), successful routing rates, the number of timeouts (caused by a finger pointing to a departed node), lookup failure rates (caused by nodes that temporarily point to the wrong successor during churn) and clustering measures (edge expansion and node expansion). Application-level robustness metrics need to drive a consistent assessment of the underlying search mechanics.

7. Bibliography

Aberer, K. (2002a). "Efficient Search in Unbalanced, Randomized Peer-to-Peer Search Trees." EPFL Technical Report IC/2002/79.

Aberer, K. (2002b). "Scalable Data Access in P2P Systems Using Unbalanced Search Trees." Workshop on Distributed Data and Structures (WDAS-2002), Paris, France.

Aberer, K., P. Cudre-Maroux, M. Hauswirth and T. van Pelt (2004). "Start making sense: The Chatty Web approach for global semantic agreements." Journal of Web Semantics **1**(1). December

Aberer, K., P. Cudre-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva and R. Schmidt (2003). "P-Grid: a self-organizing structured P2P system." ACM SIGMOD Record **32**(3): 29-33.

Aberer, K., A. Datta and M. Hauswirth (2003). "The Quest for Balancing Peer Load in Structured Peer-to-Peer Systems." Technical Report IC/2003/32, Ecole Polytechnique Federale de Lausanne (EPFL), Lausanne, Switzerland.

Aberer, K., A. Datta and M. Hauswirth (2004). "A decentralized public key infrastructure for customer-to-customer e-commerce." International Journal of Business Process Integration and Management.

Aberer, K. and M. Hauswirth (2001). "Peer-to-peer information systems: concepts and models, state-of-the-art, and future systems." ACM SIGSOFT Software Engineering Notes, Proceedings of the 8th European software engineering conference held jointly with 9th ACM SIGSOFT international symposium on foundations of software engineering **26**(5).

Aberer, K. and M. Hauswirth (2002). "An Overview on Peer-to-Peer Information Systems." Workshop on Distributed Data and Structures (WDAS-2002), Paris, France.

Abraham, I., D. Malkhi and O. Dubzinski (2004). "LAND:Stretch (1+epsilon) Locality Aware Networks for DHTs." Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA04), New Orleans, LA.

Adamic, L., R. Lukose, A. Puniyani and B. Huberman (2001). "Search in power-law networks." Physical review E, The American Physical Society **64**(046135).

Adler, M., E. Halperin, R. Karp and V. Vazirani (2003). "A stochastic process on the hypercube with applications to peer-to-peer networks." Proceedings of the 35th ACM symposium on Theory of Computing, San Diego, California, USA: 575-584.

Adya, A., R. Wattenhofer, W. Bolosky, M. Castro, G. Cermak, R. Chaiken, J. Douceur, J. Howell, J. Lorch and J. Thiemer (2002). "Farsite: federated, available and reliable storage for an incompletely trusted environment." ACM SIGOPS Operating Systems Review, Special issue on Decentralized storage systems: 1-14.

Ahlgren, B., M. Brunner, L. Eggert, R. Hancock and S. Schmid (2004). "Invariants: a new design methodology for network architectures." Proceedings of the ACM SIGCOMM workshop on Future Direction in Network Architecture, Portland, Oregon, USA, August 30: 65-70.

Ajmani, S., D. Clarke, C.-H. Moh and S. Richman (2002). "ConChord: Cooperative SDSI Certificate Storage and Name Resolution." First International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, March.

Albrecht, K., R. Arnold and R. Wattenhofer (2003). "Join and Leave in Peer-to-Peer Systems: The DASIS Approach." Technical Report 427, Department of Computer Science, ETH Zurich, November.

Albrecht, K., R. Arnold and R. Wattenhofer (2004). "Aggregating Information in Peer-to-Peer Systems for Improved Join and Leave." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

Alima, L., S. El-Ansary, P. Brand and S. Haridi (2003). "DKS(N,k,f): a family of low communication, scalable and fault-tolerant infrastructures for P2P applications." Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid: 344-350.

Anagnostakis, K. and M. Greenwald (2004). "Exchange-based Incentive Mechanism for Peer-to-Peer File Sharing." Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), March 23-26, Tokyo, Japan.

Andrzejak, A. and Z. Xu (2002). "Scalable, efficient range queries for grid information services." Proceedings of the Second IEEE International Conference on Peer to Peer Computing, Linkoping University, Sweden, September.

Annexstein, F., K. Berman, M. Jovanovic and K. Ponnavaikko (2002). "Indexing techniques for file sharing in scalable peer to peer networks." 11th IEEE International Conference on Computer Communications and Networks: 10-15.

Arenas, M., V. Kantere, A. Kementsietsidis, I. Kiringa, R. Miller and J. Mylopoulos (2003). "The hyperion project: from data integration to data collection." ACM SIGMOD Record **32**(3): 53-58. September

Aspnes, J., J. Kirsch and A. Krishnamurthy (2004). "Load Balancing and Locality in Range-Queriable Data Structures." Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), St. John's, Newfoundland, Canada, July 25-28.

Aspnes, J. and G. Shah (2003). "Skip graphs." Proceedings of the 14th annual ACM-SIAM symposium on discrete algorithms: 384-393. January

Avnur, R. and J. Hellerstein (2000). "Eddies: continuously adaptive query processing." Proceedings of the 2000 ACM SIGMOD international conference on Management of Data, Dallas, Texas: 261-272.

Awerbuch, B. and C. Scheideler (2003). "Peer-to-peer systems for prefix search." Proceedings of the 22nd annual ACM Symposium on Principles of Distributed Computing, Boston, Massachusetts: 123-132.

Awerbuch, B. and C. Scheideler (2004a). "Consistent, Order-Preserving Data Management in Distributed Storage Systems." Proceedings of the Sixteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA 2004), Barcelona, Spain, June 27-30: 44-53.

Awerbuch, B. and C. Scheideler (2004b). "Robust Distributed Name Service." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Babaoglu, O., H. Meling and Montresor (2002). "Anthill: a framework for the development of agent-based peer-to-peer systems." Proceedings of IEEE International Conference on Distributed Computer systems: 15-22.

Baeza-Yates, R. and B. Ribeiro-Neto (1999). Modern Information Retrieval, Addison Wesley.

Bailey, N. (1975). Epidemic Theory of Infectious Diseases and its Applications, Hafner Press, Second Edition.

Balakrishnan, H., M. F. Kaashoek, D. Karger, R. Morris and I. Stoica (2003). "Looking up data in P2P systems." Communications of the ACM **46**(2): 43-48.

Balakrishnan, H., S. Shenker and M. Walfish (2003). "Semantic-Free Referencing in Linked Distributed Systems." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Balazinska, M., H. Balakrishnan and D. Karger (2002). "INS/Twine: A scalable Peer-to-Peer architecture for Intentional Resource Discovery." Proceedings of the First International Conference on Pervasive Computing (IEEE).

Banaei-Kashani, F. and C. Shahabi (2003). "Criticality-based analysis and design of unstructured peer-to-peer networks as "complex systems"." Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid: 351-358.

Baquero, C. and N. Lopes (2003). "Towards peer to peer content indexing." ACM SIGOPS Operating Systems Review **37**(4): 90-96.

Bawa, M., R. Bayardo, S. Rajagopalan and E. Shekita (2003). "Make it fresh, make it quick: searching a network of personal webservers." Proceedings of the 12th international conference on World Wide Web: 577-586.

Bawa, M., A. Gionis, H. Garcia-Molina and R. Motwani (2004). "The price of validity in dynamic networks." Proceedings of the 2004 ACM SIGMOD International Conference on the management of data, Paris, France: 515-526.

Bawa, M., G. S. Manku and P. Raghavan (2003). "SETS: search enhanced by topic segmentation." Proceedings of the 26th annual international ACM SIGIR conference on Research and Development in Information Retrieval: 306-313.

Bawa, M., Q. Sun, P. Vinograd, B. Yang, B. Cooper, A. Crespo, N. Daswani, P. Ganesan, H. Garcia-Molina, S. Kamvar, S. Marti and M. Schlossed (2003). "Peer-to-peer research at Stanford." ACM SIGMOD Record **32**(3): 23-28. Sept

Bentley, J. L. (1975). "Multidimensional binary search trees used for associative searching." Communications of the ACM **18**(9): 509-517.

Berket, K., A. Essiari and A. Muratas (2004). "PKI-Based Security for Peer-to-Peer Information Sharing." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

Bernstein, P. A., F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini and I. Zaihrayeu (2002). "Data management for peer-to-peer computing: A vision." Proceedings of the Fifth International Workshop on the Web and Databases.

Bhagwan, R., G. Varghese and G. Voelker (2003). "Cone: Augmenting DHTs to support distributed resource discovery." Technical Report, CS2003-0755, UC, San Diego, July.

Bharambe, A., M. Agrawal and S. Seshan (2004). "Mercury: Supporting Scalable Multi-Attribute Range Queries." SIGCOMM'04, Portland, Oregon, USA, Aug 30-Sept 3.

Birman, K. (2003). "The Surprising Power of Epidemic Communication." Springer-Verlag Heidelberg Lecture Notes in Computer Science **Volume 2584/2003**: 97-102.

Birman, K. and I. Gupta (2004). "Building Scalable Solutions to Distributed Computing Problems using Probabilistic Components." Submitted to The International Conference on Dependable Systems and Networks (DSN-2004), Dependable Computing and Computing Symposium (DCCS), June 28-July 1. Florence, Italy.

Blaze, M., J. Feigenbaum and J. Lacy (2003). "Trust and Reputation in P2P networks." http://www.neurogrid.net/twiki/bin/view/Main/ReputationAndTrust.

Bloom, B. (1970). "Space/time trade-offs in hash coding with allowable errors." Communications of the ACM **13**(7): 422-426. July

Boag, S., D. Chamberlin, M. Fernandez, D. Florescu, J. Robie and J. Simeon (2003). "XQuery 1.0: An XML Query Language." W3C Working Draft. 12 November

Brickley, D. and R. V. Guha (2000). "W3C Resource Description Framework (RDF) Schema Specification." http://www.w3.org/TR/1998/WD-rdf-schema/, W3C Candidate Recommendation, March.

Buragohain, C., D. Agrawal and S. Subhash (2003). "A game theoretic framework for incentives in P2P systems." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 48-56.

Byers, J., J. Considine and M. Mitzenmacher (2003). "Simple Load Balancing for Distributed Hash Tables." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Byers, J., J. Considine, M. Mitzenmacher and S. Rost (2002). "Informed content delivery across adaptive overlay networks." Proceedings of the 2002 conference on applications, technologies, architectures and protocols for computer communications: 47-60.

Cai, M. and M. Frank (2004). "RDFPeers: a scalable distributed RDF repository based on a structured peer-to-peer network." Proceedings of the 13th conference on World Wide Web, New York, NY, USA, May 17-20: 650-657.

Cai, M., M. Frank, J. Chen and P. Szekely (2003). "MAAN: A Multi-Attribute Addressable Network for Grid Information Services." Proceedings of the International Workshop on Grid Computing, Phoenix, AZ., November.

Cao, P. and Z. Wang (2004). "Efficient top-K query calculation in distributed networks." Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), St. John's, Newfoundland, Canada, July 25-28: 206-215.

Caronni, G. and M. Waldvogel (2003). "Establishing trust in distributed storage providers." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 128-133.

Castro, M., M. Costa and A. Rowstron (2003). "Performance and dependability of structured peer-to-peer overlays." Microsoft Research Technical Report MSR-TR-2003-94, December. Also 2004 International Conference on Dependable Systems and Networks, Palazzo dei Congressi, Florence, Italy, June 28-July 1.

Castro, M., M. Costa and A. Rowstron (2004). "Should we build Gnutella on a structured overlay?" ACM SIGCOMM Computer Communication Review **34**(1): 131-136.

Castro, M., P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh (2003). "SplitStream: high-bandwidth multicast in cooperative environments." Proceedings of the 19th ACM symposium on operating systems principles, Bolton Landing, New York, USA: 298-313.

Castro, M., P. Druschel, A.-M. Kermarrec and A. Rowstron (2002). "SCRIBE: a large-scale and decentralized application-level multicast infrastructure." IEEE Journal on Selected Areas in Communications **20**(8, October).

Castro, M., P. Rodrigues and B. Liskov (2003). "BASE: Using abstraction to improve fault tolerance." ACM Transactions on Computer Systems **21**(3): 236-269.

Castro, P., J. Lee and A. Misra (2004). "CLASH: A Protocol for Internet-Scale Utility-Oriented Distributed Computing." Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), March 23-26, Tokyo, Japan.

Cates, J. (2003). "Robust and Efficient Data Management for a Distributed Hash Table." Master's Thesis, Massachusetts Institute of Technology, May.

Chander, A., S. Dawson, P. Lincoln and D. Stringer-Calvert (2002). "NEVRLATE: scalable resource discovery." Second IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID2002): 56-65.

Chawathe, Y., S. Ratnasamy, L. Breslau, N. Lanham and S. Shenker (2003). "Making gnutella-like P2P systems scalable." Proceedings of the 2003 conference on Applications, Technologies, Architectures and Protocols for Computer Communications, Karlsruhe, Germany, August 25-29, 2003: 407-418.

Cholvi, V., P. Felber and E. Biersack (2004). "Efficient Search in Unstructured Peer-to-Peer Networks." Proceedings of the Symposium on Parallel Algorithms and Architectures, July.

Christin, N. and J. Chuang (2003). "On the cost of participating in a peer-to-peer network." Technical Report, University of California, Berkeley, http://p2pecon.berkeley.edu/pub/TR-2003-12-cc.pdf.

Christin, N. and J. Chuang (2004). "On the Cost of Participating in a Peer-to-Peer Network." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Chun, B., I. Stoica, J. Hellerstein, R. Huebsch, S. Jeffery, B. T. Loo, S. Mardanbeigi, T. Roscoe, S. Rhea and S. Schenker (2004). "Proceedings of the 2004 ACM SIGMOD international conference on management of data, Paris, France, Demonstration Session." 935-936.

"Citeseer Scientific Literature Digital Library." (2004). http://citeseer.ist.psu.edu/.

Clark, D. (1988). "The design philosophy of the DARPA internet protocols." ACM SIGCOMM Computer Communication Review, Symposium proceedings on communications architectures and protocols **18**(4).

Clark, D., J. Wroclawski, K. Sollins and R. Braden (2002). Tussle in cyberspace: defining tomorrow's internet. Conference on Applications, Technologies, Architectures and Protocols for Computer Communications, Pittsburgh, Pennsylvania, USA, ACM Press.

Clarke, I. (1999). "A Distributed Decentralised Information Storage and Retrieval System." Undergraduate Thesis, Division of Informatics, University of Edinburgh.

Clarke, I. (2003). "Freenet's Next Generation Routing Protocol." http://freenetproject.org/index.php?page=ngrouting. 20th July

Clarke, I., S. Miller, T. Hong, O. Sandberg and B. Wiley (2002). "Protecting Free Expression Online with Freenet." IEEE Internet Computing **6**(1). Jan/Feb

Clarke, I., O. Sandberg, B. Wiley and T. Hong (2001). Freenet: A Distributed Anonymous Information Storage and Retrieval System. International Workshop on Design Issues in Anonymity and Unobservability. H. Federrath. New York, USA, Springer.

Clip2 (2000). "The Gnutella Protocol Specification." http://www.clip2.com.

Cohen, E., A. Fiat and H. Kaplan (2003). "Associative Search in Peer to Peer Networks: Harnessing Latent Semantics." IEEE Infocom 2003, The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies, March 30-April 3, San Francisco.

Cohen, E. and S. Shenker (2002a). "P2P and multicast: replication strategies in unstructured peer to peer networks." Proceedings of the 2002 conference on applications, technologies, architectures and protocols for computer communications: 177-190.

Cohen, E. and S. Shenker (2002b). "Replication strategies in unstructured peer to peer networks." Proceedings of the 2002 conference on applications, technologies, architectures and protocols for computer communications: 177-190.

Considine, J., M. Walfish and D. G. Andersen (2003). "A pragmatic approach to DHT adoption." Technical Report, CS Department, Boston University, December.

Cooper, B. and H. Garcia-Molina (2003a). "Ad hoc, Self-Supervising Peer-to-Peer Search Networks." Submitted for publication.

Cooper, B. and H. Garcia-Molina (2003b). "Studying search networks with SIL." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Costa, P., M. Migliavacca, G. P. Picco and G. Cugola (2003). "Introducing reliability in content-based publish-subscribe through epidemic algorithms." Proceedings of the 2nd international workshop on Distributed event-based systems, San Diego, California: 1-8.

Costa, P., M. Migliavacca, G. P. Picco and G. Cugola (2004). "Epidemic Algorithms for Reliable Content-Based Publish-Subscribe: An Evaluation." The 24th International Conference on Distributed Computing Systems (ICDCS-2004), Mar 23-26, Tokyo University of Technology, Hachioji, Tokyo, Japan.

Cox, L. and B. Noble (2003). "Samsara: honor among thieves in peer-to-peer storage." Proceedings of the nineteenth ACM symposium on Operating System Principles, Bolton Landing, NY, USA: 120-132.

Cox, R., F. Dabek, F. Kaashoek, J. Li and R. Morris (2004). "Practical,distributed network coordinates." ACM SIGCOMM Computer Communication Review **34**(1): 113-118.

Cox, R., A. Muthitacharoen and R. Morris (2002). "Serving DNS using a Peer-to-Peer Lookup Service." First International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, March.

Crespo, A. and H. Garcia-Molina (2002). "Routing Indices for Peer-to-Peer Systems." Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS, Vienna, Austria, July 2002).

Cuenca-Acuna, F., C. Peery, R. Martin and T. Nguyen (2002). "PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities." Proceedings of the 12th international symposium on High Performance Distributed Computing (HPDC) **June**.

Dabek, F., R. Cox, F. Kaashoek and R. Morris (2004). "Vivaldi: A Decentralized Network Coordinate System." SIGCOMM'04, Portland, Oregon, USA, Aug 30-Sept 3.

Dabek, F., M. F. Kaashoek, D. Karger, R. Morris and I. Stoica (2001). "Wide-area cooperative storage with CFS." Proceedings of the 18th ACM symposium on Operating System Principles: 202-215.

Dabek, F., J. Li, E. Sit, J. Robertson, F. Kaashoek and R. Morris (2004). "Designing a DHT for low latency and high throughput." Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, California, March 29-31: 85-98.

Dabek, F., B. Zhao, P. Druschel, J. Kubiatowicz and I. Stoica (2003). "Towards a common API for structured P2P overlays." Proceedings of IPTPS, Berkeley, CA, USA, February.

Damiani, E., D. C. di Vimercati, S. Paraboschi, P. Samarati and F. Violante (2002). "A reputation-based approach for choosing reliable resources in peer to peer networks." Proceedings of the 9th conference on computer and communications security: 207-216.

Daswani, N. and H. Garcia-Molina (2002). "Query-flood DoS attacks in gnutella." Proceedings of the 9th ACM Conference on Computer and Communications Security: 181-192.

Daswani, N., H. Garcia-Molina and B. Yang (2003). "Open Problems in Data-sharing Peer-to-peer Systems." The 9th International Conference on Database Theory (ICDT 2003), Siena, Italy, 8-10 January.

Daswani, S. and A. Fisk (2002). "Gnutella UDP Extension for Scalable Searches (GUESS) v0.1." http://www.limewire.org/fisheye/viewrep/~raw,r=1.2/limecvs/core/guess_01.html.

Datta, A., S. Girdzijauskas and K. Aberer (2004). "On de Bruijn routing in distributed hash tables: there and back again." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

de Bruijn, N. D. (1946). "A combinatorial problem." Koninklijke Netherlands: Academe Van Wetenschappen **49**: 758-764.

Demers, A., D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart and D. Terry (1987). "Epidemic Algorithms for Replicated Data Management." Proceedings of the Sixth ACM Symposium on Principles of Distributed Computing: 1-12.

Demirbas, M. and H. Ferhatosmanoglu (2003). "Peer-to-peer spatial queries in sensor networks." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 32-39.

DePaoli, F. and L. Mariani (2004). "Dependability in Peer-to-Peer Systems." IEEE Internet Computing **8**(4): 54-61. July

Devine, R. (1993). "Design and Implementation of DDH: A Distributed Dynamic Hashing Algorithm." Proceedings of 4th International Conference on Foundations of Data Organizations and Algorithms.

"Direct Connect." http:www.neo-modus.com.

Eugster, P., R. Guerraoiu, S. Handurukande, P. Kouznetsov and A.-M. Kermarrec (2003). "Lightweight Probabilistic Broadcast." ACM Transactions on Computer Systems **21**(4): 341-374.

Eugster, P., R. Guerraoiu, A. Kermarrec and L. Massoulie (2004). "Epidemic information dissemination in distributed systems." IEEE Computer **37**(5): 60-67. May

Feamster, N. and H. Balakrishnan (2003). "Towards a logic for wide-area Internet routing." Proceeings of the ACM SIGCOMM workshop on Future Directions in Network Architecture, Karlsruhe, Germany, August 25-27: 289-300.

Feldman, M., K. Lai, I. Stoica and J. Chuang (2004). "Robust Incentive Techniques for Peer-to-Peer Networks." ACM E-Commerce Conference (EC'04), May.

Feldman, M., C. Papadimitriou, J. Chuang and I. Stoica (2004). "Free-Riding and Whitewashing in Peer-to-Peer Systems." 3rd Annual Workshop on Economics and Information Security (WEIS04), May.

Fessant, F., S. Handurukande, A.-M. Kermarrec and L. Massoulie (2004). "Clustering in Peer-to-Peer File Sharing Workloads." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Fiat, A. and J. Saia (2002). "Censorship resistant peer to peer content addressable networks." Proceedings of the 13th annual ACM-SIAM symposium on discrete algorithms: 94-103.

Fisk, A. (2003a). "Gnutella Dynamic Query Protocol." http://groups.yahoo.com/group/the_gdf/files/Proposals/Working%20Proposals/search/Dynamic%20Querying/ **v0.1**.

Fisk, A. (2003b). "Gnutella Ultrapeer Query Routing." http://groups.yahoo.com/group/the_gdf/files/Proposals/Working%20Proposals/search/Ultrapeer%20QRP/ **v0.1**. May 20

Foster, I. and A. Iamnitchi (2003). "On Death, Taxes and the Convergence of Peer-to-Peer and Grid Computing." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Fraigniaud, P. and P. Gauron (2003). "The content-addressable networks D2B." Technical Report 1349, Laboratoire de Recherche en Informatique, Universit´e de Paris Sud, January.

Freedman, M., E. Freudenthal and D. Mazieres (2004). "Democratizing Content Publication with Coral." Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, California, March 29-31: 239-252.

Freedman, M. and D. Mazieres (2003). "Sloppy Hashing and Self-Organizing Clusters." Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), February 2003.

Freedman, M. and R. Morris (2002). "Tarzan: a peer-to-peer anonymizing network layer." Proceedings of the 9th ACM Conference on Computer and Communications Security: 193-206.

Furnas, G. W., S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter and K. E. Lochbaum (1988). "Information retrieval using a singular value decomposition model of latent semantic structure." Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval: 465-480.

Ganesan, P., M. Bawa and H. Garcia-Molina (2004). "Online Balancing of Range-Partitioned Data with Applications to Peer-to-Peer Systems." Proceedings of the 30th International Conference on Very Large Data Bases (VLDB 2004), Toronto, Canada. 29 August - 3 September

Ganesan, P., G. Krishna and H. Garcia-Molina (2004). "Canon in G Major: Designing DHTs with Hierarchical Structure." Proceedings of the International Conference on Distributed Computing Systems (ICDCS 2004).

Ganesh, A., A.-M. Kermarrec and L. Massoulie (2003). "Peer-to-peer membership management for gossip-based protocols."

Gao, J. and P. Steenkiste (2004). "Design and Evaluation of a Distributed Scalable Content Delivery System." IEEE Journal on Selected Areas in Communications **22**(1): 54-66.

Gedik, B. and L. Liu (2003a). "PeerCQ: A Decentralized and Self-Configuring Peer-to-Peer Information Monitoring System." Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS2003), Providence, Rhode Island, USA, May 19-22.

Gedik, B. and L. Liu (2003b). "Reliable peer-to-peer information monitoring through replication." Proceedings of the 22nd International Symposium on Reliable Distributed Systems, 6-8 Oct: 56-65.

Geels, D. and J. Kubiatowicz (2003). "Replica management should be a game." Proceedings of the SIGOPS European Workshop 2002, September 2002.

Ghemawat, S., H. Gobioff and S.-T. Leung (2003). "The Google file system." Proceedings of the 19th ACM symposium on operating systems principles, Bolton Landing, New York, USA: 29-43.

Gil, T., F. Kaashoek, J. Li, R. Morris and J. Stribling (2003). "p2psim, a simulator for peer-to-peer protocols." http://www.pdos.lcs.mit.edu/p2psim/.

Gilbert, S., N. Lynch and A. Shvartsman (2004). "RAMBO II: Rapidly Reconfigurable Atomic Memory for Dynamic Networks." Technical Report, MIT-CSAIL-TR-890, CSAIL, Massachusetts Institute Technology, Cambridge,MA.

Gnawali, O. (2002). "A Keyword Set Search System for Peer-to-Peer Networks." Master's Thesis, Massachusetts Institute of Technology.

Gold, R. and D. Tidhar (2001). Towards a content-based aggregation network. Proceedings of the First International Conference on Peer to Peer Compuuting, IEEE.

Gopalakrishnan, V., B. Silaghi, B. Bhattacharjee and P. Keleher (2004a). "Adaptive Replication in Peer-to-Peer Systems." Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), March 23-26, Tokyo, Japan.

Gopalakrishnan, V., B. Silaghi, B. Bhattacharjee and P. Keleher (2004b). "Adaptive Replications in Peer-to-Peer Systems." The 24th International Conference on Distributed Computing Systems (ICDCS-2004), Mar 23-26, Tokyo University of Technology, Hachioji, Tokyo, Japan.

Gray, J. (1981). "The transaction concept: Virtues and limitations." Proceedings of VLDB, Cannes, France, September.

Gribble, S., E. Brewer, J. M. Hellerstein and D. Culler (2000). "Scalable, Distributed Data Structures for Internet Service Construction." Proceedings of the 4th Symposium on Operating Systems Design and Implementation (OSDI 2000), San Diego, California, USA, October.

Gribble, S., A. Halevy, Z. Ives, M. Rodrig and D. Suciu (2001). "What can databases do for P2P?" Proceedings of WebDB2001.

Gummadi, K., R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker and I. Stoica (2003). "The impact of DHT routing geometry on resilience and proximity." Proceedings of the 2003 conference on Applications, Technologies, Architectures and Protocols for Computer Communications: 381-394.

Gummadi, K., S. Saroui, S. Gribble and D. King (2002). "Estimating latency between Arbitrary Internet End Hosts." Proceedings of SIGCOMM IMW 2002, Marsille, France, November.

Gupta, A., D. Agrawal and A. E. Abbadi (2003). "Approximate range selection queries in peer-to-peer systems." Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003), Asilomar, CA, USA.

Gupta, A., B. Liskov and R. Rodrigues (2003). "One Hop Lookups for Peer-to-Peer Overlays." 9th Workshop on Hot Topics in Operating Systems (HotOS), Lihue, Hawaii, 18-21 May.

Gupta, A., B. Liskov and R. Rodrigues (2004). "Efficient Routing for Peer-to-Peer Overlays." First Symposium on Networked Systems Design and Implementation (NSDI), San Francisco, CA, March.

Gupta, I. (2004). "On the design of distributed protocols from differential equations." Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), St. John's, Newfoundland, Canada, July 25-28: 216-225.

Gupta, I., K. Birman, P. Linga, A. Demers and R. Van Renesse (2003). "Kelips: Building an Efficient and Stable P2P DHT through increased memory and background overhead." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February. Feb 20-21

Gupta, I., K. P. Birman and v. Renesse (2001). "Fighting Fire with Fire: Using randomized Gossip to combat stochastic scalability limits." Cornell University Dept of Computer Science Technical Report, March.

Halepovic, E. and R. Deters (2002). "Building a P2P forum system with JXTA." Proceedings of the Second IEEE International Conference on Peer to Peer Computing (P2P'02), September 5-7, Linkoping, Sweden.

Halevy, A. Y., Z. Ives, P. Mork and I. Tatarinov (2003a). "Piazza: Data management infrastructure for semantic web applications." Proceedings of the Twelfth International conference on World Wide Web , Budapest, Hungary (WWW2003).

Halevy, A. Y., Z. Ives, D. Suciu and I. Tatarinov (2003b). "Schema mediation in peer data management systems." Proceedings of the 19th International Conference on Data Engineering (ICDE), Bangalore, India, March 2003.

Harren, J., J. Hellerstein, R. Huebsch, B. Loo, S. Shenker and I. Stoica (2002). "Complex queries in DHT-based peer-to-peer networks." Proceedings of the First International Workshop on Peer to Peer Systems (IPTPS 2002), March, Cambridge, MA.

Harvey, N., J. Dunagan, M. B. Jones, S. Saroiu, M. Theimer and A. Wolman (2002). "SkipNet: A Scalable overlay Network with Practical Locality Properties." Microsoft Research Technical Report MSR-TR-2002-92.

Harvey, N., M. B. Jones, S. Saroiu, M. Theimer and A. Wolman (2003a). "SkipNet: A Scalable Overlay Network with Practical Locality Properties." Proceedings of the Fourth USENIX Symposium on Internet Technologies and Systems (USITS '03), Seattle, WA, March.

Harvey, N., M. B. Jones, M. Theimer and A. Wolman (2003b). "Efficient Recovery from Organization Disconnects in SkipNet." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Hazel, S. and B. Wiley (2002). "Achord: A Variant of the Chord Lookup Service for Use in Censorship Resistant Peer-to-Peer Publishing Systems." Proceedings of the Second International Conference on Peer to Peer Computing, March, MIT Faculty Club, Cambridge, MA, USA.

Hefeeda, M., A. Habib, B. Botev, D. Xu and B. Bhargava (2003). "PROMISE: peer-to-peer media streaming using CollectCast." Proceedings of the eleventh ACM international conference on multimedia, Berkeley, CA, USA: 45-54.

Hellerstein, J. and W. Wang (2004). "Optimization of In-Network Data Reduction." Proceedings of the First Workshop on Data Management for Sensor Networks (DMSN 2004), Toronto, Canada, August 30th.

Hellerstein, J. M. (2003). "Toward network data independence." ACM SIGMOD Record **32**(3): 34-40. Sept

Hildrum, K., R. Krauthgamer and J. Kubiatowicz (2004). "Object Location in Realistic Networks." Proceedings of the Sixteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA 2004), June: 25-35.

Hildrum, K., J. Kubiatowicz, S. Rao and B. Zhao (2002). "Distributed object location in a dynamic network." Proceedings of the 14th annual ACM symposium on parallel algorithms and architectures: 41-52.

Hildrum, K., J. D. Kubiatowicz, S. Rao and B. Y. Zhao (2003). "Distributed Object Location in a Dynamic Network." To appear in Theory of Computing Systems, November 2002.

Hodes, T., S. Czerwinski, B. Zhao, A. Joseph and R. Katz (2002). "An architecture for secure wide-area service discovery." Wireless Networks **8**(2/3): 213-230.

Honicky, R. and E. Miller (2003). "A fast algorithm for online placement and reorganization of replicated data." Proceedings of the 17th International Parallel and Distributed Processing Symposium, Nice, France, April.

Hoschek, W. (2002). "Peer-to-Peer Grid Databases for Web Service Discovery." Concurrency - Practice and Experience: 1-7.

Hsiao, H. and C.-T. King (2003). "A tree model for structure peer-to-peer protocols." Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid: 336-343.

Hsieh, H.-Y. and R. Sivakumar (2004). "On Transport Layer Support for Peer-to-Peer Networks." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Huebsch, R., J. M. Hellerstein, N. Lanham, B. T. Loo, S. Shenker and I. Stoica (2003). "Querying the Internet with PIER." Proceedings of the 29th International Conference on Very Large Databases (VLDB '03), Berlin, Germany, September, 2003.

Iamnitchi, A. (2003). "Resource Discovery in Large Resource-Sharing Environments." Doctoral Dissertation, Department of Computer Science, University of Chicago.

Iamnitchi, A., I. Foster and D. Nurmi (2002). A peer-to-peer approach to resource discovery in grid environments. IEEE High Performance Distributed Computing, Edinboourgh, UK.

Imase, M. and M. Itoh (1981). "Design to minimized diameter on build-block network." IEEE Transactions on Computers **C-30**(6): 439-442. June

Iyer, S., A. Rowstron and P. Druschel (2002). "Squirrel: a decentralized peer-to-peer web cache." Proceedings of the 21st annual symposium on principles of distributed computing: 213-222.
Jarke, M. and J. Koch (1984). "Query Optimization in Database Systems." ACM Computing Surveys **16**(2): 111-152. June
Jelasity, M., W. Kowalczyk and M. van Steen (2004). "An Approach to Aggregation in Large and Fully Distributed Peer-to-Peer Overlay Networks." Proceedings of the 12th Euromicro Conference on Parallel, Distributted and Network based Processing (PDP 2004), A Coruna, Spain, February.
Joseph, S. (2002). "NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks." Proceedings of the International Workshop on Peer-to-Peer Computing, Pisa, Italy, May.
Joseph, S. and T. Hoshiai (2003). "Decentralized Meta-Data Strategies: Effective Peer-to-Peer Search." IEICE Trans. Commun. **E86-B**(6 June): 1740-1753.
Josephson, W., E. Sirer and F. Schneider (2004). "Peer-to-Peer Authentication with a Distributed Single Sign-On Service." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
Jovanovic, M. (2001). "Modeling Large-scale Peer-to-Peer Networks and a Case Study of Gnutella." Master's Thesis, Department of Electrical and Computer Engineering and Computer Science, College of Engineering, University of Cincinnati.
Junginger, M. and Y. Lee (2004). "A self-organizing publish/subscribe middleware for dynamic peer-to-peer networks." IEEE Network **18**(1): 38-43.
Kaashoek, F. and D. Karger (2003). "Koorde: A Simple Degree-optimal Hash Table." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.
Kalogaraki, V., D. Gunopulos and D. Zeinalipour-Yasti (2002). "XML schemas: integration and translation: A local search mechanism for peer to peer networks." Proceedings of the 11th ACM international conference on Information and Knowledge management: 300-307.
Kan, G. and Y. Faybishenko (2001). "Introduction to Gnougat." First International Conference on Peer-to-Peer Computing 2001: 4-12.
Kangasharju, J., K. Ross, D. Turner, J. Syrjala and D.-S. Digeon (2002). "Peer-to-peer email." Submitted for publication, November.
Karger, D., E. Lehman, T. Leighton, R. Panigraphy, M. Levin and D. Lewin (1997). "Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web." ACM Symposium on Theory of Computing. May
Karger, D. and M. Ruhl (2002). "Finding nearest neighbours in growth-restricted metrics." Proceedings of the 34th annual ACM symposium on Theory of computing: 741-750.
Karger, D. and M. Ruhl (2004a). "Diminished Chord: A Protocol for Heterogeneous SubGroup Formation in Peer-to-Peer Networks." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
Karger, D. and M. Ruhl (2004b). "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
Karger, D. and M. Ruhl (2004c). "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems." Proceedings of the Sixteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA 2004), Barcelona, Spain, June 27-30.
Karp, B., S. Ratnasamy, S. Rhea and S. Shenker (2004). "Spurring Adoption of DHTs with OpenHash, a Public DHT Service." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
"KaZaa Media Desktop, Sharman Networks Ltd." (2001). www.kazaa.com.
Keleher, P., S. Bhattacharjee and B. Silaghi (2002). "Are Virtualized Overlay Networks Too Much of a Good Thing?" First International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, March.
Klampanos, I. and J. Jose (2004). "An architecture for information retrieval over semi-collaborated peer-to-peer networks." Proceedings of the 2004 ACM symposium on applied computing, Nicosia, Cyprus: 1078-1083.
Klingberg, T. and R. Manfredi (2002). "Gnutella 0.6." June
Koloniari, G., Y. Petrakis and E. Pitoura (2003). "Content-Based Overlay Networks for XML Peers Based on Multi-Level Bloom Filters." Proceedings of the First International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P), Berlin, Germany, Sept 7-8: 232-247.
Koloniari, G. and E. Pitoura (2003). "Bloom-Based Filters for Hierarchical Data." Proceedings of the 5th Workshop on Distributed Data and Structures (WDAS).
Koloniari, G. and E. Pitoura (2004). Content-based Routing of Path Queries in Peer-to-Peer Systems. Proceedings of the 9th International Conference on Extending DataBase Technology (EDBT), Heraklion-Crete, Greece, March 14-18.
Kossmann, D. (2000). "The state of the art in distributed query processing." ACM Computing Surveys **32**(4): 422-469. Dec
Krohn, M., M. Freedman and D. Mazieres (2004). "On-the-fly verification of rateless erasure codes for efficient content distribution." Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, May.
Kronfol, A. Z. (2002). "FASD: A fault-tolerant, Adaptive scalable distributed search engine." Master's Thesis, Princeton University, http://www.cs.princeton.edu/~akronfol/fasd/.
Kubiatowicz, J., D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells and B. Zhao (2000). "Oceanstore: An Architecture for global-scale persistent storage." Proceedings of the Ninth International Conference on Architecture Support for Programming Languages and Operating Systems (ASPLOS 2000), November: 190-201.
Laprie, J.-C. (1995). "Dependable Computing and Fault Tolerance: Concepts and Terminology." Twenty-Fifth International Symposium on Fault-Tolerant Computing, 1995, Highlights from Twenty-Five Years: 2-13.
Lee, K. Y., G. Liu and H. F. Jordan (2000). "Hierarchical networks for optical communications." Journal of Parallel and Distributed Computing **60**: 1-16.
Lei, S. and A. Grama (2004). "Extended Consistent Hashing: A Framework for Distributed Servers." Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS 2004), March 23-26, Tokyo, Japan.
Li, G. (2002). "Peer to Peer Networks in Action." IEEE Internet Computing **6**(1): 37-39.
Li, J., B. T. Loo, J. Hellerstein, F. Kaashoek, D. Karger and R. Morris (2003). "On the Feasibility of Peer-to-Peer Web Indexing and Search." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.
Li, J. and K. Sollins (2003). "Implementing Aggregation and Broadcast over Distributed Hash Tables (full report)." (November).
Li, J., J. Stribling, T. Gil, R. Morris and F. Kaashoek (2004). "Comparing the performance of distributed hash tables under churn." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
Li, X. and C. Plaxton (2002). "On name resolution in peer to peer networks." Proceedings of the second ACM international workshop on principles of mobile computing: 82-89.
Liben-Nowell, D., H. Balakrishnan and D. Karger (2002). Analysis of the evolution of peer-to-peer systems. Annual ACM Sumposium on Principles of Distributed Computing, Monterey, California, United States, ACM Press.
Lienhart, R., M. Holliman, Y.-K. Chen, I. Kozintsev and M. Yeung (2002). "Improving media services on P2P networks." IEEE Internet Computing **6**(1): 58-67. Jan/Feb
"Limewire Host Count." (2004). http://www.limewire.com/english/content/netsize.shtml.
Lin, S.-D., Q. Lian, M. Chen and Z. Zhang (2004). "A Practical Distributed Mutual Exclusion Protocol in Dynamic Peer-to-Peer Systems." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.
Litwin, W. (2004). "Scalable Distributed Data Structures and Peer-to-Peer Computing." Tutorial, P2P&DB2004.
Litwin, W. (2004a). ""Re: Chord & LH*", Email to Ion Stoica, 23rd March."

Litwin, W., M. Neimat and D. Schneider (1996). "LH* - a scalable, distributed data structure." ACM Transactions on Database Systems (TODS) **21**(4): 480-525.

Litwin, W. and M.-A. Neimat (1996). "k-RP*s: a scalable distributed data structure for high-performance multi-attributed access." Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems: 120-131. 18-20 December

Litwin, W., M.-A. Neimat and D. Schneider (1994). "RP*: A Family of Order-Preserving Scalable Distributed Data Structures." Proceedings of the 20th International Conference on Very Large Data Bases (VLDB).

Litwin, W., M.-A. Niemat and D. Schneider (1993). "LH* - Linear Hashing for Distributed Files." Proceedings of ACM International Conference on Management of Data (SIGMOD), May.

Litwin, W. and S. Sahri (2004). "Implementing SD-SQL Server: a Scalable Distributed Database System." CERIA Research Rerpot 2004-04-02, April.

Liu, Y., X. Liu, L. Xiao, L. Ni and X. Zhang (2004). "Location-aware topology matching in P2P systems." Proceedings of IEEE Infocomm, Hong Kong, Mar 7-11.

Lo, V., D. Zappala, D. Zhou, Y. Liu and S. Zhao (2004). "Cluster computing on the Fly: P2P Scheduling of Idle Cycles in the Internet." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Loguinov, D., A. Kumar and S. Ganesh (2003). "Graph-theoretic analysis of structured peer-to-peer systems: routing distances and fault resilience." Proceedings of the 2003 conference on Applications, Technologies, Architectures and Protocols for Computer Communications, Karlsruhe, Germany, August 25-29: 395-406.

Lomet, D. (1996). "Replicated indexes for distributed data." Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems: 108-119. 18-20 December

Loo, A. (2003). "The Future or Peer-to-Peer Computing." Communications of the ACM **46**(9): 56-61. Sept

Loo, B. T., J. M. Hellerstein, R. Huebsch, S. Shenker and I. Stoica (2004). "Enhancing P2P File-Sharing with Internet-Scale Query Processor." Proceings of the 30th International Conference on Very Large Data Bases (VLDB 2004), Toronto, Canada, 29 August - 3 September.

Loo, B. T., R. Huebsch, J. M. Hellerstein, T. Roscoe and I. Stoica (2004a). "Analyzing P2P Overlays with Recursive Queries." Technical Report, Computer Science Division at the University of California at Berkeley, CSD-04-1301, 14th January.

Loo, B. T., R. Huebsch, I. Stoica and J. M. Hellerstein (2004b). "The Case for a Hybrid P2P Search Infrastructure." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Loo, B. T., S. Krishnamurthy and O. Cooper (2004). "Distributed Web Crawling over DHTs." Technical Report, Computer Science Division at the University of California at Berkeley, CSD-04-1305, 9th February.

Loser, A., F. Naumann, W. Siberski, W. Nejdl and U. Thaden (2003). "Semantic overlay clusters within super-peer networks." Proceedings of the First International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P), Berlin, Germany, Sept 7-8: 33-47.

Lua, E.-K., J. Crowcroft and M. Pias (2004). "Highways: Proximity Clustering for Massively Scaleable Peer-to-Peer Network Routing." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

Lv, Q., P. Cao, E. Cohen, K. Li and S. Shenker (2002). "Search and replication in unstructured peer to peer networks." Proceedings of the 16th international conference on supercomputing: 84-95.

Lv, Q., S. Ratnasamy and S. Shenker (2002). "Can Heterogeneity Make Gnutella Scalable?" Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS2002).

Lynch, N., D. Malkhi and D. Ratajczak (2002). "Atomic Data Access in Distributed Hash Tables." Proceedings of the International Peer-to-Peer Symposium, March.

Lynch, N. and I. Stoica (2004). "MultiChord: A resilient namespace management algorithm." To appear as Technical Memo MIT-LCS-TR-936, CSAIL, Massachusetts Institute of Technology, Cambridge, MA.

Mache, J., M. Gilbert, J. Guchereau, J. Lesh, F. Ramli and M. Wilkinson (2002). "Request algorithms in Freenet-style peer-to-peer systems." Proceedings of the Second IEEE International Conference on Peer to Peer Computing (P2P'02), September 5-7, Linkoping, Sweden.

Mahajan, R., M. Castro and A. Rowstron (2003). "Controlling the cost of reliability in peer-to-peer overlays." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Malkhi, D., M. Naor and D. Ratajczak (2002). "Viceroy: a scalable and dynamic emulation of the butterfly." Proceedings of the 21st annual symposium on principles of distributed computing: 183-192.

Manber, U. (1999). Forward. Modern Information Retrieval. R. Baeza-Yates and B. Ribeiro-Neto. Reading, MA, Addison-Wesley**:** 5-8.

Manku, G. S. (2003). "Routing networks for distributed hash tables." Proceedings of the 22nd annual ACM Symposium on Principles of Distributed Computing, Boston, Massachusetts: 133-142.

Manku, G. S. (2004). "Balanced binary trees for ID management and load balance in distributed hash tables." Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), St. John's, Newfoundland, Canada, July 25-28: 197-205.

Manku, G. S., M. Bawa and P. Raghavan (2003). "Symphony: Distributed Hashing in a Small World." Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems, Seattle, WA, USA, March 26-28.

Mao, J.-W. (2003). "The Coloring and Routing Problems on de Bruijn Interconnection Networks." Doctoral Dissertation, National Sun Yat-sen University, July 18.

Marti, S., P. Ganesan and H. Garcia-Molina (2004). "DHT Routing using Social Links." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Massoulie, L., A. Kermarrec and A. Ganesh (2003). "Network awareness and failure resilience in self-organizing overlay networks." Proceedings of the 22nd International Symposium on Reliable Distributed Systems, 6-8 Oct: 47-55.

Maymounkov, P. and D. Mazieres (2002). "Kademlia: A peer-to-peer information system based on the XOR metric." Proceedings of the First International Workshop on Peer to Peer Systems (IPTPS 2002), March **kademia.scs.cs.nyu.edu**.

Maymounkov, P. and D. Mazieres (2003). "Rateless Codes and Big Downloads." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

McCanne, S. and S. Floyd "The LBNL/UCB Network Simulator, Lawrence Berkeley Laboratory, University of California, Berkeley."

McGuinness, D. L. and F. van Harmelen (2004). "OWL web ontology language overview, W3C Recommendation 10 February." http://www.w3.org/TR/owl-features/.

Miller, L., A. Seaborne and A. Reggiori (2002). "Three Implementations of SquishQL, a Simple RDF Query Language." First International Semantic Web Conference (ISWC2002), Sardinia, June.

Milo, T., S. Abiteboul, B. Amann, O. Benjelloun and F. D. Ngoc (2003). "Exchanging intensional XML data." Proceedings of the 2003 ACm SIGMOD international conference on management of data, San Diego, California: 289-300.

Milojicic, D., V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins and Z. Xu (2002). "Peer-to-Peer Computing." HP Technical Report, HPL-2002-57.

Mishchke, J. and B. Stiller (2004). "A methodology for the design of distributed search in P2P middleware." IEEE Network **18**(1): 30-37.

Mislove, A. and P. Druschel (2004). "Providing Administrative Control and Autonomy in Structured Peer-to-Peer Overlays." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Mislove, A., A. Post, C. Reis, P. Willmann, P. Druschel, D. Wallach, X. Bonnaire, P. Sens, J.-M. Busca and L. Arantes-Bezerra (2003). "POST: A Secure, Resilient, Cooperative Messaging System." 9th Workshop on Hot Topics in Operating Systems (HotOS), Lihue, Hawaii, May.

Mizrak, A., Y. Cheng, V. Kumar and S. Savage (2003). "Structured superpeers: Leveraging heterogeneity to provide constant-time lookup." IEEE Workshop on Internet Applications.

Mohan, A. and V. Kalogaraki (2002). "Speculative Routing and Update Propagation: A Kundali Centric Approach." IEEE International Conference on Communications (ICC'03), May.

Montresor, A., M. Jelasity and O. Babaoglu (2003). "Robust Aggregation Protocol for Large-Scale Overlay Networks." Technical Report UBLCS-2003-16, Department of Computer Science, University of Bologna, Italy, December.

Muhl, G. (2002). "Large-Scale Content-Based Publish/Subscribe Systems." Doctoral Dissertation, Darmstadt University of Technology, September.

Muller, W. and A. Henrich (2003). "Fast retrieval of high-dimensional feature vectors in P2P networks using compact peer data summaries." Proceedings of the 5th ACM SIGMM international workshop on Multimedia Information Retrieval: 79-86.

Muthitacharoen, A., R. Morris, T. Gil and B. Chen (2002a). "Ivy: a read/write peer-to-peer file system." ACM SIGOPS Operating Systems Review, Special issue on Decentralized storage systems: 31-44.

Muthitacharoen, A., R. Morris, T. Gil and B. Chen (2002b). "A read/write peer-to-peer file system." Proceedings of the 5th Symposium on Operating System Design and Implementation (OSDI 2002), Boston, MA, December.

Naor, M. and U. Wieder (2003a). "Novel architectures for P2P applications: the continuous-discrete approach." Proceedings of the fifteenth annual ACM Symposium on Parallel Algorithms and Architectures (SPAA 2003), San Diego, California, USA, June 7-9: 50-59. June

Naor, M. and U. Wieder (2003b). "A Simple Fault Tolerant Distributed Hash Table." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Naor, M. and U. Wieder (2004). "Know thy Neighbor's Neighbor: Better Routing for Skip-Graphs and Small Worlds." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

"Napster." (1999). http://www.napster.com.

Nejdl, W., S. Decker and W. Siberski (2003). "Edutella Project, RDF-based Metadata Infrastructure for P2P Applications." http://edutella.jxta.org/.

Nejdl, W., W. Siberski and M. Sintek (2003). "Design Issues and Challenges for RDF- and schema-based peer-to-peer systems." ACM SIGMOD Record **32**(3): 41-46.

Nejdl, W., B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer and T. Risch (2002). "Edutella: A P2P Networking Infrastructure Based on RDF." WWW2002, May 7-11, Honolulu, Hawaii, USA.

Ng, T. S. E. and H. Zhang (2002). "Predicting Internet Network Distance with Coordinates-Based Approaches." IEEE Infocom 2002, The 21st Annual Joint Conference of the IEEE Computer and Communication Societies, New York, 23-27 June.

Ng, W. S., B. C. Ooi, K. L. Tan and A. Zhou (2003). "PeerDB: A p2p-based system for distributed data sharing." Proceedings of the 19th International Conference on Data Engineering (ICDE), Bangalore, India, March 2003.

Ng, W. S., B. C. Ooi, K. L. Tan and A. Y. Zhou (2002). "PeerDB: A p2p-based system for distributed data sharing." Technical Report, University of Singapore.

Ngan, T.-W., D. Wallach and P. Druschel (2003). "Enforcing Fair Sharing of Peer-to-Peer Resources." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Nicolosi, A. and D. Mazieres (2004). "Secure Acknowledgment of Multicast Messages in Open Peer-to-Peer Networks." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

O'Donnel, C. and V. Vaikuntanathan (2004). "Information Leak in the Chord Lookup Protocol." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

On, G., J. Schmitt and R. Steinmetz (2003). "The effectiveness of realistic replication strategies on quality of availability for peer-to-peer systems." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 57-64.

Ooi, B., Y. Shu and K.-L. Tan (2003). "Relational data sharing in peer-based data management systems." ACM SIGMOD Record, Special topic section on peer to peer data managment: 59-64.

Ozsu, M. T. and P. Valduriez (1999). Principles of Distributed Database Systems, Prentice Hall.

Padmanabhan, V., H. Wang and P. Chou (2004). "Supporting Heterogeneity and Congestion Control in Peer-to-Peer Multicast Streaming." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Papadimos, V., D. Maier and K. Tufte (2003). "Distributed query processing and catalogs for Peer-to-Peer Systems." Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR 2003), Asilomar, CA, USA. 5-8 January

Papaioannou, T. and G. Stamoulis (2004). "Effective Use of Reputation in Peer-to-Peer Environments." Fourth International Workshop on Global and Peer-to-Peer Computing, Chicago, USA, 20-21 April 2004.

Pias, M., J. Crowcroft, S. Wilbur, T. Harris and S. Bhatti (2003). "Lighthouses for Scalable Distributed Location." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Pitoura, E., S. Abiteboul, D. Pfoser, G. Samaras and M. Vazirgiannis (2003). "DBGlobe: a service-oriented P2P system for global computing." ACM SIGMOD Record, Special topic section on peer to peer data managment **32**(3 September): 77-82.

Plank, J., S. Atchley, Y. Ding and M. Beck (2003). "Algorithms for High Performance, Wide-Area Distributed File Downloads." Parallel Processing Letters **13**(2): 207-223.

Plaxton, C., R. Rajaraman and A. Richa (1997). "Accessing nearby copies of replicated objects in a distributed environment." ACM Syposium on Parallel Algorithms and Architectures. June

Psaltoulis, D., I. Kostoulas, I. Gupta, K. Birman and A. Demers (2004). "Practical algorithms for size estimation in large and dynamic groups." Proceedings of the Twenty-Third Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), New Foundland, Canada, July 25-28.

Pugh, W. (1989). "Skip lists: a probabilistic alternative to balanced trees." Proceedings of the Workshop on Algorithms and Data Structures, pages 437-449.

Pugh, W. (1990). "Skip lists: a probabilistic alternative to balanced trees." Communications of the ACM **33**(6): 668-676. June

Ramabhadran, S., S. Ratnasamy, J. Hellerstein and S. Shenker (2004). "Brief announcement: prefix hash tree." Proceedings of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC 2004), St. John's, Newfoundland, Canada, July 25-28: 368-368.

Ramaswamy, L. and L. Liu (2003). "FreeRiding: A New Challenge for Peer-to-Peer File Sharing Systems." Proceedings of the 2003 Hawaii International Conference on System Sciences (P2P Track) HICSS2003.

Rao, A., K. Lakshminarayanan, S. Surana, R. Karp and I. Stoica (2003). "Load Balancing in Structured P2P Systems." Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), February 2003.

Ratnasamy, S. (2002). "A Scalable Content-Addressable Network." Doctoral Dissertation, University of California at Berkeley, Berkeley, California, USA.

Ratnasamy, S., P. Francis and M. Handley (2003). "Range queries in DHTs." Technical Report IRB-TR-03-009, Intel Research, July.

Ratnasamy, S., P. Francis, M. Handley, R. Karp and S. Shenker (2001). "A Scalable content-addressable network." ACM SIGCOMM 2001.

Ratnasamy, S., B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin and F. Yu (2003). "Data Centric Storage in Sensornets with GHT, a geographic hash table." Mobile Networks and Applications **8**(4): 427-442.

Ratnasamy, S., B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan and S. Shenker (2002). GHT: a geographic hash table for data-centric storage. Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (Mobicom), Atlanta, Georgia, USA, ACM Press.

Ratnasamy, S., S. Shenker and I. Stoica (2002). "Routing Algorithms for DHTs: Some Open Questions." Proceedings of the First International Workshop on Peer to Peer Systems (IPTPS 2002), March, Cambridge, MA.

Reddy, S. M., D. K. Pradhan and J. G. Kuhl (1980). "Direct graphs with minimal and maximal connectivity." Technical Report, School of Engineering, Oakland University. July

Reynolds, P. and A. Vahdat (2003). "Efficient Peer-to-Peer Keyword Searching." ACM/IFP/USENIX International Middleware Conference (Middleware 2003), Rio de Janeiro, Brazil, 16-20 June.

Rhea, S., D. Geels, T. Roscoe and J. Kubiatowicz (2003). "Handling Churn in a DHT." Report No. UCB/CSD-03-1299, University of California, Berkeley, California, USA, also Proceedings of the USENIX Annual Technical Conference, June 2004. December

Rhea, S., T. Roscoe and J. Kubiatowicz (2003). "Structured Peer-to-Peer Overlays Need Application-Driven Benchmarks." Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), February.

Rhea, S., C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon and J. Kubiatowicz (2001). "Maintenance-Free Global Data Storage." IEEE Internet Computing **5**(5): 40-49.

Ripeanu, M., A. Iamnitchi and P. Foster (2002). "Mapping the Gnutella network." IEEE Internet Computing **6**(1): 50-57. Jan/Feb

Robertson, S. E., S. Walker and M. Beaulieu (1999). "Okapi at TREC-7: automatic ad hoc, filtering, VLC and filtering tracks." Proceedings of the Seventh Text REtrieval Conference (TREC-7), pages 253-264. NIST Special Publication 500-242, July.

Rodrigues, R., B. Liskov and L. Shrira (2002). "The design of a robust peer-to-peer system." SIGOPS European Workshop.

Rohrs, C. (2002). "Query Routing for the Gnutella Networks." http://www.limewire.com/developer/query_routing/keyword%20routing.htm **Version 1.0**. May 16

Roussopoulos, M., M. Baker, D. Rosenthal, T. Guili, P. Maniatis and J. Mogul (2004). "2 P2P of Not 2 P2P?" The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Rowley, R. A. and B. Bose (1993). "Fault-tolerant ring embedding in de Bruijn networks." IEEE Transactions on Computers **42**(12): 1480-1486. December

Rowstron, A. and P. Druschel (2001a). "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems." IFIP/ACM Middleware 2001, Heidelberg, Germany, Nov.

Rowstron, A. and P. Druschel (2001b). "Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility." Proceedings ACM SOSP'01, Banff, Canada, October 2001: 188-201.

Ruhl, M. (2003). "Efficient Algorithms for New Computational Models." Doctoral Dissertation, Massachusetts Institute of Technology, September.

Salton, G., A. Wong and C. S. Yang (1975). "A vector space model for automatic indexing." Communications of the ACM **18**(11): 613-620. November

Sankaralingam, K., S. Sethumadhavan and J. Browne (2003). "Distributed Pagerank for P2P Systems." Proceedings of the 12th international symposium on High Performance Distributed Computing (HPDC).

Saroiu, S., K. Gummadi, R. Dunn, S. Gribble and H. Levy (2002a). "An analysis of Internet content delivery systems." ACM SIGOPS Operating Systems Review **36**: 315-327.

Saroiu, S., P. Gummadi and S. Gribble (2002b). A measurement study of peer-to-peer file sharing systems. Proceedings of Multimedia Computing and Networking 2002 (MMCN'02), January.

Sartiani, C., P. Manghi, G. Ghelli and G. Conforti (2004). "XPeer: A self-organizing XML P2P database system." Proceedings of the First International Workshop on Peer-to-Peer Computing and Databases, Heraklion, Crete, Greece, March 14.

Sasabe, M., N. Wakamiya, M. Murata and H. Miyahara (2003). "Scalable and continuous media streaming on peer-to-peer networks." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 92-99.

Schlosser, M., M. Sintek, S. Decker and W. Nejdl (2002). "HyperCuP - Hypercubes, Ontologies and P2P Networks." Springer Lecture Notes on Computer Science, Vol 2530 - Agents and Peer-to-Peer Systems.

Schlumberger, M. L. (1974). "De Bruijn communication networks." Doctoral Dissertation, Department of Computer Science, Stanford University, Palo Alto, CA.

Schmidt, C. and M. Parashar (2004). "Enabling flexible queries with guarantees in P2P systems." IEEE Internet Computing **8**(3): 19-26.

Schneidman, J. and D. Parkes (2003). "Rationality and Self-Interest in Peer to Peer Networks." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Scholl, D. (2001). "OpenNap Home Page." http://opennap.sourceforge.net/.

Selcuk, A., E. Uzun and M. R. Pariente (2004). "A Reputation-based Trust Management System for P2P Networks." Fourth International Workshop on Global and Peer-to-Peer Computing, Chicago, USA, 20-21 April 2004.

Sen, S. and J. Wang (2002). "Analyzing peer-to-peer traffic across large networks." Proceedings of the second ACM SIGCOMM workshop on Internet measurement: 137-150.

Sen, S. and J. Wang (2004). "Analyzing Peer-to-Peer Traffic Across Large Networks." IEEE/ACM Transactions on Networking **12**(2): 219-232.

Serjantov, A. (2002). "Anonymizing Censorship Resistant Systems." Proceedings of the Second International Conference on Peer to Peer Computing, March, MIT Faculty Club, Cambridge, MA, USA.

Shenker, S. (2003). "The Data-Centric Revolution in Networking." Keynote Speech, 29th International Conference on Very Large Data Bases, September 9-12, Berlin, Germany.

Shi, S.-M., Y. Guangwen, D. Wang, J. Yu, S. Qu and M. Chen (2004). "Making Peer-to-Peer Keyword Searching Feasible Using Multi-level Partitioning." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Sieka, B., A. Kshemkalyani and M. Singhal (2004). "On the Security of Polling Protocols in Peer-to-Peer Systems." Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 25-27 August.

Singh, A. and L. Liu (2003). "TrustMe: anonymous management of trust relationships in decentralized P2P systems." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept.

Singhal, A. (2001). "Modern Information Retrieval: A Brief Overview." IEEE Data Engineering Bulletin **24**(4): 35-43.

Singhal, A., J. Choi, D. Hindle, D. Lewis and F. Pereira (1999). "AT&T at TREC-7." Proceedings of the Seventh Text REtrieval Conference (TREC-7), pages 253-264. NIST Special Publication 500-242, July.

Singla, A. and C. Rohrs (2002). "Ultrapeers: Another Step Towards Gnutella Scalability,." http://groups.yahoo.com/group/the_gdf/files/Proposals/Working%20Proposals/Ultrapeer/ **Version 1.0, 26 November**.

Sit, E., F. Dabek and J. Robertson (2004). "UsenetDHT: A Low overhead Usenet Server." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Sit, E. and R. Morris (2002). "Security Considerations for Peer-to-Peer Distributed Hash Tables." First International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, March.

Sollins, K. (2003). "Designing for Scale and Differentiation." Proceedings of the ACM SIGCOMM workshop on Future Directions in network architecture, Karlsruhe, Germany, August 25-27.

Stavrou, A., D. Rubenstein and S. Sahu (2004). "A Lightwight, Robust P2P System to Handle Flash Crowds." IEEE Journal on Selected Areas in Communications **22**(1): 6-17. Jan

Stoica, I., D. Adkins, S. Zhuang, S. Shenker and S. Surana (2002). "Internet indirection infrastructure." Proceedings of the 2002 conference on applications, technologies, architectures and protocols for computer communications: 73-86.

Stoica, I., R. Morris, D. Liben-Nowell, D. Karger, M. Kaashoek, F. Dabek and H. Balakrishnan (2001). Chord: A scalable peer-to-peer lookup service for internet applications. Proceedings of ACM SIGCOMM 2001, San Diego, CA, USA.

Stoica, I., R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek and H. Balakrishnan (2003). "Chord: a scalable peer-to-peer lookup protocol for Internet applications." IEEE/ACM Transactions on Networking **11**(1): 17-32.

Stonebraker, M., P. Aoki, W. Litwin, A. Pfeffer, A. Sah, J. Sidell, C. Staelin and A. Yu (1996). "Mariposa: a wide-area distributed database system." THE VLDB Journal - The International Journal of Very Large Data Bases(5): 48-63.

Sunaga, H., M. Takemoto and T. Iwata (2002). "Advanced peer to peer network platform for various services - SIONet(Semantic Information Oriented Network)." Proceedings of the Second International Conference on Peer to Peer Computing: 169-170.

Surridge, M. and C. Upstill (2003). "Grid security: lessons for peer-to-peer systems." Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 1-3 Sept: 2-6.

Tang, C. and S. Dwarkadas (2004). "Hybrid Global-Local Indexing for Efficient Peer-to-Peer Information Retrieval." Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, California, March 29-31: 211-224.

Tang, C., S. Dwarkadas and Z. Xu (2004). "On Scaling Latent Semantic Indexing for Large Peer-to-Peer Systems." The 27th Annual International ACM SIGIR Conference (SIGIR'04), ACM Special Interest Group on Information Retrieval), Sheffield, UK, July.

Tang, C., Z. Xu and S. Dwarkadas (2003). "Peer-to-peer information retrieval using self-organizing semantic overlay networks." Proceedings of the 2003 conference on Applications, Technologies, Architectures and Protocols for Computer Communications, Karlsruhe, Germany, August 25-29: 175-186.

Tang, C., Z. Xu and M. Mahalingam (2002a). "PeerSearch: Efficient information retrieval in peer-to-peer networks." Proceedings of HotNets-I, ACM SIGCOMM.

Tang, C., Z. Xu and M. Mahalingam (2002b). "pSearch: Information Retrieval in Structured Overlays." First Workshop on Hot Topics in Networks, Princeton, New Jersey, Oct 28-29. Also Computer Communication Review, Volume 33, Number 1, January 2003.

Tanin, E., A. Harwood and H. Samet (2004). "Indexing distributed complex data for complex queries." Proceedings of the National Conference on Digital Government Research, Seattle,WA: 81-90.

Tatarinov, I. and A. Halevy (2004). "Efficient query reformulation in peer data management systems." Proceedings of the 2004 ACM SIGMOD International Conference on the management of data, Paris, France: 539-550.

Tatarinov, I., P. Mork, Z. Ives, J. Madhavan, A. Halevy, D. Suciu, N. Dalvi, X. Dong, Y. Kadiyska and G. Miklau (2003). "The Piazza peer data management project." ACM SIGMOD Record **32**(3): 47-52.

Tempich, C., S. Staab and A. Wranik (2004). "Remindin': semantic query routing in peer-to-peer networks." Proceedings of the 13th conference on World Wide Web, New York, NY, USA, May 17-20: 640-649.

Terpstra, W., S. Behnel, L. Fiege, J. Kangasharju and A. Buchmann (2004). "Bit Zipper Rendezvous, Optimal Data Placement for General P2P Queries." Proceedings of the First International Workshop on Peer-to-Peer Computing and Databases, Heraklion, Crete, Greece, March 14.

Thadani, S. (2001a). "Meta Data searches on the Gnutella Network (addendum)." http://www.limewire.com/developer/MetaProposal2.htm.

Thadani, S. (2001b). "Meta Information Searches on the Gnutella Networks." http://www.limewire.com/developer/metainfo_searches.html.

Tran, D., K. Hua and T. Do (2004). "A Peer-to-Peer Architecture for Media Streaming." IEEE Journal on Selected Areas in Communications **22**(1): 121-133.

Triantafillou, P. and T. Pitoura (2003). "Towards a Unifying Framework for Complex Query Processing over Structured Peer-to-Peer Data Networks." Proceedings of the First International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P), Berlin, Germany, Sept 7-8: 169-183.

Tsangou, M., S. Ndiaye, M. Seck and W. Litwin (2003). "Range Queries to Scalable Distributed Data Structure RP*." Proceedings of the Fifth Workshop on Distributed Data and Structures, WDAS 2003, Thessaloniki, June.

van Renesse, R. (2003). "The Importance of Aggregation." Springer-Verlag Lecture Notes in Computer Science volume 2584 "Future Directions in Distributed Computing". A. Schiper, A. A. Shvartsman, H. Weatherspoon, and B. Y. Zhao, editors. Springer-Verlag, Heidelberg, April.

van Renesse, R., K. P. Birman and W. Vogels (2003). "Astrolabe: A robust and scalable technology for distribute system monitoring, management and data mining." ACM Transactions on Computer Systems **21**(2): 164-206.

van Renesse, R. and A. Bozdog (2004). "Willow: DHT, Aggregation and Publish/Subscribe in One Protocol." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Vogels, W., R. v. Renesse and K. Birman (2003). "The power of epidemics: robust communication for large-scale distributed systems." ACM SIGCOMM Computer Communication Review **33**(1): 131-135. January 2003

Voulgaris, S. and M. van Steen (2003). "An Epidemic Protocol for Managing Routing Tables in very large Peer to Peer Networks." Proceedings of the 14th IFIP/IEEE Workshop on Distributed Systems: Operations and Management (DSOM 2003), Heidelberg, Germany. October 2003

Walfish, M., H. Balakrishnan and S. Shenker (2004). "Untangling the Web from DNS." Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, California, March 29-31: 225-238.

Wang, X., Y. Zhang, X. Li and D. Loguinov (2004). "On Zone-Balancing of Peer-to-Peer Networks: Analysis of Random Node Join." ACM SIGMETRICS, June.

Wawrzoniak, M., L. Peterson and T. Roscoe (2004). "Sophia: an Information Plane for networked systems." ACM SIGCOMM Computer Communication Review **34**(1): 15-20.

Weatherspoon, H. and J. Kubiatowicz (2002a). "Efficient Heartbeats and Repair of Softstate in Decentralized Object Location and Routing Systems." Proceedings of the SIGOPS European Workshop 2002, September 2002.

Weatherspoon, H. and J. Kubiatowicz (2002b). "Erasure Coding vs Replication: A Quantative Comparison." Proceedings of the First International Workshop on Peer to Peer Systems (IPTPS 2002), March.

Weatherspoon, H., T. Moscovitz and J. Kubiatowicz (2002). "Introspective Failure Analysis: Avoiding Correlated Failures." Proceedings of the International Workshop on Reliable Peer-to-Peer Distributed Systems **October**.

Yalagandula, P. and M. Dahlin (2004). "A Scalable Distributed Information Management System." SIGCOMM'04, Portland, Oregon, USA, Aug 30-Sept 3.

Yang, B. and H. Garcia-Molina (2001). "Comparing Hybrid Peer-to-Peer Systems (extended)." 27th International Conference on Very Large Data Bases, Rome, Italy, 11-14 September.

Yang, B. and H. Garcia-Molina (2002a). "Efficient Search in Peer-to-Peer Networks." Proceedings of the 22nd International Conference on Distributed Computing Systems, 2-5 July, Vienna, Austria.

Yang, B. and H. Garcia-Molina (2002b). "Improving search in peer-to-peer networks." Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems: 5-14.

Yang, B. and H. Garcia-Molina (2002c). "Improving Search in Peer-to-Peer Systems." Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS, Vienna, Austria, July 2002).

Yang, B. and H. Garcia-Molina (2003). "Designing a Super-peer Network." Proceedings of the 19th International Conference on Data Engineering (ICDE), Bangalore, India, March 2003.

Yang, B., P. Vinograd and H. Garcia-Molina (2004). "Evaluating GUESS and Non-Forwarding Peer-to-Peer Search." The 24th International Conference on Distributed Computing Systems (ICDCS-2004), Mar 23-26, Tokyo University of Technology, Hachioji, Tokyo, Japan.

Yeager, B. (2003). "Proposed Research Tracks." Email to the Internet Research Task Force (IRTF) P2P Research Group, 10th Nov.

Yeager, B. and B. Bhattacharjee (2003). "Peer-to-Peer Research Group Charter." http://www.irtf.org/charters/p2prg.html. June

Zhang, R. and C. Hu (2003). "Borg: a hybrid protocol for scalable application-level multicast in peer-to-peer networks." Proceedings of the 13th international workshop on network and operating systems for digital audio and video, Monterey, CA, USA: 172-179.

Zhang, X., Q. Zhang, G. Song and W. Zhu (2004). "A Construction of Locality-Aware Overlay Network: mOverlay and its Performance." IEEE Journal on Selected Areas in Communications **22**(1): 18-28. Jan

Zhang, Z., S.-M. Shi and J. Zhu (2003). "SOMO: Self-organized metadata overlay for resource management in P2P DHTs." Second International Workshop on Peer-to-Peer Systems (IPTPS 03), Berkeley, CA, USA, 20-21 February.

Zhao, B., Y. Duan, L. Huang, A. Joseph and J. Kubiatowicz (2002). "Brocade: Landmark Routing on Overlay Networks." First International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, March.

Zhao, B., L. Huang, J. Stribling, S. Rhea, A. Joseph and J. Kubiatowicz (2004). "Tapestry: A Resilient Global-Scale overlay for Service Deployment." IEEE Journal on Selected Areas in Communications **22**(1): 41-53.

Zhao, B., J. Kubiatowicz and A. Joseph (2001). "Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing." Report No. UCB/CSD-01-1141, University of California, Berkeley, California, USA.

Zhou, L. and R. van Renesse (2004). "P6P: A Peer-to-Peer Approach to Internet Infrastructure." The 3rd International Workshop on Peer-to-Peer Systems, San Diego, CA, USA, February 26-27.

Zhuang, S., B. Zhao, A. Joseph, R. Katz and J. Kubiatowicz (2001). "Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination." Proceedings of the 11th ACM international workshop on network and operating systems support for digital audio and video **Jan**: 11-20.