Chapter 2

Background

In this chapter we introduce the concepts underlying epidemic algorithms, define our programing model and assumptions and describe the conventions used throughout the rest of this dissertation.

2.1 Model

In general, distributed systems are classified according to the interprocess communication mechanism, the timing model and the fault model (Lynch 1996).

The fault model specifies the type of faults in the system and their detectability. For simplicity, we assume a failstop model: processes may fail by prematurely halting their execution and such failures are eventually detected. This means that if a process fails and later recovers and rejoins the system, for instance as a consequence of churn, it does so as a new process. Moreover, in the algorithms we propose, the precision of failure detection only impacts performance not correctness. Processes that do not fail are said to be correct. Interprocess communication is the mechanism used by processes to exchange operations and data, and can be implemented by shared memory, point-to-point or broadcast of messages, or remote procedure calls. Due to its simplicity and wide availability, we focus only on point-to-point communication over an IP network using a transport protocol, such as TCP or UDP. By assumption, each process can be uniquely identified and reached by its IP address and port, i.e., we do not consider processes behind firewalls or NATs. Note that this limitation can be overcome by the use of several techniques, using epidemic algorithms readily available (Kermarrec et al.

2009; Leitão et al. 2010). The virtual link between any two pair of processes is called a channel. We assume channels to be fair-lossy, i.e., losses might occur but if a correct process sends a message infinitely often to another correct process, the latter will receive that message infinitely many times. Moreover, we further assume that channels do not corrupt, duplicate or create spurious messages. In practice, this can be implemented by a reliable transport protocol, such as TCP, or by the application by using message retransmissions or the stubborn channel abstraction (Guerraoui et al. 1998).

The timing model is concerned with the assumptions done regarding relative process speeds and communication channels timeliness. In one extreme, we have synchronous systems which have a well-known upper bound on the time it takes for processes to execute operations and on the time taken since a message is sent until it is received. On the other hand, in asynchronous systems there is no such upper-bound and both processes execution speed and message transmission time can take indefinitely long. Despite being harder to reason about, asynchronous systems are more generic and therefore we focus only on asynchronous systems in this dissertation. Moreover, unless otherwise stated, there is no assumption in the availability of a global clock.

2.2 Overlay Networks

In order for the system to function properly, each process needs to know the identifier of other processes with which it can communicate with. This set of process identifiers is known as the view and the size of the view is known as the degree. When a process p has a process q in its view, q is said to be a neighbor of p and the set of all processes in p's view is called the neighborhood of p. The set of all views establishes a *who knows who* relationship and is known as an overlay network - a logical network imposed on top of the physical infrastructure. When analyzing the global properties of an overlay network, it is often useful to model it as a graph where processes are vertices connected by the links or edges induced by the views. This graph should have some key properties that any algorithm should strive to obtain and preserve. These properties are (Jelasity et al. 2007):

- **Connectivity:** indicates process reachability and is obtained when any process is able to reach every other process in the system in a finite number

of hops. Failure to ensure connectivity, known as a partition, severely impairs the usefulness of algorithms as not all processes are able to receive the desired application data.

- **Average path length:** measures the average number of hops separating any two processes in the system. It is related to the overlay diameter which is given by the greatest path length between any two processes. The average path length should be as small as possible as it imposes a lower bound on the time needed to disseminate data among all processes.

- **Clustering coefficient:** measures the closeness of neighbor relations among processes. It is defined as the number of links among the neighbors of a given process divided by the number of all possible links among those processes. This property affects redundancy because the number of duplicates received directly increases with it, and also robustness because graphs with high clustering coefficients are more prone to partitions.

- **Degree distribution:** is the distribution of the number of neighbors of each process - the degree or size of the view - and measures processes' reachability and their contribution to the connectivity.

In the following, we present the two main approaches to build overlay networks: structured and unstructured.

2.2.1 Structured Overlays

In the class of structured overlay networks, the neighboring relations among processes are established judiciously according to some criteria, such as latency or distance. Due to the tight control over link establishment, structured overlay networks are efficient in routing data and/or requests to the appropriate process, as the location of those processes could be calculated in a deterministic fashion. Thus, structured overlay networks are popular to store and retrieve arbitrary data and build distributed hash tables (DHT) (Plaxton et al. 1997). DHT algorithms define a topology by assigning identifiers to each process, and a function that determines the distance, in number of hops, between any two identifiers in the space. Nonetheless, the inherent overlay structure can also be used to provide data dissemination primitives to applications (Jannotti et al. 2000; Ratnasamy

et al. 2001; Zhuang et al. 2001; Castro et al. 2002). Structured overlay networks are typically built as spanning trees (Gallager et al. 1983) or more complex structures, such as hypercubes (Rowstron and Druschel 2001; Zhao et al. 2001; Stoica et al. 2003) and Cartesian hyperspaces (Ratnasamy et al. 2001).

Despite the frugality in resource consumption of both processes and links, structured overlay networks are highly sensitive to churn and failures. The frugality comes from the before hand construction of the network structure that is able to take advantage of links and processes with higher capacity. However, upon failures the overlay must be rebuilt, precluding the dissemination of data to all processes while this process takes place. As such, in highly dynamic environments where the churn rate is considerable, the cost of constantly rebuilding the overlay may become unbearable. Furthermore, processes closer to the root of the spanning tree handle most of the load of the dissemination, thus impairing the scalability of the approach. This also applies to the aforementioned structures, as certain processes become critical in reaching a large part of the network, and therefore are responsible for handling the network load of large portions of the system.

2.2.2 Unstructured Overlays

In the unstructured approach, links are established randomly among processes without taking into account any efficiency criteria. Therefore, to guarantee that all processes are reachable, and thus the connectivity property ensured, links need to be established with enough redundancy, which has a significant impact on the overlay. The main advantage is that because there are multiple paths available between any two pair of processes, failures and churn do not impair the successful delivery of a given message as it will be routed by some other available path. Furthermore, as there is no implicit structure on the overlay, the churn effect is mitigated as there is no need for global coordination or rebuilding of the overlay. These characteristics yield strong desirable properties in distributed systems: reliability, as connectivity is preserved despite faults; and resilience, as the effect of churn is negligible when compared to structured approaches. Scalability is obtained by requiring each process to know only a small subset of neighbors, typically bounded by the logarithm of the size of system, thus minimizing the load imposed on the maintenance of the overlay and in the dissemination of appli-

cation data. However, departing from global knowledge to only a partial view of the system has a serious impact on the algorithms as they need to address several design questions in order to be successful, which include uniformity and adaptivity (Eugster et al. 2004). The reliability of the overlay stems from the fact that links are established randomly among all the processes in the system. However, when the algorithm is restricted to knowledge of only a subset of processes, this uniform randomness can only be preserved if the partial view of the system is itself a uniform sample from the system. Adaptivity is concerned with the size of the partial view of the system. If the system size is known before hand then the appropriate view size can be easily determined (Kermarrec et al. 2001). However, when the system size is unknown and/or it varies along the time, the partial view size maintained by each process needs to be adapted in order to ensure that the connectivity of the overlay is preserved. Finally, the degree distribution of the overlay should be even, i.e., the variance of the average degree should be small. This is fundamental to ensure load balancing as the load imposed on processes - both in management overhead and in the dissemination effort - is closely related to the degree.

The mechanism used to construct the overlay in the unstructured approach is known as the Peer Sampling Service (PSS) (Jelasity et al. 2007). Due to its importance as the most fundamental building block in unstructured overlays, there is an extensive body of research on building a PSS in a fully decentralized fashion (Lin and Marzullo 1999; Ganesh et al. 2001, 2002; Massoulié et al. 2003; Voulgaris et al. 2005a,b; Leitão et al. 2007; Melamed and Keidar 2008). Existing PSS proposals can be roughly classified as reactive or proactive according to the way they update the processes' view. In the proactive case, processes periodically exchange their views with their neighbors regardless of the actual need to replace failed entries, resulting in each view being a continuous stream of process samples from the network. Examples of proactive PSSs include Cyclon (Voulgaris et al. 2005a) and Newscast (Voulgaris et al. 2005b). In the reactive case, the view is kept unchanged unless some of its entries need to be updated, i.e., for replacing a failed process or for accommodating a process joining the system. Typical examples include Scamp (Ganesh et al. 2001), Araneola (Melamed and Keidar 2008) and HyParView (Leitão et al. 2007b). The trade-off between reactive versus proactive strategies is essentially one between the frugality in terms

of bandwidth consumption of reactive approaches versus the view freshness and diversity provided by the proactive approaches.

Random walks Many distributed algorithms over unstructured overlays often need to sample the network to collect some application specific information. This procedure can be modeled as a random walk, a graph traversal procedure (Gkantsidis et al. 2006; Massoulié et al. 2006). Briefly, a process initiates a random walk by randomly selecting a neighbor from its view and sending it a specific message. The receiver executes some application specific logic, adds some information to the one already carried in the message from the random walk, and forwards the random walk to a randomly selected neighbor. Each random walk is configured with a maximum number of hops it needs to take after which it returns to the initiator. Upon receiving the random walk, the initiator uses the information collected in an application specific manner.

2.2.3 Discussion

The trade-off between the structured approach and the unstructured one is clear. In the structured approach it is possible to take advantage of processes and links with high capabilities thus improving the efficiency of the solution. However, those approaches are sensitive to faults and churns and thus require a stable environment in order to operate properly. On the other hand, unstructured approaches are able to operate under considerable amounts of faults and churn, but the toll to pay is increased overhead when disseminating application data. The trade-off here is between a very efficient, brittle approach or a robust, less efficient one.

Because we target very large scale systems where churn is the norm rather than the exception, our design philosophy throughout this dissertation is to start with a robust unstructured algorithm and then judiciously optimize it for performance. To this end, all algorithms developed assume the existence of a PSS implemented by one of the aforementioned proposals.

2.3 Data Dissemination

The goal of constructing an overlay network, regardless of the particular approach taken, is usually to offer its capabilities to other services able to disseminate application data from one or more sources. In this section we briefly introduce different data dissemination algorithms and highlight the trade-offs among them. We consider three different approaches, namely flooding, trees and epidemic algorithms. Because these approaches rely on the membership information provided by the overlay network, there are naturally some combinations more adequate than others while others overlap in terms of functionality. For instance, flooding a structured overlay with the shape of a tree is similar to using a tree dissemination strategy on an unstructured overlay network. Nonetheless, because these approaches are at different abstraction levels, we conceptually separate them.

2.3.1 Flooding

Flooding is the simplest dissemination strategy. Essentially, all application messages received are relayed to all neighbors on the overlay network. As expected, flooding is very demanding in bandwidth and as such, several optimizations to this naive strategy exist that take advantage of the location of processes in order to reduce the number of duplicates received. In one of those strategies, flooding is only done in the same 'direction' as the received message, as processes on the opposite direction are already expected to have received the message (Ratnasamy et al. 2001).

2.3.2 Tree

In tree approaches, such as (Castro et al. 2002), the dissemination of application level messages uses a reverse path forwarding mechanism to construct and maintain the multicast group, encompassing all processes interested in the dissemination. For each multicast group, the dissemination protocol creates a multicast tree with a unique identifier and uses it to relay messages to the relevant processes. To join the group, a process uses the overlay network to send a message to the multicast group. As the joining request traverses the overlay, each process checks whether it is already part of the desired multicast group, and if it is, it

stops forwarding the message and adds the joining process as a child in the multicast tree. If not, the request is forwarded to the parent until it is adopted by a process or it reaches the root of the tree. In the latter, the root will adopt the joining process as a direct child. The protocol carefully balances the multicast tree in order to ensure an evenly load distribution among the participating processes. To further prevent bottlenecks in certain processes, the protocol provides mechanisms to demote a process's child to a grandchild, thus transferring some of the dissemination effort to its children. Further details of the deployment of these protocols on top of the structured overlay construction mechanisms available, and a detailed comparison of the trade-offs between each one can be found in (Castro et al. 2003b). As the mechanism used to construct the dissemination tree ensures loop-free paths, there are no message duplicates delivered to the application.

2.3.3 Epidemic

Epidemic or gossip dissemination approaches rely on the mathematical models of epidemics (Bailey 1975; Demers et al. 1987; Birman et al. 1999; Eugster et al. 2003b, 2004): if each infected element spreads its infection to a number of random elements in the universe, then all the population will be infected w.h.p. The number of elements that need to be infected by a given element is called the fanout and is a fundamental parameter of the model. Note that even if the model specifies that the elements to be infected need to be selected uniformly at random from the universe, processes usually know only a small fraction of all processes - those in their view. This is addressed by works such as lpbcast which ensure that the view of processes has the same properties than a uniform sample of all processes (Eugster et al. 2003b). Thus, processes pick $fanout$ elements from its view and send the message to them. The choice of the value of the fanout highly influences the fraction of the population that becomes infected. As specified in (Eugster et al. 2004), the ideal fanout value defines a phase transition: below that value the dissemination will reach almost no processes, and above it the dissemination will reach almost all processes. The decision of *when* and *how to* send the message payload to the chosen processes may follow several approaches (Karp et al. 2000), which we describe next. In the *how* to send the message decision there are two options available: push and pull. With push the sender takes the initiative and relays the message to its neighbors as soon as it is

received. On the other hand, with pull the receivers ask periodically the sender for new messages, which will then relay any new message to the receiver. In the *when* decision there are also two options: eager and lazy. Essentially this defines if the message payload should be sent immediately, the eager variant, or only an advertisement of the message, the lazy variant. When combining both design decisions we have four options:

- **Eager push:** the message payload is sent as soon as it is received. This minimizes latency, but at the expense of bandwidth as processes are likely to receive many duplicates. It is the most common strategy and is used by several well-known protocols, such as (Ganesh et al. 2001; Eugster et al. 2003b; Pereira et al. 2003).

- **Lazy push:** upon reception of the message payload the process sends an advertisement of the message to its neighbors. Interested processes can then ask the sender for the payload. In this approach the latency increases considerably as three communication steps are necessary to receive the payload, in a pure lazy push system duplicates are eliminated. This strategy is used in protocols, such as (Liu and Zhou 2006; Carvalho et al. 2007).

- **Eager pull:** periodically processes will ask their neighbors for new messages. Upon reception of the request, processes will send all new messages to the requester. As in the push variant, this approach minimizes latency but at the cost of high bandwidth usage. It is used in protocols, such as (Nguyen et al. 2010; Frey et al. 2010).

- **Lazy pull:** periodically processes will ask their neighbors for new messages. Upon reception of the request, processes will send a message with the identifiers of all new known messages to the requester, who can then selectively pull the relevant messages. This strategy is also known as two-phase pull and allows for an optimal use of bandwidth even though its latency is considerable. It is used in the Network News Transfer Protocol (Feather 2006), which powers Usenet.

The eager versus lazy strategy is clearly a trade-off between bandwidth and latency, while the difference between a push and pull scheme is more subtle. With push processes behave reactively to message exchanges, while with pull

processes behave in a proactive fashion by periodically asking for new messages. Thus, in an environment where messages are sparing, a push strategy has no communication overhead, while the pull approach presents a constant noise due to the periodically check for new messages. Proposals such as (Pianese et al. 2007; Carvalho et al. 2007; Wang et al. 2010) try to overcome the disadvantages of each strategy by combining them in the same protocol.

2.3.4 Discussion

Tree approaches are very efficient in bandwidth usage as, by construction, they avoid sending and receiving message duplicates. Furthermore, by manipulating the depth and branching factor of the tree it is possible to obtain a wide range in end-to-end latency at the cost of putting more load on the interior processes of the tree. However, similarly to structured overlay networks, trees are vulnerable to faults and churn, as the failure of an interior process will preclude the reception of messages in its entire sub-tree. On the other hand, the flooding approach is completely oblivious to faults and churn, as long as the overlay network is connected, all processes will receive all messages. The cost of this resilience is however a large amount of duplicates received, as each process will receive as many copies of a given message as the view size - one for each neighbor. Technically, in the tree there is also a flooding process through its branches, however this is done only to the selected processes (the ones that define the tree according to the propagation strategy), whereas in a pure flooding the message is sent to all the neighbors obtained from the overlay network. Epidemic approaches present an interesting mid-term between the two extremes. The resilience is comparable to flooding, however, they are much less demanding in terms of bandwidth usage. With the use of proper strategies, epidemic approaches can even offer a bandwidth usage similar to the tree, where no duplicates are received.

2.4 Conventions

For readability, we use some conventions throughout this dissertation mostly regarding presentation style.

When presenting algorithm listings we use the following keyword conventions:

- **initially:** invoked when the process starts, used to initialize data structures
- **every** δ**:** invoked every δ time units, usually contains the main loop of the algorithm
- **procedure:** invoked locally by the process
- **send MSG to** p**:** sending of a message MSG from the current process to target process p
- **upon receive MSG:** invoked when a message MSG is received by the current process
- **RandomPick(lst):** picks an element uniformly at random from the list *lst*.

In the literature one can often find the terms peer, process, processor, node or machine to refer to slightly different concepts. Technically, a node, machine or processor is the physical hardware. On top of that we have processes or peers (software) participating in a given distributed algorithm. While it is possible to have several processes running on the same node, for simplicity we do not make such distinction and use all the terms interchangeably.

Finally, one can also find in the literature the related terms message and event. In this dissertation, we consider an event to be a piece of information created and delivered by a process, while the message is the network level entity (usually an Ethernet frame) carrying one or more events.

Bibliography

L.A. Adamic and B.A. Huberman. Zipf's law and the Internet. *Glottometrics*, 3 (1):143–150, 2002. - **Cited** on pages 38 and 40.

Akamai Technologies. Akamai. http://www.akamai.com, 2013. - **Cited** on page 6.

S. Baehni, P.T. Eugster, and R. Guerraoui. Data-aware multicast. In *Dependable Systems and Networks*, 2004. - **Cited** on pages 30 and 57.

Norman Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Hafner Press, second edition, 1975. - **Cited** on pages 3 and 20.

R. Baldoni, R. Beraldi, V. Quema, L. Querzoni, and S. Tucci-Piergiovanni. TERA: topic-based event routing for peer-to-peer architectures. In *International conference on distributed event-based systems*, 2007a. - **Cited** on pages 5, 26, 30, 31 and 57.

Roberto Baldoni, Rachid Guerraoui, R Levy, V Quéma, and Sara Tucci Piergiovanni. Unconscious eventual consistency with gossips. *Stabilization, safety, and security of distributed systems*, 2006. - **Cited** on page 129.

Roberto Baldoni, Roberto Beraldi, Vivien Quema, Leonardo Querzoni, and Sara Tucci-Piergiovanni. Tera: topic-based event routing for peer-to-peer architectures. In *Proceedings of the International Conference on Distributed Event-based Systems*, DEBS, pages 2–13, New York, NY, USA, 2007b. ACM. - **Cited** on page 92.

Raphaël Barazzutti, Pascal Felber, Christof Fetzer, Emanuel Onica, Jean-François Pineau, Marcelo Pasin, Etienne Rivière, and Stefan Weigert.

StreamHub: A Massively Parallel Architecture for High-Performance Content-Based Publish/Subscribe. In *Proceedings of the 7th ACM international conference on Distributed event-based systems - DEBS '13*, page 63, New York, New York, USA, June 2013. ACM Press. ISBN 9781450317580. - **Cited** on page 25.

Michael Ben-Or. Another advantage of free choice (Extended Abstract). In *Proceedings of the second annual ACM symposium on Principles of distributed computing - PODC '83*, pages 27–30, New York, New York, USA, August 1983. ACM Press. ISBN 0897911105. - **Cited** on pages 3 and 131.

R Bhagwan, S Savage, and G Voelker. Understanding availability. In *Proc. of IPTPS: international workshop on Peer-to-Peer Systems*, February 2003a. - **Cited** on page 49.

Ranjita Bhagwan, Stefan Savage, and Geoffrey Voelker. Understanding availability. In *Peer-to-Peer Systems II*, Lecture Notes in Computer Science, pages 256–267. Springer Berlin / Heidelberg, 2003b. - **Cited** on page 97.

Kenneth Birman, Mark Hayden, Oznur Ozkasap, Zhen Xiao, Mihai Budiu, and Yaron Minsky. Bimodal Multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, 1999. ISSN 0734-2071. - **Cited** on pages 2, 3, 20, 26, 61, 62, 99, 100, 103, 119 and 128.

Burton Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13:422–426, 1970. - **Cited** on page 70.

Eric A. Brewer. Towards robust distributed systems (abstract). In *Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing*, page 7, New York, New York, USA, July 2000. ACM Press. ISBN 1581131836. - **Cited** on pages 2 and 99.

Nuno A. Carvalho, José Pereira, Rui Oliveira, and Luís Rodrigues. Emergent Structure in Unstructured Epidemic Multicast. In *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2855-4. - **Cited** on pages 3, 21, 22 and 100.

Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf. Achieving scalability and expressiveness in an Internet-scale event notification service. In

Proceedings of the nineteenth annual ACM symposium on Principles of distributed computing - PODC '00, pages 219–227, New York, New York, USA, July 2000. ACM Press. ISBN 1581131836. - **Cited** on page 4.

M. Castro, P. Druschel, A.M. Kermarrec, and A.I.T. Rowstron. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications*, 20(8):1489–1499, 2002. - **Cited** on pages 4, 16, 19, 56, 61, 75, 91 and 92.

Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. Splitstream: high-bandwidth multicast in cooperative environments. In *Proceedings of the 19th ACM symposium on Operating systems principles*, SOSP, pages 298–313, New York, NY, USA, 2003a. ACM. - **Cited** on pages 7, 61, 63, 74, 76, 84, 92 and 97.

Miguel Castro, Michael Jones, Anne-Marie Kermarrec, Antony Rowstron, Marvin Theimer, Helen Wang, and Alec Wolman. An Evaluation of Scalable Application-Level Multicast Built Using Peer-to-Peer Overlays. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1510–1520, 2003b. - **Cited** on page 20.

R. Chand and P. Felber. Semantic peer-to-peer overlays for publish/subscribe networks. In *International Conference on Parallel and Distributed Computing*, 2005. - **Cited** on page 27.

Chen Chen, Hans-Arno Jacobsen, and Roman Vitenberg. Divide and Conquer Algorithms for Publish/Subscribe Overlay Design. In *Int. Conference on Distributed Computing Systems*. IEEE, 2010. ISBN 978-1-4244-7261-1. - **Cited** on page 58.

Chen Chen, Roman Vitenberg, and Hans-Arno Jacobsen. Scaling Construction of Low Fan-out Overlays for Topic-Based Publish/Subscribe Systems. In *International Conference on Distributed Computing Systems*. IEEE, 2011. ISBN 978-1-61284-384-1. - **Cited** on page 58.

G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg. Constructing scalable overlays for pub-sub with many topics. In *Principles of Distributed Computing*, 2007a. - **Cited** on pages 5, 26, 27, 31 and 58.

G. Chockler, R. Melamed, Y. Tock, and R. Vitenberg. Spidercast: a scalable interest-aware overlay for topic-based pub/sub communication. In *International Conference on Distributed Event-Based Systems*, 2007b. - **Cited** on pages 5, 26, 27, 30, 52, 57 and 60.

Y. Chu, S.G. Rao, S. Seshan, and H. Zhang. A case for end system multicast. *IEEE Journal on Selected Areas in Communications*, 20:1456–1471, 2002. - **Cited** on pages 4 and 61.

S. Cimmino, C. Marchetti, and R. Baldoni. A Guided Tour on Total Order Specifications. In *The Ninth IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS' 03)*, pages 187–187. IEEE, 2003. ISBN 0-1795-2054-5. - **Cited** on page 8.

Bram Cohen. Incentives build robustness in bittorrent, 2003. - **Cited** on pages 4 and 7.

Bram Cohen. The bittorrent protocol specification. http://www.bittorrent.org/beps/bep_0003.html, January 2008. - **Cited** on pages 4 and 7.

James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. Spanner: Google's globally-distributed database. In *Operating Systems Design and Implementation*, 2012. ISBN 978-931971-96-6. - **Cited** on page 101.

Alan Dcmers Dan, Carl Hauser, Wes Irish, John Larson, Scott Shenkcr, Howard Sturgis, Dan Swinehart, Doug Terry, Alan Demers, Dan Greene, and Scott Shenker. Epidemic algorithms for replicated database maintenance. In *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, 1987. ISBN 0-89791-239-4. - **Cited** on page 100.

Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter

Vosshall, and Werner Vogels. Dynamo: amazon's highly available key-value store. *SIGOPS Operating Systems Review*, 41:205–220, 2007. - **Cited** on pages 3, 4 and 62.

Xavier Défago, André Schiper, and Péter Urbán. Total order broadcast and multicast algorithms: Taxonomy and survey. In *ACM Computing surveys*, volume 36, 2004. - **Cited** on pages 4, 7, 8, 99, 102, 118, 127 and 128.

Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the 6th ACM Symposium on Principles of distributed computing*, PODC, pages 1–12, New York, NY, USA, 1987. ACM. - **Cited** on pages 2, 3, 20, 62, 65, 87, 128 and 132.

P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. *ACM Computing Survey*, 35(2), 2003a. ISSN 0360-0300. - **Cited** on pages 5, 25, 26 and 132.

Patrick Eugster, Rachid Guerraoui, Sidath Handurukande, Petr Kouznetsov, and Anne-Marie Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems*, 21(4):341–374, 2003b. ISSN 0734-2071. - **Cited** on pages 3, 20, 21, 26, 30, 31, 61, 62, 100, 103 and 119.

Patrick Eugster, Rachid Guerraoui, Anne-Marie Kermarrec, and Laurent Massoulié. From Epidemics to Distributed Computing. *IEEE Computer*, 37(5): 60–67, May 2004. - **Cited** on pages 7, 17, 20, 28, 31 and 32.

C. Feather. Network News Transfer Protocol (NNTP). RFC 3977 (Proposed Standard), October 2006. URL `http://www.ietf.org/rfc/rfc3977.txt`. Updated by RFC 6048. - **Cited** on page 21.

Zongming Fei and Mengkun Yang. A proactive tree recovery mechanism for resilient overlay multicast. *IEEE/ACM Transactions on Networking*, 15:173–186, 2007. - **Cited** on page 94.

Pascal Felber and Fernando Pedone. Probabilistic Atomic Broadcast. In *International Symposium on Reliable Distributed Systems*, 2002. - **Cited** on pages 8, 127 and 132.

Mario Ferreira, Joao Leitao, and Luis Rodrigues. Thicket: A Protocol for Building and Maintaining Multiple Trees in a P2P Overlay. In *Proceedings of the 29th IEEE International Symposium on Reliable Distributed Systems*, SRDS, pages 293–302, New Delhi, India, 2010. IEEE Computer. - **Cited** on pages 95 and 97.

Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, April 1985. ISSN 0004-5411. - **Cited** on page 2.

S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6):784–803, 1997. ISSN 10636692. - **Cited** on page 2.

P. Fraigniaud, P. Gauron, and M. Latapy. Combining the use of clustering and scale-free nature of exchanges into a simple and efficient P2P system. In *International Conference on Parallel and Distributed Computing*, 2005. - **Cited** on pages 28, 35 and 38.

D. Frey, R. Guerraoui, A.-M. Kermarrec, M. Monod, and V. Quema. Stretching gossip with live streaming. In *Proceedings of the 39th IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN, pages 259–264, Budapest, Hungary, 2009. IEEE Computer Society. - **Cited** on page 61.

Davide Frey, Rachid Guerraoui, Anne-Marie Kermarrec, and Maxime Monod. Boosting Gossip for Live Streaming. In *2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P)*, pages 1–10. IEEE, August 2010. ISBN 978-1-4244-7140-9. - **Cited** on page 21.

Robert Gallager, Pierre Humblet, and Philip Spira. A Distributed Algorithm for Minimum-Weight Spanning Trees. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 5(1), 1983. ISSN 0164-0925. - **Cited** on page 16.

Ayalvadi Ganesh, Anne-Marie Kermarrec, and Laurent Massoulié. Scamp: Peer-to-Peer Lightweight Membership Service for Large-Scale Group Communication. In *Networked Group Communication*, Lecture Notes in Computer Science,

pages 44–55. Springer Berlin / Heidelberg, 2001. - **Cited** on pages 3, 7, 17, 21, 28, 31 and 62.

Ayalvadi Ganesh, Anne-Marie Kermarrec, and Laurent Massoulié. HiScamp: self-organizing hierarchical membership protocol. In *Proceedings of the 10th workshop on ACM SIGOPS European workshop*, pages 133–139. ACM, 2002. - **Cited** on page 17.

John Gantz. The Expanding Digital Universe. Technical report, IDC White Paper - sponsored by EMC, 2007. - **Cited** on pages 1, 61 and 96.

John Gantz. The Diverse and Exploding Digital Universe. Technical report, IDC White Paper - sponsored by EMC, 2008. - **Cited** on pages 1, 61 and 96.

Seth Gilbert and Nancy Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News*, 33(2), 2002. ISSN 01635700. - **Cited** on pages 2 and 99.

Sarunas Girdzijauskas, Gregory Chockler, Ymir Vigfusson, Yoav Tock, and Roie Melamed. Magnet: practical subscription clustering for Internet-scale publish/-subscribe. In *International Conference on Distributed Event-Based Systems*. ACM Press, 2010. ISBN 9781605589275. - **Cited** on pages 5 and 56.

C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks: algorithms and evaluation. *Performance Evaluation - P2P Computing Systems*, 63(3), 2006. ISSN 0166-5316. - **Cited** on page 18.

R Guerraoui and A Schiper. Total order multicast to multiple groups. *Distributed Computing Systems, 1997. ...*, 1997. - **Cited** on page 135.

R. Guerraoui and A. Schiper. The generic consensus service. *IEEE Transactions on Software Engineering*, 27(1):29–41, 2001. ISSN 00985589. - **Cited** on page 2.

Rachid Guerraoui, Rui Oliveira, and André Schiper. Stubborn communication channels. Technical report, Tech. Rep.98-278, Département d'Informatique, École Polytechnique Fédérale de Lausanne, 1998. - **Cited** on page 14.

Indranil Gupta, R van Renesse, and KP Birman. A probabilistically correct leader election protocol for large groups. In Maurice Herlihy, editor, *Distributed*

Computing, volume 1914 of *Lecture Notes in Computer Science*, pages 89–103, Berlin, Heidelberg, March 2000. Springer Berlin Heidelberg. ISBN 978-3-540-41143-7. - **Cited** on page 3.

S. Handurukande, A.-M. Kermarrec, F. Le Fessant, L. Massoulié, and S. Patarin. Peer sharing behaviour in the eDonkey network, and implications for the design of server-less file sharing systems. *ACM Eurosys*, 2006. - **Cited** on pages 28, 35 and 38.

Mark Hayden and Kenneth Birman. Probabilistic Broadcast. Technical Report TR96-1606, Cornell University, 1996. - **Cited** on pages 100, 127 and 132.

John Jannotti, David Gifford, Kirk Johnson, M. Frans Kaashoek, and James O'Toole. Overcast: Reliable Multicasting with an Overlay Network. In *Usenix OSDI Symposium 2000*, pages 197–212, October 2000. - **Cited** on page 15.

Márk Jelasity, Spyros Voulgaris, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. Gossip-based peer sampling. *ACM Transactions on Computer Systems*, 25, 2007. - **Cited** on pages 3, 14, 17, 26, 27, 28, 30, 31, 47, 58, 62 and 64.

Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. T-man: Gossip-based fast overlay topology construction. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 53:2321–2339, 2009. - **Cited** on pages 27, 32, 92 and 98.

S. Jun and M. Ahamad. Feedex: collaborative exchange of news feeds. In *Int. Conference on World Wide Web*, 2006. - **Cited** on page 25.

R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *Symposium on Foundations of Computer Science*. IEEE Computer Society, 2000. - **Cited** on page 20.

A-M. Kermarrec, L. Massoulié, and A. Ganesh. Probabilistic reliable dissemination in large-scale systems. *Transactions on Parallel and Distributed Systems*, 14, 2001. - **Cited** on pages 7, 17 and 42.

Anne-Marie Kermarrec, Alessio Pace, Vivien Quema, and Valerio Schiavoni. NAT-resilient Gossip Peer Sampling. In *2009 29th IEEE International Conference on Distributed Computing Systems*, pages 360–367. IEEE, June 2009. ISBN 978-0-7695-3659-0. - **Cited** on page 13.

Boris Koldehofe. Simple gossiping with balls and bins. *International Conference on Principles of Distributed Systems*, 2002. - **Cited** on pages 100, 101, 103, 104, 109, 117, 118 and 119.

Boris Koldehofe. Buffer management in probabilistic peer-to-peer communication protocols. In *Proceedings of the 22nd IEEE International Symposium on Reliable Distributed Systems*, SRDS, pages 76–85, Florence, Italy, 2003. IEEE Computer. - **Cited** on pages 36 and 72.

Dejan Kostic, Adolfo Rodriguez, Jeannie Albrecht, and Amin Vahdat. Bullet: High bandwidth data dissemination using an overlay mesh. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP, pages 282–297, New York, NY, USA, 2003. ACM. - **Cited** on page 93.

Avinash Lakshman and Prashant Malik. Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 44(2):35, April 2010. ISSN 01635980. - **Cited** on page 4.

Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7), 1978. ISSN 00010782. - **Cited** on pages 7, 8, 99 and 128.

J. Leitão, J. Pereira, and L. Rodrigues. Hyparview: A membership protocol for reliable gossip-based broadcast. In *International Conference on Dependable Systems and Networks (IEEE DSN)*, pages 419–428. IEEE Computer Society, 2007. - **Cited** on page 17.

João Leitão, Robbert van Renesse, and Luís Rodrigues. Balancing gossip exchanges in networks with firewalls. In *International Workshop on Peer-to-Peer Systems (IPTPS '10)*, page 7. USENIX Association, April 2010. - **Cited** on page 14.

Joao Leitão, José Pereira, and Luís Rodrigues. Epidemic Broadcast Trees. In *Proceedings of the 22nd IEEE International Symposium on Reliable Distributed*

Systems, SRDS, pages 301–310, Beijing, China, 2007a. IEEE Computer. - **Cited** on pages 94 and 95.

Joao Leitão, José Pereira, and Luís Rodrigues. HyParView: A membership protocol for reliable gossip-based broadcast. In *Proceedings of the 37th IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN, pages 419–429, Edinburgh, Scotland, 2007b. IEEE Computer Society. - **Cited** on pages xvi, 3, 17, 62, 64, 66 and 73.

Lorenzo Leonini, Etienne Rivière, and Pascal Felber. SPLAY: Distributed systems evaluation made simple (or how to turn ideas into live systems in a breeze). In *Symposium on Networked Systems Design and Implementation*, NSDI, pages 185–198, Berkely, CA, USA, 2009. Usenix Association. - **Cited** on pages 10, 11, 41 and 76.

Z. Li, G. Xie, and Z. Li. Towards reliable and efficient data dissemination in heterogeneous peer-to-peer systems. In *Proceedings of the 22th IEEE International Parallel and Distributed Processing Symposium*, IPDPS, pages 1–12, Miami, FL, USA, 2008. IEEE Computer Society. - **Cited** on page 94.

Zhenyu Li, Gaogang Xie, Kai Hwang, and Zhongcheng Li. Churn-resilient protocol for massive data dissemination in p2p networks. *IEEE Transactions on Parallel and Distributed Systems*, 22:1342–1349, 2011. - **Cited** on page 94.

Jin Liang, Steven Ko, Indranil Gupta, and Klara Nahrstedt. MON : On-demand Overlays for Distributed System Management. In *Proceedings of the 2nd conference on Real, Large Distributed Systems*, WORLDS, pages 13–18, Berkely, CA, USA, 2005. Usenix Association. - **Cited** on pages 61, 63 and 93.

Meng Lin and Keith Marzullo. Directional Gossip: Gossip in a Wide Area Network. In *Proceedings of Third European Dependable Computing Conference*, volume 1667 of *Lecture Notes in Computer Science*, pages 364–379. Springer, 1999. - **Cited** on page 17.

H. Liu, V. Ramasubramanian, and E.G. Sirer. Client behavior and feed characteristics of RSS, a publish-subscribe system for web micronews. In *Internet Measurement Conference*, 2005. - **Cited** on pages 27, 38 and 40.

Jiangchuan Liu and Ming Zhou. Tree-assisted gossiping for overlay video distribution. *Multimedia Tools and Applications*, 29:211–232, 2006. - **Cited** on pages 21, 63 and 88.

LiveJournal, Inc. `http://www.livejournal.com/stats.bml`, 2013. - **Cited** on page 37.

Thomas Locher, Remo Meier, Stefan Schmid, and Roger Wattenhofer. Push-to-pull peer-to-peer live streaming. In *Distributed Computing*, Lecture Notes in Computer Science, pages 388–402. Springer Berlin / Heidelberg, 2007. - **Cited** on page 7.

M. Luby. *Pseudorandomness and Cryptographic Applications*. Princeton University Press, 1994. ISBN 0691025460. - **Cited** on page 32.

NA Lynch. *Distributed algorithms*. Morgan Kaufmann Publishers Inc., January 1996. ISBN 1558603484. - **Cited** on pages 2 and 13.

L. Massoulié, A-M. Kermarrec, and A. Ganesh. Network awareness and failure resilience in self-organising overlay networks. In *Symposium on Reliable Distributed Systems*, 2003. - **Cited** on pages 17 and 27.

Laurent Massoulié, Erwan Le Merrer, Anne-Marie Kermarrec, and Ayalvadi Ganesh. Peer counting and sampling in overlay networks: random walk methods. In *Principles of Distributed Computing*, 2006. - **Cited** on page 18.

M. McGlohon. *Structural Analysis of Large Networks: Observations and Applications*. PhD thesis, Carnegie Mello University, 2010. - **Cited** on pages 35 and 53.

Wang Mea, Li Baochun, Mea Wang, and Baochun Li. R2: Random Push with Random Network Coding in Live Peer-to-Peer Streaming. *IEEE Journal on Selected Areas in Communications*, 25(9):1655–1666, December 2007. ISSN 0733-8716. - **Cited** on page 97.

Roie Melamed and Idit Keidar. Araneola: A scalable reliable multicast system for dynamic environments. *Journal of Parallel and Distributed Computing*, 68 (12):1539–1560, 2008. - **Cited** on page 17.

Alberto Montresor, Márk Jelasity, and Ozalp Babaoglu. Chord on demand. In *Proceedings of the 5th IEEE International Conference on Peer-to-Peer Computing*, P2P, pages 87–94, Washington, DC, USA, 2005. IEEE Computer Society. - **Cited** on pages 3 and 62.

Anh Tuan Nguyen, Baochun Li, and Frank Eliassen. Chameleon: Adaptive Peer-to-Peer Streaming with Network Coding. In *2010 Proceedings IEEE INFOCOM*, pages 1–9. IEEE, March 2010. ISBN 978-1-4244-5836-3. - **Cited** on pages 21 and 97.

A. Nunes, J. Marques, and J. Pereira. Seeds: The social internet feed caching and dissemination architecture. In *INForum Simpósio de Informática*, 2009. - **Cited** on page 25.

Melih Onus and Andréa W. Richa. Parameterized Maximum and Average Degree Approximation in Topic-Based Publish-Subscribe Overlay Network Design. In *International Conference on Distributed Computing Systems*. IEEE, 2010. ISBN 978-1-4244-7261-1. - **Cited** on page 58.

Jay A. Patel, Etienne Rivière, Indranil Gupta, and Anne-Marie Kermarrec. Rappel: Exploiting interest and network locality to improve fairness in publish-subscribe systems. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 53:2304–2320, 2009. - **Cited** on pages 26, 38 and 93.

Fernando Pedone and André Schiper. Optimistic atomic broadcast: a pragmatic viewpoint. *Theoretical Computer Science*, 291(1), 2003. ISSN 03043975. - **Cited** on pages 124 and 127.

J. Pereira, L. Rodrigues, R. Oliveira, and A.-M. Kermarrec. Neem: Network-friendly epidemic multicast. In *Symposium on Reliable Distributed Systems*, 2003. - **Cited** on pages 21 and 28.

Fabio Pianese, Diego Perino, JoaquÍn Keller, and Ernst Biersack. PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System. *IEEE Transactions on Multimedia*, 9(8):1645–1660, 2007. ISSN 15209210. - **Cited** on page 22.

Boris Pittel. On Spreading a Rumor. *SIAM Journal on Applied Mathematics*, 47 (1), 1987. ISSN 0036-1399. - **Cited** on page 109.

PlanetLab. PlanetLab. http://www.planet-lab.org, 2013. - **Cited** on pages 41, 63 and 76.

Charles Plaxton, Rajmohan Rajaraman, and Andréa Richa. Accessing nearby copies of replicated objects in a distributed environment. *ACM Symposium on Parallel Algorithms and Architectures*, 1997. - **Cited** on page 15.

L. Querzoni. Interest clustering techniques for efficient event routing in large-scale settings. In *International Conference on Distributed Event-Based Systems*, 2008. - **Cited** on page 58.

Sylvia Ratnasamy, Mark Handley, Richard M. Karp, and Scott Shenker. Application-level multicast using content-addressable networks. In *Workshop on Networked Group Communication*, NGC '01. Springer-Verlag, 2001. - **Cited** on pages 15, 16, 19 and 56.

Etienne Rivière and Spyros Voulgaris. *Gossip-Based Networking for Internet-Scale Distributed Systems*, volume 78 of *Lecture Notes in Business Information Processing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-20861-4. - **Cited** on page 3.

Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Middleware*, Lecture Notes in Computer Science, pages 329–350. Springer Berlin / Heidelberg, 2001. - **Cited** on pages 16 and 92.

Laura S. Sabel and Keith Marzullo. Election Vs. Consensus in Asynchronous Systems. Technical report, Cornell University, February 1995. - **Cited** on page 2.

Yasushi Saito and Marc Shapiro. Optimistic replication. *ACM Computing Surveys*, 37(1), 2005. ISSN 03600300. - **Cited** on pages 101 and 128.

Stefan Saroiu, P. Krishna Gummadi, and Steven D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking*, 2002. - **Cited** on pages 28, 35 and 38.

Bianca Schroeder and Garth Gibson. Disk failures in the real world: What does an MTTF of 1, 000, 000 hours mean to you? In *Proceedings of the 5th USENIX conference on File and Storage Technologies*, number September in FAST '07, pages 1–16, 2007. - **Cited** on pages 2, 7 and 96.

Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. DRAM errors in the wild. In *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems - SIGMETRICS '09*, page 193, New York, New York, USA, June 2009. ACM Press. ISBN 9781605585116. - **Cited** on pages 2, 7 and 96.

António Sousa, José Pereira, Francisco Moura, and Rui Oliveira. Optimistic total order in wide area networks. In *IEEE Symposium on Reliable Distributed Systems*, 2002. - **Cited** on pages 101, 124, 127 and 128.

Ion Stoica, Robert Morris, David Liben-Nowell, David Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Networking Transactions*, 11(1):17–32, 2003. ISSN 1063-6692. - **Cited** on page 16.

Chunqiang Tang, Rong N. Chang, and Christopher Ward. Gocast: Gossip-enhanced overlay multicast for fast and dependable group communication. In *Proceedings of the 35th IEEE/IFIP International Conference on Dependable Systems and Networks*, DSN, pages 140–149, Washington, DC, USA, 2005. IEEE Computer Society. - **Cited** on page 94.

Twitter Engineering. Murder: Fast datacenter code deploys using BitTorrent. `http://t.co/uo5rEN4`, September, 2012. - **Cited** on page 61.

Robbert van Renesse, Ken Birman, and Werner Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems*, 21:164–206, 2003. - **Cited** on page 63.

Robbert van Renesse, Yaron Minsky, and Mark Hayden. A gossip-style failure detection service. In *Proceedings of the ACM/IFIP/USENIX International Conference on Middleware*, Middleware, pages 389–409, New York, Inc. New York, NY, USA, 2007. Springer-Verlag. - **Cited** on pages 3, 62 and 63.

Vidhyashankar Venkataraman, Kaouru Yoshida, and Paul Francis. Chunkyspread: Heterogeneous Unstructured Tree-Based Peer-to-Peer Multicast. In *Proceedings of the 14th IEEE International Conference on Network Protocols*, ICNP, pages 2–11. IEEE Computer Society, 2006. - **Cited** on page 92.

Hakon Verespej and Joseph Pasquale. A Characterization of Node Uptime Distributions in the PlanetLab Test Bed. *2011 IEEE 30th International Symposium on Reliable Distributed Systems*, pages 203–208, October 2011. - **Cited** on pages 2, 7 and 96.

Werner Vogels. Eventually consistent. *Communications of the ACM*, 52(1), 2009. ISSN 00010782. - **Cited** on pages 2, 99 and 128.

Spyros Voulgaris and Maarten van Steen. Hybrid dissemination: Adding determinism to probabilistic multicasting in large-scale p2p systems. In *Proceedings of the ACM/IFIP/USENIX International Conference on Middleware*, Middleware, pages 389–409, New York, Inc. New York, NY, USA, 2007. Springer-Verlag. - **Cited** on page 93.

Spyros Voulgaris, Daniela Gavidia, and Maarten van Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13:197–217, 2005a. - **Cited** on pages 17 and 87.

Spyros Voulgaris, Márk Jelasity, and Maarten Van Steen. A Robust and Scalable Peer-to-Peer Gossiping Protocol. In *Agents and Peer-to-Peer Computing*, volume 2872 of *Lecture Notes in Computer Science*, pages 47–58. Springer-Verlag Berlin, Heidelberg, 2005b. - **Cited** on page 17.

Spyros Voulgaris, Etienne Rivière, Anne-Marie Kermarrec, and Maarten van Steen. Sub-2-sub: Self-organizing content-based publish subscribe for dynamic large scale collaborative networks. In *International Workshop on Peer-to-Peer Systems*, 2006. - **Cited** on page 25.

Feng Wang, Yongqiang Xiong, and Jiangchuan Liu. mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming. *IEEE Transactions on Parallel and Distributed Systems*, 21(3):379–392, March 2010. ISSN 1045-9219. - **Cited** on page 22.

S. Whittaker, L. Terveen, W. Hill, and L. Cherny. The dynamics of mass interaction. In *Conference on Computer supported cooperative work*, 1998. - **Cited** on pages 35 and 53.

Wikimedia Foundation. Wikipedia database dumps. `http://dumps.wikimedia.org/`, 2013. - **Cited** on page 37.

N.C. Wormald. Models of random regular graphs. *Surveys in combinatorics*, 276: 239–298, 1999. - **Cited** on page 57.

Ben Zhao, John Kubiatowicz, and Anthony Joseph. Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Technical Report UCB/CSD-01-1141, UC Berkeley, April 2001. - **Cited** on page 16.

Shelley Zhuang, Ben Zhao, Anthony Joseph, Randy Katz, and John Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*, NOSSDAV, pages 11–20, New York, NY, USA, 2001. ACM. - **Cited** on page 16.