

Paths to Stardom: Calibrating the Potential of a Peer-based Data Management System

Mihai Lupu
Singapore-MIT Alliance
National University of Singapore
mihailup@comp.nus.edu.sg

Beng Chin Ooi
School of Computing
National University of Singapore
ooibc@comp.nus.edu.sg

Y. C. Tay
School of Computing
National University of Singapore
dcstayyc@nus.edu.sg

ABSTRACT

As peer-to-peer (P2P) networks become more familiar to the database community, intense interest has built up in using their scalability and resilience properties to scale database applications. Indexing methods are adapted on top of P2P networks and querying methods are developed to handle the data distribution on different nodes. These procedures largely depend on how nodes are connected to each other. So far, limited attempts have been made to compare all these systems in a generalized framework. This is because the systems are quite different from each other, and there are so many of them that brute force comparison is practically impossible. Fortunately, it has recently been observed that a large subset of the most important P2P networks share a common algebraic and combinatorial base, in the form of Cayley graphs.

The specific requirements of Peer-based Data Management Systems (PDMS), such as query completeness, range queries, load balancing, communication overhead, and scalability are strongly related to the properties of the underlying graphs, and naturally, some graphs are better than others. We conduct a comprehensive graph-theoretic analysis from the point of view of PDMS and identify the necessary conditions for a graph to be considered a potential network structure for a PDMS. In so doing, we provide a basis for the future development of such networks. We complement our analytical study with extensive experimental results and identify three measures that provide significant information about the potential of a [Cayley] graph to support the requirements of a PDMS.

Categories and Subject Descriptors

C.4 [Computer Systems Organization]: Performance of systems—*Fault tolerance*; E.1 [Data]: Data Structures—*Graphs and networks*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGMOD'08, June 9–12, 2008, Vancouver, BC, Canada.
Copyright 2008 ACM 978-1-60558-102-6/08/06 ...\$5.00.

General Terms

Performance, Reliability, Theory

Keywords

Cayley graph, peer-to-peer, query completeness

1. INTRODUCTION

A large number of peer-to-peer (P2P) networks have been introduced in the literature since their popular advent in the late 1990s. In particular, structured P2P overlays have gained much attention since 2001. They are noted mainly for their theoretical properties such as balancing of communication, storage and processing load, and elegance of design. While unstructured P2P networks may have received much public attention for their file-sharing utility, they are not suitable for most other applications as they do not provide guarantees of any kind. In particular, existing data may not always be retrievable in unstructured P2P networks even in optimal operating conditions. This renders the networks largely unusable for distributed databases, or any other application where the existence or non-existence of a piece of information makes an essential and otherwise unpredictable difference in the result of a query.

In the context of structured overlays, sensitivity to temporary node absence has also not been fully addressed. Though there has been considerable interest in the database community for P2P systems [1, 14, 16, 27] (a result of which has been the creation of a new acronym: PDMS - Peer-based Data Management System), most works assume that node failures are dealt with at the lower levels of the systems' architectures via expensive periodic network stabilizations. Note for instance that PDMS research papers often describe node failures as being outside their scope and focus on data distribution and query processing.

We have to take a step back and look at the underlying structure and, in particular, at its potential of tolerating node failures. Function of this tolerance, more or less costly compensation measures will need to be taken in order to guarantee a proper functionality. These measures introduce overhead, and based on our expectation of node failures, we need, at design time, to make a decision on what kind of structure to use for the underlying interconnection network. In this context, our work provides an analytical basis, coupled with a simulation environment, by which the designer of a PDMS can choose the underlying network. What makes CAN [30] better than Chord[33], or BATON [15] better than CAN, or more generally, one structure better than another?

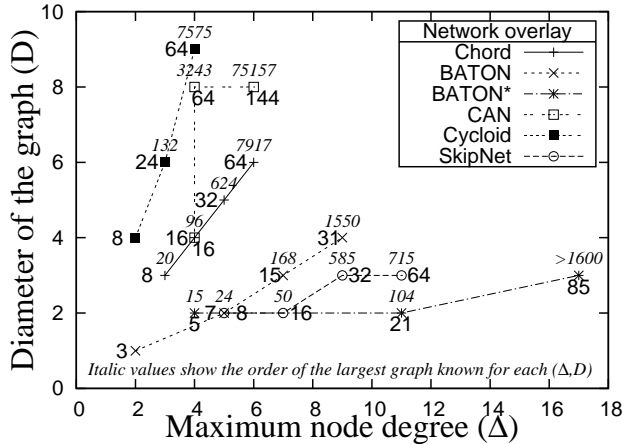


Figure 1: Maximal cardinality of instances of some well-known networks

And what does *better* mean? These are the questions that we seek to answer now.

There is a long stream of proposals on structured P2P networks. Chord [33], CAN [30], BATON [15], SkipNet [12], Viceroy [25], HyperCuP [31] and Overcast [17] form a representative sample of the variety that exists. A good survey may be found in [24].

The various structured P2P networks may be grouped into two types: those having small routing tables but large routing paths, and those having small routing paths but large routing tables. Two extremes are BATON* [16] and Cycloid [32]. While the former increases its maximum degree exponentially but keeps the diameter logarithmic with a large base, the latter maintains a constant maximum degree, but increases its diameter polynomially. The balance between the number of neighbors and the maximum path between two nodes is known as the (*degree, diameter*)-problem in graph theory. In Figure 1 we plot the order (maximum number of nodes) of some small instances of well-known P2P networks, to show how each overlay increases its order either by an increased diameter or by an increased degree. We also plot the order of the largest known graph for each (degree, diameter) pair. Interestingly, this reveals much space for improvement: known graphs are able to hold many more nodes using the same degree and diameter as the overlays exemplified in this figure. Therefore, we conjecture that not all graphs are good models for PDMS and we seek to identify a set of measures to help us decide which is a potentially good structure and which is not.

Structured overlays also differ as a consequence of their intended usage, and correspondingly, of the algorithms implemented on top of them. However, this may be seen as just a particular way to label the nodes, and given an appropriate re-labeling function, the algorithms can be applied on top of each overlay. The existence of such a re-labeling function depends only on the type of structure that is used to connect the nodes. We will see that even apparently very different network overlays, such as BATON (based on a tree structure) and Chord (based on a ring structure) can be very similar if an appropriate relabeling function is found.

Recently, [29] and [28] have observed that most structured overlay networks have Cayley graphs as their static architecture. These abstract algebra graphs have been studied in the late 1980s and early 1990s in the context of parallel com-

puters (processor interconnection networks), but properties that are considered important in that context, like planarity - the possibility to arrange the nodes without intersecting the edges, are not relevant in P2P overlays. We propose a new analysis for a better understanding of a graph’s potential to constitute the basis of the next PDMS. Additionally, we address the proposal of [29] and [28] to use the Star graph as underlying structure and analyze it from the point of view of a PDMS.

This work is part of a larger, comprehensive project under development at the School of Computing, NUS, entitled BESTPEER. This framework provides a broad range of querying and searching facilities over various types of data sources and is build on a flexible P2P platform that can switch smoothly between structured and unstructured overlays.

1.1 Motivation

It is important to understand the extent to which the overlay network is capable of handling faults. This is particularly true for systems that deal with aggregate queries, and even if the assumption is that an optimal routing scheme is implemented. For instance, in defining their aggregate query semantic, Bawa et al. [4] assume that the failure of a node can be compensated with an update in the neighborhood. While this may be true in most cases, there may also be situations where the quasi-simultaneous failure of a relatively small sets of nodes could divide the network in two disjoint subnetworks. Such a situation will trigger a massive update process in all the nodes of the original network. A system designer will need to consider this problem.

The same problem appears in [19], where the authors deal with data mappings on P2P networks. Temporarily unavailable nodes may in this case result in a lack of constraints, and consequently, the deduction of false mappings.

More recently, the database community has been interested in providing quality guarantees to systems developed on top of P2P networks. In [22] and later in [8] and [7], the authors argue for correctness, availability and load balancing guarantees for data and query distribution on P2P networks. In the meantime, they developed their techniques “in the context of a general P2P indexing framework that can be instantiated with most P2P index structures in the literature”. As we will show in this paper, interconnection networks have significantly different fault tolerances. Therefore, before guaranteeing correctness or load balancing, it is important to take a closer look at whether the network even allows a maximal connectivity in the presence of failures.

Our preliminary analysis leads us to the observation that many existing PDMS are mappable to Cayley graphs despite their apparent differences. For instance, we have taken a close look at BATON [15], a recently proposed tree-based PDMS capable of answering both equality and range queries. We have observed that, provided the addition of one node (regardless of the size of the tree), there exists a mapping between the tree and the chordal ring representations. In particular, if directed, this chordal ring representation forms the well-known Chord network [33]. In Figure 2, it’s not difficult to observe that the function $\Upsilon_h : \{1, \dots, h\} \times \{1, \dots, 2^{l-1}\} \rightarrow \{1, \dots, 2^h - 1\}$, $\Upsilon_h((l, j)) = 2^{h-l}(2j - 1)$ represents an isomorphism between the two graphs.

The existence of such a mapping indicates that the study of the subclass of structured P2P networks based on Cayley

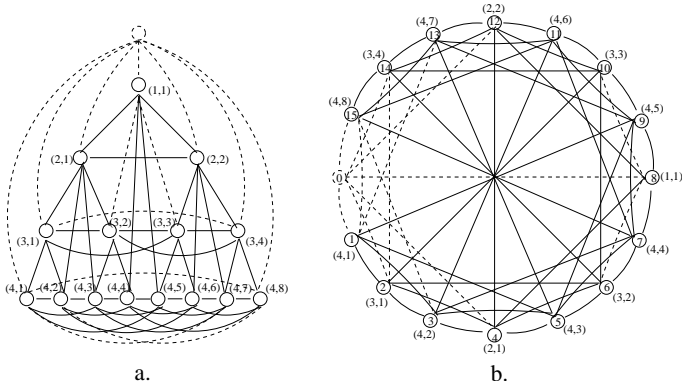


Figure 2: The BATON structure drawn as in [23] (a.) and as the Cayley graph representing Chord (b.)

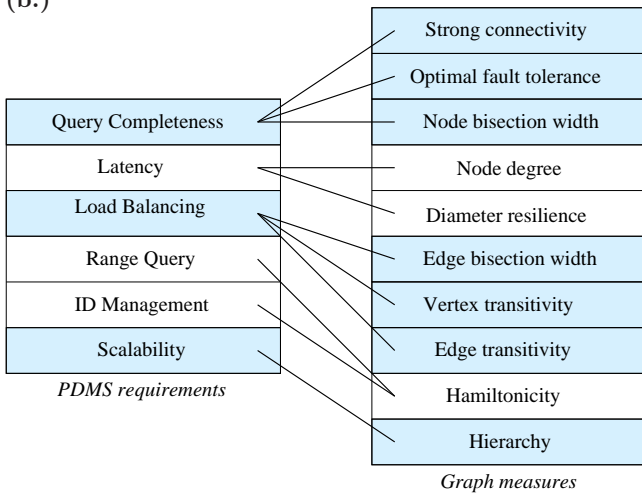


Figure 3: Relationships between PDMS requirements and graph properties

graphs provides useful insights for a much larger array of networks than what may be initially expected.

1.2 Contributions

We systematically analyze a representative subset of the existing P2P networks used in PDMS and show the necessary conditions for the underlying graph structures, in terms of common measures of graph optimality. Each such measure has a direct effect on one of the properties of a PDMS, as indicated in Figure 3.

In particular, we are interested in query completeness, which we define here as the amount of relevant information present in the live nodes of the network when a query is issued. This is obviously determined by the connectivity of the graph underlying the network, by fault tolerance, and as we will see, by node bisection width. The main idea is to consider a network as being better if the failure of a node or a small set of nodes does not create a “cone of shadow”, i.e., it does not make another set of nodes unavailable.

Our contributions are as follows:

- We improve upon the Cayley graph models presented previously for P2P networks, correcting two existing misrepresentations in the models of the CAN and Viceroy networks. We do this in Section 4.
- We re-examine the “basic” set of graph-theoretic mea-

asures which have been previously considered in the literature and show that they provide little and misleading decision support for PDMS system designers in Section 5:

Node degree: Refers to the number of neighbors that each peer maintains. A higher value increases reliability but also maintenance cost.

Vertex transitivity: Expresses the symmetry of the network. If a graph is vertex transitive, then any node can be equally loaded. A perfect counter-example of this property is a tree – the root will always be overloaded.

Edge transitivity: Expresses the balancing of the load on each edge and is assumed to correspond to communication bandwidth balancing.

Optimal fault tolerance: Indicates that the number of nodes that can fail without segmenting the network into two disjoint sub-networks is minimal. Though apparently useful, we will show that for P2P networks, this measure is insufficiently informative.

- Having observed that previously used graph measures are not sufficiently informative for P2P networks, in Section 6 we provide an in-depth analysis of the overlays based on:

Diameter resilience: Characterizes the effects of node failure on the maximum distance between any two nodes.

Bisection width: Refers to the number of nodes or edges that need to be removed to segment the network into two subnetworks of (almost) equal size.

Hierarchy: Characterizes the capacity of a network overlay to expand its ID space.

Hamiltonicity: The existence or absence of a way to traverse all the nodes in an orderly fashion affects the fault tolerance of the network, the routing method and the possibility of answering range queries efficiently.

- Along with the analysis of the properties of each network, we perform extensive simulation experiments in Section 7:

- We provide a plug-and-play network simulator based on Cayley graphs, with which we have implemented and tested 25 different network architectures.
- We test network performance in terms of query success rates and maximum path length in the context of an optimal routing method which always finds a path if one exists.
- We implement routing protocols described in the original papers of Cycloid [32], Viceroy [25], HyperCuP [31] and Chord [33] and compare them with their estimated performance based on the default graph routing method.

- We show the flexibility of our simulator by implementing a non-Cayley network, BATON, and compare its performance against the execution on PlanetLab of the real network.

In Section 8, we conclude with the set of measures that need to be considered when choosing or designing a particular interconnection strategy for a PDMS.

2. TERMINOLOGY AND NOTATION

We provide a set of basic definitions from graph and group theory to facilitate discussions in the sections that follow.

DEFINITION 1. A digraph $G = (V_G, E_G)$ is a set of vertices V_G and a set E_G of directed edges (or arcs) $(u, v) \in V_G \times V_G$.

Throughout this paper we will consider the graphs to be directed graphs (digraphs) because we are counting entries in the routing tables, and for each undirected edge, we have two such entries. It is thus natural to consider an undirected edge as two directed edges. Where obvious, we will avoid using the index G .

DEFINITION 2. The degree of a node $v \in V$, denoted $\delta(v)$, is the number of outgoing edges from v .

DEFINITION 3. A digraph G is strongly connected if there exists a path between any two vertices. The connectivity number of the graph G (denoted by λ) is the minimal number of nodes whose removal disconnects the graph. Then, the fault tolerance of a graph is $\lambda - 1$.

DEFINITION 4. A graph G is called optimally fault tolerant if its connectivity number is equal to the smallest degree of any of its nodes: $\lambda = \min_v \{\delta(v) \mid v \in V\}$.

Now, some basic abstract algebra definitions:

DEFINITION 5. A group $G = (X, *)$ is a set of elements X together with a binary operation $*$: $X \times X \rightarrow X$ with the following properties:

closure: $\forall x, y \in X : x * y \in X$
associativity: $\forall x, y, z \in X : x * (y * z) = (x * y) * z$
identity: $\exists e \in X \text{ s.t. } \forall x \in X : x * e = e * x = x$
inverse: $\forall x \in X, \exists y \in X \text{ s.t. } x * y = y * x = e$. (Not. $x^{-1} = y$)

By an abuse of notation, we will often write G as the set of elements of the group. We will refer to the binary operation of a generic group as multiplication.

DEFINITION 6. A subset $S \subset G$ is called a generating set of G if every element $g \in G$ can be written as the multiplication of a finite set of elements from S . S is called a minimal generating set if $\forall k \in S, S \setminus \{k\}$ is not a generating set.

A generating set S is symmetric if it is closed under inverses: $s \in S \Rightarrow s^{-1} \in S$.

Finally, the definition of a Cayley graph (Group graph):

DEFINITION 7. A Cayley graph $\text{Cay}(G, S)$ given by a group G and a subset $S \subset G$ is the graph whose vertices are the elements of the group and there exists an arc between any two vertices u, v if and only if there exists an element $s \in S$ such that $v = u * s$. Where useful, we will label this arc with the element s .

3. RELATED WORK

Assessing the quality of P2P networks is a problem that has received considerable attention, mostly in comparing some new proposal with an existing one. Virtually all newly introduced P2P networks have such a characterization, showing improvement against previous solutions.

Still, it remains difficult for a system designer to choose among the various structures. One solution is, of course, to take all the networks, or as many as possible, and run them with the same data to see which one performs best according to some specific metric of interest. A recent such approach is that of [21]. The problem with such a test is that one has to implement, or at least obtain, the different networks and run them with the same input. This option is of little use to system designers as they need to make a decision before implementation.

Another solution is to provide a general model that unites all the available architectures. One of the first such approaches comes from the developers of Chord: In [9], they proposed a unified API for structured P2P overlays. Their work divides a P2P application into tiers, where tier 0 is limited to the basic key-based routing API (KBR); tier 1 implements abstractions such as DHT, multicast or DOLR (Decentralized Object Location and Routing); and eventual higher tiers, not discussed there, implement applications on top of the two basic levels. The authors argued that their set of API methods can be easily implemented on four types of structured overlay networks but did not provide further results in that direction. Another recent example in this category is [2]. Though the authors of [2] also do not present a real implementation, they take an even more ambitious approach to modeling P2P overlays by considering both structured and unstructured overlays in their work. Even without the benefit of an implementation, these works are a motivating exercise, arguing for the case of a unifying framework. The reason for which these two proposals lack an implementation is, we suspect, that they still require a fair amount of programming to launch a new system. Moreover, they may be too general to provide real feedback on the features that have a particular beneficial or detrimental effect on the performance.

In parallel to the above systems, works such as that of Ratajczak and Hellerstein [29], Qu et al. [28] or Gummadi et al. [11], successfully argue for an analysis of the underlying static structure of the P2P overlays before looking at the algorithms implemented on top of it to deal with load balancing and churn. First, [11] presents an empirical analysis based on the simulations of various types of networks and concludes that routing geometry is fundamental and that flexibility is a key parameter. They also suggest that the ring (here by ring they actually mean the chordal ring graph, i.e., a ring plus a set of chords connecting different sections of the ring) is a powerful candidate for the universal network, on top of which most routing and balancing algorithms may be implemented. Their analysis still lacks a cohesive analytical framework; even in the case of the ring, the results do not take into account that the arrangement of chords of the ring has a significant effect on the graph's properties.

The empirical results of [11] call for an analytical study to identify the graph theoretic properties of the networks. Such a study was initiated in [29] and independently extended in [28]. Both works identify Cayley graphs as the common

underlying structure of the most popular P2P overlays. Additionally, they suggest the possibility of defining new networks based on so-far unexplored Cayley graphs, such as the *star graph* or the *pancake graph*. A useful continuation is the implementation of these proposals, and, maybe more importantly, the identification of significant measures to differentiate existing P2P overlays. With regards to this last aspect, they observe that all structured P2P networks are very similar in terms of vertex and edge transitivity, hierarchy, fault tolerance and connectivity, and hamiltonicity. We are thus left with the impression that, for instance, the hypercube, which is substantially similar in all these metrics with the Chord ring, has been somehow miss-favored. We will show in this paper that this is not the case.

Finally, in this work, we will make multiple references to results in graph theory, parallel algorithms and architectures. Where necessary, explicit citations will be used, but otherwise, the reader is directed to a graduate level textbook such as [20].

4. STRUCTURED OVERLAYS AS CAYLEY GRAPHS

We begin by re-examining the groups and generating sets that define some of the most popular P2P structured overlays and then continue by observing their common properties. In this sense, our approach differs from existing works in that we do not only try to create a new overlay based on Cayley graph properties, but also discover Cayley graph properties from existing overlays. This will allow us to identify a sub-class of Cayley graphs that have a good potential as P2P overlays, based on the experience accumulated in the field over the past half decade.

4.1 Cayley graph definitions

We start with the definition of one of the most popular P2P overlays, the Chord:

PROPOSITION 1. *Consider G to be the group \mathbf{Z}_{2^m} (the set of positive integers smaller than 2^m) with binary operation $+(\text{mod } 2^m)$ and generating set $S = \{2^i, i = 0, \dots, m-1\}$. Then $\text{Cay}(G, S)$ is a chordal ring as in the Chord [33] overlay.*

Observe that if we close the generating set to inverses by addition of the inverse of each generator to S , then $\text{Cay}(G, S)$ is an undirected chordal ring as in the SkipNet [12] overlay.

A small change to the definition of the Cayley graph defining the Chord gives us the hypercube:

PROPOSITION 2. *Consider G to be the group \mathbf{Z}_2^n (the set of binary strings of length n) with binary operation \oplus (component wise addition modulo 2) and generating set $S = \{1_i, i = 1, \dots, n\}$ (the set of elements with exactly one 1). Then $\text{Cay}(G, S)$ is a hypercube.*

In this case, the set S is closed to inverses (every element is its own inverse) and consequently the hypercube is undirected.

We continue the modifications and from the definition of the hypercube we define the d -torus that forms the CAN overlay:

PROPOSITION 3. *Consider G to be the group $\mathbf{Z}_2^{d \cdot c}$. Here, c is the maximum number of times an area can be split and*

d is the dimensionality of the CAN overlay (usually taken to be 2). Note how this can be viewed either as $\mathbf{Z}_{2^c}^d$, or as $\mathbf{Z}_{2^d}^c$ as in CAN. It is simpler to use the $\mathbf{Z}_{2^c}^d$ version with binary operation \oplus (component wise addition 2^c) and generating set $S = \{1_i, i = 1 \dots d\} \cup \{-1_i, i = 1 \dots d\}$. Then, provided a local relabeling of nodes is performed by changing the basis from c to d , the Cayley graph of this group is precisely the CAN structured overlay graph.

This definition improves upon earlier ones in [28, 29] because it eliminates artificial distinctions between the allowable number of splits in each dimension. Previous definitions also confused the dimensionality of CAN with the dimensionality of the identifier space (note that CAN increases its identifiers based on the number of splits, not the dimensionality, which is fixed apriori). We go on to more complicated Cayley graphs: the cube-connected cycles ($\text{CCC}(n)$) and the Butterfly ($\text{BF}(n)$) networks. For the first one, we have the Cycloid [32] P2P overlay network, while for the second Viceroy [25] is frequently cited as an example. Both have $O(1)$ routing tables and a logarithmic size network diameter. Their Cayley graphs are very related: They are defined on the same group and only have different generating sets, as shown in the next proposition.

PROPOSITION 4. *Consider G to be the group formed by the cartesian product $\mathbf{Z}_n \times \mathbf{Z}_n^\ell$ with binary operation $*$ given by: $(l, x) * (l', x') = (l + l'(\text{mod } n), x \oplus \sigma^l(x'))$, where σ^ℓ is a circular permutation of length ℓ . Then, considering the generating set $S = \{(1, 00\dots 0), (0, 10\dots 0), (-1, 00\dots 0)\}$, we have $\text{Cay}(G, S)$ is $\text{CCC}(n)$.*

Using the same group, consider $S' = \{(1, 00\dots 0), (1, 10\dots 0), (-1, 00\dots 0), (-1, 10\dots 0)\}$. Then we have $\text{Cay}(G, S')$ is $\text{BF}(n)$.

However, as shown in Figure 4, the underlying network used in Viceroy is not the canonical definition of Butterfly. We will denote the Viceroy network as $\mathcal{VB}\mathcal{F}(n)$ to differentiate it from the classic understanding of a Butterfly network. $\mathcal{VB}\mathcal{F}(n)$ is defined by the following proposition:

PROPOSITION 5. *Consider G to be the monoid formed by the cartesian product $\mathbf{Z}_n \times \mathbf{Z}_{2^n}$ with binary operation $*$ given by: $(l, x) * (l', x') = (l + l'(\text{mod } n), x + x' \cdot 2^l(\text{mod } 2^n))$. Then, considering the generating set $S = \{(1, 0), (1, 1)\}$, we have that $\text{Cay}(G, S)$ is $\mathcal{VB}\mathcal{F}(n)$.*

$\mathcal{VB}\mathcal{F}(n)$ is based on a monoid because the new operation precludes the existence of an inverse (a monoid is just a group without the inverse property). Despite this, our analytical and experimental framework continues to apply to the $\mathcal{VB}\mathcal{F}(n)$ and we will analyze it together with all the other networks.

We should also note the parallels between the Hypercube - Chord pair and the Butterfly - Viceroy pair. They are both based on the same group and generating set, and the only difference is the binary operation used: In two cases it is bit-wise (Hypercube and Butterfly) and in the other two it is regular addition modulo N (Chord and Viceroy). The fact that the first two are not proven to be successful P2P networks while the later two are, should tell us something about the requirements on the structure of the Cayley graphs that represent them.

Finally, we also analyze the star graph, which has been suggested as a possible candidate for P2P overlays in [28]

Table 1: P2P networks and their respective Cayley graph parameters, where \oplus is component wise addition, \circ is permutation composition, $\sigma^l(x)$ is a circular permutation of length l and negation in \pm represents the inverse with respect to the group's operation.

Cayley Group and generating set			
Group set (Cardinality)	Group operation	Generators	Overlay
$\mathbf{Z}_2^n (2^n)$	$\oplus \text{ mod } 2$	$\{1_i, i = 1, \dots, n\}$	Hypercube
$\mathbf{Z}_{2^n} (2^n)$	$+ \text{ mod } 2^n$	$\{2^i, i = 0, \dots, n-1\}$	Chord
$\mathbf{Z}_{2^n} (2^n)$	$+ \text{ mod } 2^n$	$\{\pm 2^i, i = 0, \dots, n-1\}$	SkipNet
$\mathbf{Z}_2^{d \cdot c} (2^{d \cdot c})$	$\oplus \text{ mod } 2^c$	$\{\pm 1_i, i = 1 \dots d\}$	CAN
$\mathbf{Z}_n \times \mathbf{Z}_2^n (n2^n)$	$(l, x) * (l', x') = (l + l'(\text{mod } n), x \oplus \sigma^l(x'))$	$\{(1, 00\dots 0), (0, 10\dots 0)\}$	Cycloid
$\mathbf{Z}_n \times \mathbf{Z}_2^n (n2^n)$	$(l, x) * (l', x') = (l + l'(\text{mod } n), x \oplus \sigma^l(x'))$	$\{(1, 00\dots 0), (1, 10\dots 0)\}$	Butterfly
$\mathbf{Z}_n \times \mathbf{Z}_{2^n} (n2^n)$	$(l, x) * (l', x') = (l + l'(\text{mod } n), x + x' \cdot 2^l (\text{mod } 2^n))$	$\{(1, 0), (1, 1)\}$	Viceroy
$\mathbf{S}_n (n!)$	\circ	$\{(1, i), 1 < i \leq n\}$	Star

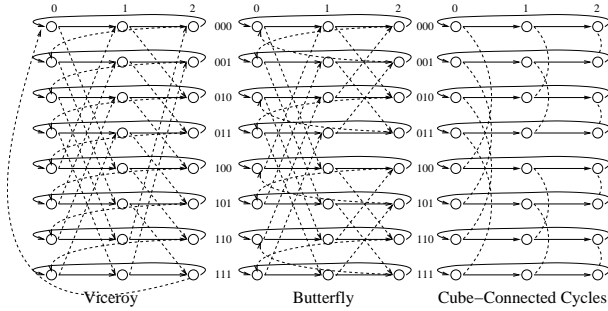


Figure 4: The three fixed-degree overlay networks

but which has not yet been analyzed from the PDMS perspective.

PROPOSITION 6. Consider G to be S_n , the group of permutations of n symbols, with permutation composition as its binary operation. Then, considering the generating set $\{(1, i), 1 < i \leq n\}$, we have $\text{Cay}(G, S)$ is the Star graph.

Many other Cayley graphs can be defined, but for space considerations we had to limit the presentation to the above, which we have summarized in Table 1. One significant omission is the Pancake graph. In this case, our analysis, as well as our experimental results were similar to those of the Star network, hence the decision to remove it from the current presentation.

5. INITIAL ASSESSMENT OF EXISTING CAYLEY GRAPH OVERLAYS

A thorough analysis cannot be presented without first answering some simple questions: How large is the routing table? Is the network able to inherently support load balancing? Is it *optimally* fault tolerant? We briefly answer these questions here.

5.1 Node degree

The size of the routing table in a PDMS node translates in the maximal degree of a graph's nodes. For a designer, this may be imposed by external factors. In mobile ad-hoc networks for instance, implementations on bluetooth devices are limited to seven simultaneous connections. Consequently, in order to ensure the physical limitations are not overstepped, the designer may not be able to choose a network with a variable number of neighbors per node. Consequently, we will avoid stating that a particular number is

good or bad, but rather classify the networks in two types: fixed degree ($\mathcal{BF}(n), \mathcal{VBF}(n), \mathcal{CCC}(n)$) and variable degree (Hypercube, Chord, SkipNet, Star).

Among networks with variable degree, the Star network is the best as it has the lowest node degree among networks with the same diameter and number of peers. For this reason, it has been previously suggested [28] as a potentially better P2P overlay.

5.2 Vertex transitivity

Intuitively, a graph G is vertex transitive if the any node has the same view of the graph. Formally: $\forall x, y \in V, \exists$ an automorphism $\Phi : V \rightarrow V$ such that $\Phi(x) = y$.

This is exactly the definition of a "pure" P2P network: All nodes view the network in exactly the same way, and none is more loaded, in any way, than another. Cayley graphs are the largest class of vertex transitive graphs, and because of this, provide the most opportunities for network design.

5.3 Edge transitivity

As in the case of nodes, we are equally interested that no edges be more loaded than others. Graph theory suggests edge-transitivity:

DEFINITION 8. A graph $G = (V, E)$ is arc-transitive if $\forall (u, v), (x, y) \in E, \exists \phi : V \rightarrow V$, an automorphism of G , such that $\phi(u) = x$ and $\phi(v) = y$. If we consider (u, v) and (x, y) without direction, then we say that G is edge-transitive.

Among the most popular P2P structures, there exist some that are not arc- or even edge-transitive. In particular, the chordal ring and $\mathcal{CCC}(n)$. It is easy to see why this is so: In both of them, we clearly have two types of edges – in one case we have the ones forming the ring versus the ones defining the chords, while in the other we have those defining the cycles and those forming the cube structure.

OBS. 1. Chord, SkipNet and Cycloid's overlays are not edge-transitive while Hypercube, Star, $\mathcal{VBF}(n)$ and $\mathcal{BF}(n)$ are.

The above observation indicates that edge transitivity is not a prerequisite of good PDMS despite the apparent benefits of graphs that exhibit this property. Consequently, it does not need to be considered in the design process of a PDMS.

Table 2: New measures to quantify quality, in terms of latency, query completeness and load balancing

Target	Basic measure	New measure
latency & query completeness	Optimal fault tolerance	Diameter resilience how is the maximum distance between nodes affected by the removal of some other nodes?
		Node bisection width how easy is it to separate the network in two disjoint and almost equal subnetworks?
load balancing & scalability	edge transitivity	Edge bisection width is there a bottleneck through which all messages must pass to reach one side of the network from the other?
	vertex transitivity	Hierarchy is the graph easily partitionable such that each vertex maintains an equal amount of IDs?

5.4 Optimal fault tolerance

As stated in Definition 4, a graph is optimally fault tolerant if the minimum number of nodes that have to be removed in order to disconnect the graph is equal to the minimum degree of any of its nodes. It is tempting to qualify a PDMS as “better” if its underlying graph is *optimally* fault tolerant. This is true but insufficient. In fact, all existing P2P overlays are optimally fault tolerant. The reason for which such optimality is routine among P2P networks is that it sets too low standards. We show this analytically using the following theorem by Godsil [10]:

THEOREM 1 ([10]). *Let $S = \{s_1, \dots, s_d\}$ be a minimal generating set of a group G . Then the Cayley digraph $\text{Cay}(G, S)$ is optimally fault tolerant.*

Theorem 1 illustrates how optimality is achieved. Definition 4 is based on the equality of two parameters: the connectivity number and the minimal degree. The theorem reduces the degree of the graph to reach equality, thus giving a false impression of quality. Instead, for PDMS, we want to increase the connectivity number to make the network more resilient.

In general, PDMS assume the risk of a few nodes failing, and the optimal fault tolerance requirement fails to apply as soon as one node gets disconnected from the network. This is too restrictive and does not indicate overall network resilience. As we will see in the next section, a much more informative measure is bisection width: the number of nodes that need to fail to separate the network in two almost equal subnetworks.

6. ALTERNATIVE FAULT TOLERANCE MEASURES

The previous sections have observed that all Cayley graphs that are the basis of existing structured P2P overlays are optimally fault tolerant and vertex transitive. If we are to understand their differences better, we need to look at more advanced quality measures. Considering our targets of measuring latency, query completeness and load balancing, we extend the basic measures with diameter resilience (for latency), node bisection width (for query completeness), edge bisection width (for load balancing) and hierarchy (for scalability), as summarized in Table 2. Also, we analyze hamiltonicity (the possibility to efficiently answer range queries) and conclude that it is a significant requirement for networks to become good overlays.

To get a better picture of the correspondence between the analytical results and their practical consequences, we

present experimental results in parallel to the analysis. Detailed information and further experiments are presented in Section 7. Here, we just mention that query, data and peer failure distributions are considered uniform and network sizes range between 1024 and 5040 nodes. The query success rates are averaged over 200 executions and plotted with vertical error bounds given by the standard deviation. Maximal path lengths are plotted by taking the maximum over all runs in a particular category of interest.

6.1 Diameter resilience

We have seen in Section 5.4 that all overlays are *optimally* fault tolerant and we have concluded that the kind of optimality generally considered for graph connectivity is not sufficiently informative for the analysis of P2P overlays. One way to go deeper into the study of fault tolerance is to look at how fast and how much the diameter of the network increases as the nodes fail. This has a direct effect on the maximum latency that the user experiences in a PDMS system. We conjecture that if λ is the connectivity of the graph, then a good network will be one that can sustain the removal of up to $\lambda - 1$ nodes without changing its diameter by more than a constant factor.

We use the definition of *wide diameter* introduced by Hsu [13]:

DEFINITION 9. *Let k and λ be two integers, $k \leq \lambda$. Let G be a λ -connected graph and x and y two distinct vertices.*

The k -distance between x and y is equal to the least number l such that there exist k vertex-disjoint paths between x and y whose lengths are at most l . Denote it by $d_k(x, y)$.

The k -diameter is the maximum of the k -distances $d_k(x, y)$ taken over all distinct pairs x, y . Denote it by $d_k(G)$.

The wide-diameter is the k -diameter if $k = \lambda$. Denote it by $wdiam(G)$.

A graph is then called *strongly resilient* if $wdiam(G) = d_1(G) + O(1)$. Table 3 shows the diameter and wide diameter of every network. We observe that, with the exception of Viceroy, all networks are strongly resilient, their wide diameter differing only by a constant.

We test the effects of the wide diameter using our network simulator by looking at how the maximum path increases in the first 20% of node failures. Networks with a variable number of neighbors, such as Chord, Hypercube, CAN or SkipNet, despite their slight differences in the wide diameter, consistently show very good resilience, maintaining their original diameter. Networks with a fixed number of neighbors (*VBF*, *BF*, *CCC*) are naturally more sensitive and ordering them based on the wide diameter matches their

Table 3: Wide diameters of P2P overlays

Overlay	diam	w-diam
Hypercube	n	n
Chord	n	$\leq n + 2$
SkipNet	$\lfloor \frac{n}{2} \rfloor$	$\leq \lfloor \frac{n}{2} \rfloor + 2$
CAN	$d \cdot 2^{e-1}$	$\leq 2 + d \cdot 2^{e-1}$
Cycloid	$2n + \lfloor \frac{n}{2} \rfloor - 1$	$2n + \lfloor \frac{n}{2} \rfloor + 2$
Butterfly	$n + \lfloor \frac{n}{2} \rfloor$	$n + \lfloor \frac{n}{2} \rfloor$
Viceroy	n	$n + O(\frac{n}{2})$
Star	$\lfloor \frac{3(n-1)}{2} \rfloor$	$\lfloor \frac{3(n-1)}{2} \rfloor + 4$

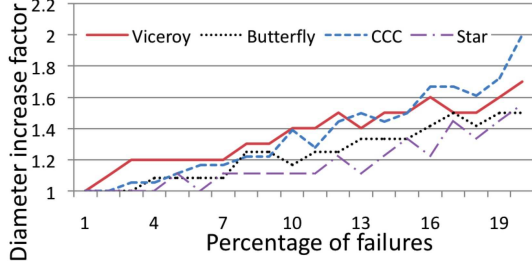


Figure 5: Maximum path increase factor in Butterfly, Viceroy, CCC and Star

ordering using experimental results. The Star network also shows an increase in the maximum route despite its wide diameter being only a constant factor greater than the regular diameter. The reason for this is that the constant factor 4 is actually significant when compared with n : The explosive size of the Star requires n to be small. For instance, $n = 10$ results in 3.6mil IDs, and in this case, the diameter is 13 and thus the constant 4 represents a 30% increase of the diameter.

Still, as shown in Figure 5, where the y-axis indicates the proportion between the maximal query path in the network with failures and the maximal path in the ideal network, the Star network increases its diameter slower than the fixed-degree networks. Viceroy, though it has the smallest diameter, has a wide diameter equal to that of Butterfly, making it jump as soon as a very small set of nodes fail. Variable-degree networks are not plotted because they show a constant factor 1.

We conclude that wide-diameter is a significant distinctive measure for fixed-degree networks, where it indicates how well the network is able to maintain the optimal diameter.

6.2 Bisection width

An orthogonal approach to extending the optimal fault tolerance is to measure the ease, in terms of number of failed nodes, with which the network fails to provide a path between two live nodes.

In the case of optimal fault tolerance, a network is considered as having “failed” as soon as a single node is disconnected from the rest of its peers by the disappearance of its direct neighbors. In P2P networks, this is obviously not the case. In general, a network contains thousands or many more nodes, and it continues to be functional when a single node is disconnected.

With this scenario in mind, we need a measure to tell us how difficult it is to disconnect the network in such a way that a significant number of peers are disconnected from an equally significant number of other peers. If this kind of

situation occurs, then each of the nodes loses connectivity to half the network, and a potentially much worse situation than the one we described in the previous paragraph arises.

The main approach to quantifying this problem is the *bisection width*. The most common definition of bisection width refers to the minimal number of *edges* that need to be removed to separate the graph into two disjoint sets of nodes of equal or almost equal size. This measure is also important, and we will consider it in the next section on load balancing. Here, since we are concerned with node failures, we prefer to use the following definition:

DEFINITION 10. Let G be a graph. A subset Ω of the nodes of G is a node bisector if G may be expressed as the disjoint union $G = \Omega_1 \cup \Omega \cup \Omega_2$, where $||\Omega_1| - |\Omega_2|| \leq 1$ and where any path from Ω_1 to Ω_2 in G must pass through Ω .

Then, we will call the node bisection width the value $\nu(G) = \min\{|\Omega| : \Omega \text{ is a node bisector}\}$

This is similar to the definition in [6], but more general, in the sense that it does not restrict the absolute size of the resulting partitions, but correlates them. In general, there is little literature on the subject. It is perhaps worth mentioning that for Cayley graphs defined on commutative groups with symmetric generating sets of size r , $\nu(G) = O(|G|^{1-\frac{1}{r}})$ [6]. This applies to Chord, Hypercube, CAN and Star, but not to \mathcal{VBF} , \mathcal{BF} or CCC . In the absence of more general results, we will be looking in more detail at our specific networks of interest.

From Table 4, we can clearly see that Chord and SkipNet perform optimally. They do not admit any set of nodes to split them into two subnetworks of almost equal size. For Star, we only know the bounds of the node bisection width, but they are enough to show that the number of nodes that need to fail is proportional to the total number of nodes in the network.

Figure 6 shows experimental results confirming the ordering of the overlays based on node bisection width as a measure of query success rate. Error bars are shown selectively to avoid clutter.

Certainly, bisection width is not the only factor in query success rate, and this may be seen if we look at SkipNet and Chord: They have the same bisection width, but SkipNet outperforms Chord. This is because SkipNet has bi-directed edges, thus effectively having twice the amount of neighbors of Chord. To see this, we have also implemented a variant of Chord, with the same number of neighbors as SkipNet, which we have called *Chord2n*. The neighbors of

Table 4: Bisection width of P2P network overlays

Overlay	Bisection width	
	Node	Edge
Hypercube	$\frac{n!}{2(\frac{n}{2})!}$ if n is even, $\frac{2n!}{\lfloor \frac{n}{2} \rfloor! \lfloor \frac{n}{2} \rfloor!}$ if n is odd	2^{n-1}
Chord	2^n	2^n
SkipNet	2^n	2^n
CAN (2D)	2^{c+1}	$2^c + 2$
Butterfly	2^n	2^n
Viceroy	$2^{n+1} - 2$	$2^{n+1} - 2$
CCC	2^{n-1}	2^{n-1}
Star	lower bound: $\Omega((n-1)!)$ upper bound: $O(n!/\sqrt{\log n})$	$\frac{n!}{4} + O(n-1)!$

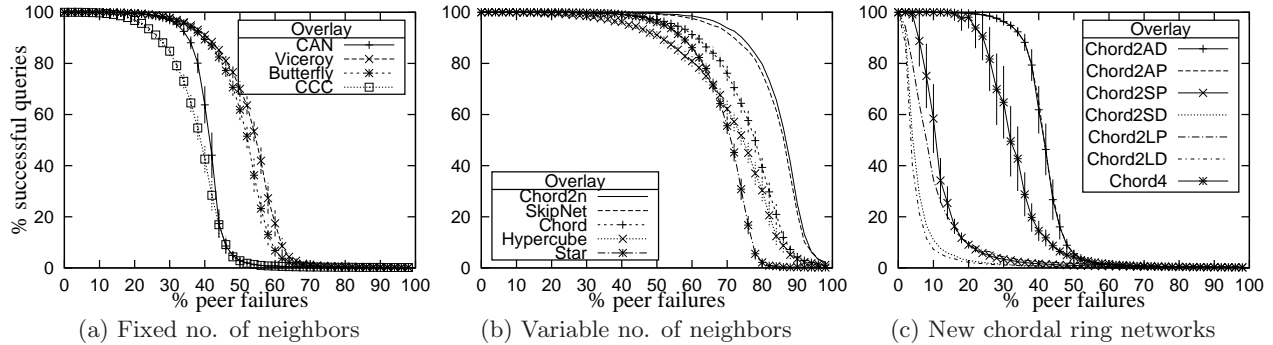


Figure 6: Query success rates for different networks is correlated with node bisection width

Table 5: New chordal ring overlays

Overlay	Generators	Node bisection width
Chord2AD	$\{1, 2^{\lfloor \frac{n}{2} \rfloor}, 2^n - 1, 2^n - 2^{\lfloor \frac{n}{2} \rfloor}\}$	$2 \cdot 2^{\lfloor \frac{n}{2} \rfloor}$
Chord2AP	$\{1, 2^{\lfloor \frac{n}{2} \rfloor - 1}, 2^n - 1, 2^n - 2^{\lfloor \frac{n}{2} \rfloor}\} \cup \{1\}$	$2 \cdot 2^{\lfloor \frac{n}{2} \rfloor}$
Chord2SD	$\{1, 2, 2^n - 1, 2^n - 2\}$	4
Chord2SP	$\{1, 2^1 + 1, 2^n - 1, 2^n - 2^1\}$	4
Chord2LD	$\{1, 2^{n-1}, 2^n - 1\}$	4
Chord2LP	$\{1, 2^{n-1} + 1, 2^n - 1, 2^n - 2^{n-1} - 1\}$	4
Chord4	$\{1, 2, 2^{n-2}, 2^{n-1}\}$	$2 \cdot 4^{\lfloor \frac{n}{2} \rfloor}$

$Chord2n$ are $\{2^i, 2^i + 1 | 0 \leq i < n\}$ hops away from the current node. As can be seen, $Chord2n$ slightly outperforms SkipNet, showing that SkipNet does better than Chord simply because of its higher number of neighbors.

We further prove our claim that node bisection width is a significant quality measure for the overlay underlying a PDMS by implementing new networks with the same degree based on the chordal ring. Table 5 indicates their generators and their node bisection widths, while Figure 6c shows the corresponding simulation results. The new chordal ring overlays are representative of the possible chordal ring networks because they take as generators numbers that are either prime with respect to the number of nodes (P), either a divisor of it (D). They are small (S), average (A) or large (L) (hence their names: $Chord2AD$ is a chordal ring with an average divisor as generator, or $Chord2SP$ - a chordal ring with a small prime as generator, etc.). $Chord4$ uses a combination of small and large divisors. The networks using generators that are prime with respect to the number of nodes create hamiltonian cycles intertwined with each other while those with generators that divide the number of nodes create vertex-independent cycles. Comparing Table 5 with Figure 6c, we see that chordal rings whose generators are such that the bisection width is not constant, have a much higher query success rate than those with constant bisection width.

6.2.1 Edge bisection width

As mentioned in the previous section, *node* bisection width is not the most common and in fact most works referring to “bisection width” actually deal with *edge* bisection width. We can then ask ourselves if, from the PDMS point of view, edge bisection width is an equally important parameter to take into account.

In P2P networks, because we are dealing with overlays and not low-level physical communications, it is generally assumed that a connection does not fail unless one of the

nodes fails. This is a fair assumption because the Transport Level of the IP Protocol Suite deals with the physical interconnection and unless there is a major disruption at one of the Internet’s backbone providers, it is always possible to find a route between two peers, such that query completeness will not be affected. In this sense, the edge bisection width is not a significant measure to look at.

However, edge bisection width has also been considered as a measure of the potential bottleneck that can arise in a network. If we look at the Butterfly or CCC graph in Figure 4, it is easy to see that all messages that pass between the peers in the upper half and the peers in the lower half have to pass by the edges that are connected to the nodes on level 0. This happens regardless of the size of the network. It is fair to assume that those edges will be, on average, more loaded than others.

Our experimental results do not confirm this assumption. Figure 7 shows the distribution of passes over the different edges. In these experiments, queries were uniformly distributed among nodes. To plot Figure 7, we counted the number of times each edge type was used. By edge type, we mean an edge of a particular length in the case of Chord, SkipNet and Hypercube, a cycle or diagonal edge in the case of Butterfly, CCC or Viceroy, a particular direction in the case of the two dimensional CAN, or a particular transposition in the case of the Star network. Experimental results show that the distribution over edge types is uniform for Chord and Hypercube. For SkipNet we see a peak at the edge that forms the diagonal (2^{n-1}) and this is only because it is its own inverse, and so it is counted twice. The other edges that are used more are those of very small length (1 and 2). They seem to “take” usages from the next larger edge (4). This is because with undirected edges, the routing decides to jump slightly further and then make a small step back rather than always making a step forward (e.g., instead of reaching 7 via 4, 2 and 1, it goes via 8 and -1).

For Butterfly and CCC, we see a clear difference between the cycle edges and the diagonal edges. The more frequent use of cycle edges in CCC is due to the fact that they represent the only way for the query to go from one level to another, as opposed to Butterfly, where diagonal edges are truly diagonal and cross levels. Overall, the expectation that the diagonal edges at level zero be used more than the rest is not met. Viceroy is more balanced, but it shows an increased usage of diagonal edges at level zero, further convincing us that edge bisection width is not a reliable way to characterize imbalance in edge usage.

Finally, the two-dimensional CAN and Star networks show almost uniform patterns as well. The slight skew in CAN

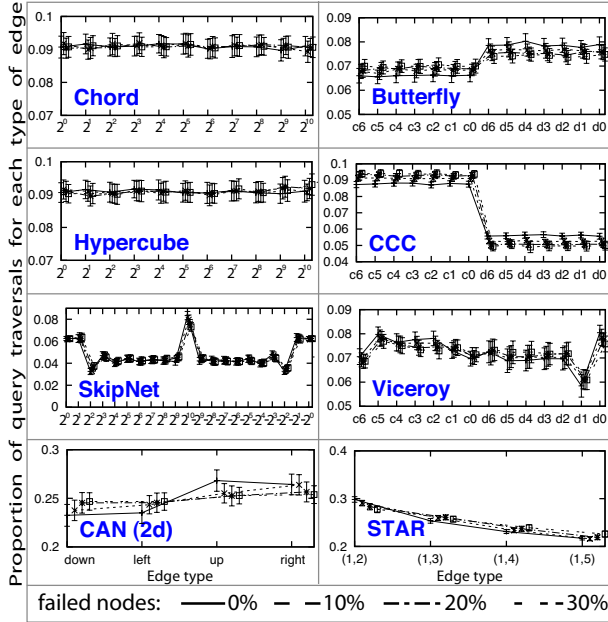


Figure 7: Distribution of queries over edge types does not show that edge bisection width has an adverse effect on the load balancing of bandwidth usage.

is due to the fact that there is an implicit ordering in the neighbors, so when a node that is half-way across the torus is searched, the “forward” edges are preferred (*up* and *right* if we consider a usual 2d plot). This imbalance disappears when failures occur because queries are given less routing choices and thus get less opportunities to favor “forward” routes.

6.3 Hierarchy

It has been observed in [28] that many overlays are *hierarchical* Cayley graphs. This property has implications both in terms of optimality of fault tolerance and definition of routing methods based on recursiveness. Recently, Loo et al. [23] have taken advantage of this property in defining a recursive declarative language for interconnection networks, though without mentioning it explicitly.

Let us first formally define two notions [3]:

DEFINITION 11. A Cayley digraph $Cay(G, S)$ is strongly hierarchical if S is a minimal generating set.

An ordered set $S = \{s_1, s_2, \dots, s_d\}$ is said to be quasi-minimal if $\forall 2 \leq i \leq d, s_i$ does not belong to the group generated by $\{s_1, s_2, \dots, s_{i-1}\}$.

A Cayley digraph $Cay(G, S)$ is hierarchical if there exists an ordering \tilde{S} of S such that \tilde{S} is a quasi-minimal set.

By looking at their generating sets, it is easy to observe that the Hypercube, $\mathcal{VBF}(n)$, $\mathcal{BF}(n)$ and $\mathcal{CCC}(n)$ networks are strongly hierarchical, and consequently, hierarchical. For Chord, we propose the following:

PROPOSITION 7. The Cayley digraph defining the Chord P2P overlay is hierarchical.

PROOF. To see this, it is sufficient to give a descending order to the generating set used in Proposition 1. Then, each prefix subset $S' \subset S$ will only generate those elements

that are a multiple of the smallest element in the subset, thus “jumping over” the elements in $S \setminus S'$. \square

In terms of node load balancing, a hierarchical network allows a node to maintain a known subset of IDs. More importantly, when it needs to insert a new peer, each node knows exactly what ID to give it: the one obtained by applying the next available largest generator (according to the given ordering of the generator set) to its own ID.

This provides the load balancing mechanism in PDMS that do not fix the IDs of nodes prior to the joining process, like BATON, for instance. If we regard this via the transformation presented in Section 1.1, BATON maintains its balanced structure by taking advantage of the hierarchical properties of Chord. However, hamiltonicity is required in this process, and we analyze it in the next section.

6.4 Hamiltonicity

After having extended the basic quality measures from Section 5 into more advanced ones in Sections 6.1-6.3, we look at *hamiltonicity* – the existence of a cycle passing through every node exactly once.

Why is this important? We identify three reasons for it:

1. **Default routing:** As shown empirically in [11], adjacency links provide significant help in terms of query success.
2. **Range queries:** To efficiently answer range queries, a node must be able to identify quickly which of its peers maintains the sub-ranges that it does not have.
3. **ID management:** In virtually all cases, the number of peers is strictly smaller than the number of available IDs. Then, each peer assigns itself a subset of IDs and this can be done efficiently if there exists an ordering of the IDs.

In general, Cayley graphs are not known to always have a hamiltonian cycle. However, all Cayley graphs defining our networks of interest are hamiltonian.

Before continuing our discussion of the three points just mentioned, we need to observe that every complete graph has a hamiltonian cycle. This is obviously true since any ordering of the nodes in a complete graph (i.e., a graph that has an edge between any two nodes) is a hamiltonian cycle. We make this trivial observation to point out that though a hamiltonian cycle is helpful in P2P overlays, the fact that a structure contains a hamiltonian cycle does not make it necessarily a good P2P structure. We need to show that there is such a cycle in a network with small routing tables. The following theorem, from Pak and Radoičić [26] helps:

THEOREM 2 ([26]). Every finite group G of size $|G| \geq 3$ has a generating set S of size $|S| \leq \log_2 |G|$, such that $Cay(G, S)$ contains a hamiltonian cycle.

For P2P overlays, this theorem gives us a measure of the “quality” of a network in terms of its hamiltonicity: if it needs more than a logarithmic number of neighbors, it is not optimal. In this sense, all the structured overlays we study are *optimally hamiltonian*. Among them, we observe that only Chord and SkipNet maintain this cycle in a very explicit manner while the rest, though they each possess a hamiltonian cycle, do not make use of it explicitly. Hypercube and CAN have hamiltonian cycles given by a space filling curve while $\mathcal{CCC}(n)$ builds its hamiltonian cycle by

a straightforward extension from the hypercube. For the $\mathcal{VB}\mathcal{F}(n)$, we have the consecutive traversal of horizontal levels in Figure 4 while for $\mathcal{B}\mathcal{F}(n)$ the hamiltonicity is shown in [5] for the directed version, and consequently, for the undirected version.

With respect to range queries, the existence of the hamiltonian cycle is useful only if the graph is undirected. This is why, for instance, SkipNet does naturally support them while Chord does not. Cycloid and Viceroy are both considered to have undirected graph and could potentially support range queries, though this is not mentioned in the respective papers. CAN is the most flexible in this regard because it supports also multi-dimensional range queries by considering cycles on each of its dimensions.

Apart from the existence of a hamiltonian path, it is equally important to have an easy way for each peer to know which other peer is to its “left” or “right”. This is easy in SkipNet: The peer with the largest ID smaller than the current ID is to the left and the one with the smallest ID greater than the current one is to the right.

It is less obvious in Viceroy, which is why the authors of [25] have introduced additional IDs (in a range between 0 and 1) to arrange the peers, even though, ultimately, the ordering they obtain matches perfectly the natural hamiltonian cycle of the $\mathcal{VB}\mathcal{F}(n)$ that we have indicated above. In the case of $CCC(n)$ the order of traversal of the cycles that bind together the cube depends on the corner of the hypercube that the particular cycle occupies and also on how the space filling curve was started. A similar situation occurs for $\mathcal{B}\mathcal{F}(n)$ [5], which results in the requirement of additional information to be managed in the network to make sure that each node is able to compute its correct position on the cycle.

Finally, the Star is a special case: though it has a hamiltonian cycle, it takes $O(n!)$ for a node to obtain its own position on the cycle [18]. This forces a node to maintain extra adjacency links.

The third argument we have identified for a network to have a hamiltonian underlying structure is ID management. In Chord and SkipNet, a node maintains all documents with IDs between itself and the next larger neighbor. This *between* implies an order in the ID space, and consequently, since nodes are also part of the ID space, an order between all nodes - a hamiltonian cycle. This, and the hierarchy presented above, are prerequisites for the ability of the network to efficiently handle an ID space larger than the number of nodes.

7. EXPERIMENTAL RESULTS

For our experiments, we have designed, implemented and executed a generic framework based on Cayley graphs. The system’s architecture is shown in Figure 8. The idea is to provide the user with a plug-and-play system that allows a rapid implementation of a simulator for any structured overlay based on a Cayley graph.

7.1 Simulation framework

All components of the simulation framework were implemented using the Java 1.5 programming language, taking extensive advantage of its generic types. The four main components of the framework, as shown in Figure 8, are:

Abstract group: Defines a generic Cayley graph, from which we derive the plug-in classes that implement different

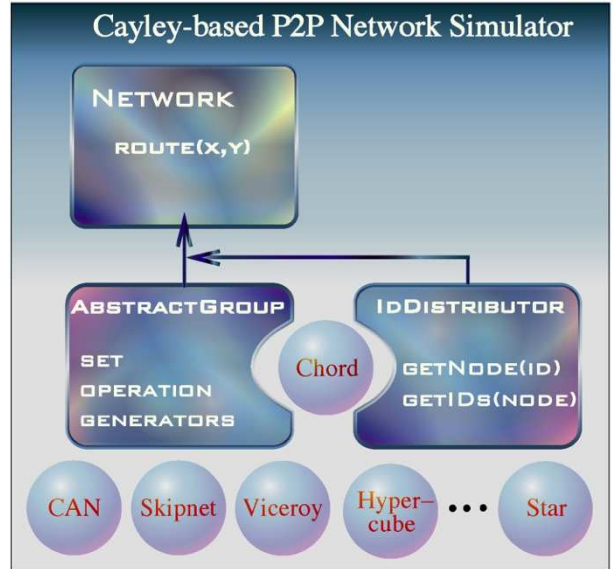


Figure 8: Architecture of Cayley-based implementation of common P2P networks simulator overlays specifications.

Plug-in modules: The modules, depicted as spheres, are classes derived from `AbstractGroup` that specify the actual group and generator set being used, as defined for instance in Table 1.

IdDistributor: When testing sparse networks, the `IdDistributor` assigns subsets of the ID space to different nodes.

Network: The network simulator launches test queries which execute according to the group and generator set defined in the plug-in module currently used. It captures success rates as well as edge and generator usages.

As different networks have different identifier types (an integer for Chord, a (level, id)-pair for Butterfly, an array of integers for CAN, etc.), each network specification is parametrized with its identifier type. The requirements for these types are restricted to the existence of a way to iterate through the set of identifiers and the possibility to pick one at random.

To test a new structured overlay, the user must only derive a new class either from `AbstractGroup` or, if the new overlay is similar to one that is already implemented, from an existing descendent of the `AbstractGroup`. Similarly, a new identifier type may be necessary, or an existing one can be used. Apart from this, the user does not need to modify the network simulator, but only instruct the framework to instantiate the correct class when given some command line parameter.

We have tested the system on a Sun cluster running the Sun Grid Engine 5.3p6 with 49 nodes using 2 Opteron 2.2GHz CPUs and 2GB RAM, and 42 nodes using 2 P4 2.8GHz CPUs and 1GB RAM. Each network simulator was instantiated 200 times for failure rates ranging between 0% and 98% with a step of 2%. Each time, 1000 random queries were generated. In total, we simulated 25 networks for a total of 250 million query tests. Results were aggregated over the 200 runs for each failure rate and for each network, by averaging when we looked at success rates and by taking the maximum when we look at the maximal distance between nodes. Network sizes ranged between 1024 and 5040 nodes.

7.2 Routing

The `AbstractGroup` class implements a default routing strategy, which takes into account only the generic group elements that are available at this level of the architecture. This generic routing method provides us with an important baseline for our experiments: it shows the shortest possible route in the presence of different levels of failures. Compared against this baseline, implementations of the real protocols of existing PDMS will show if the PDMS takes full advantage of its underlying structure.

7.2.1 Generic routing

Viewed as graphs, all structured P2P networks are endowed with an algorithm to find the shortest path between any two nodes, provided that each node has an image of the graph that forms the network. This knowledge is represented in our framework by maintaining only three items: the group used, meaning both the set of elements and the operation, and the set of generators.

We have implemented the generic routing based on Dijkstra’s algorithm in order to measure two things: how each overlay is able to sustain node failures without recurring to additional recovery methods, and how close actual P2P networks come to this optimum.

7.2.2 Overlay-specific routing

Each overlay can specify its own routing by overriding a single method in the `AbstractGroup` class. We have implemented the specific routing methods of Chord, HyperCuP, Viceroy and Cycloid.

The Chord routing chooses at each step the live neighbor closest to the target destination, without overstepping it. This is easily implemented in our framework in only 20 lines of Java code. Similarly, the HyperCuP routing tries to sequentially correct every bit, and each node forwards a query greedily to the live neighbor that can correct the left-most bit.

In Viceroy, a query first attempts to go to level zero, then traverses a tree in the network and finally routes along a cycle. In Cycloid, the query is routed by a sequence of bit-corrections like in the case of the Hypercube, but intermixed with the traversal of cycles at each corner of the hypercube. Figure 9 shows the difference between these real routing protocols and the optimal routing. To differentiate, the former are denoted by the names of the PDMS and the latter are denoted by the names of the underlying structures. As can be seen, Chord and HyperCuP maintain the same trend in their real routings as they have in the optimal case. Viceroy and Cycloid perform much worse, mainly due to the inflexibility of the routing method, unable to reroute in the presence of failures.

These observations open an interesting direction for future research: to what extent does the underlying infrastructure allow the optimal routing to be implemented in a set of simple rules that can be applied independently at each node.

7.3 Non-Cayley overlays

In addition to the networks that are based strictly on Cayley graphs, our framework can simulate networks that are “quasi” isomorphic to a Cayley graph. We show this using BATON. We have implemented the original routing algorithm as described in [15] and used the source codes available

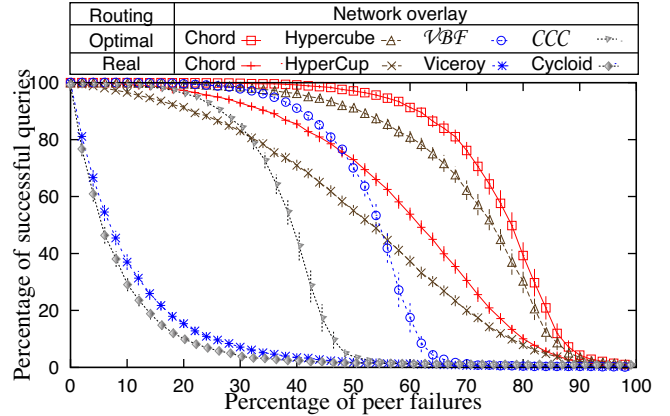


Figure 9: Difference between optimal and real routing in three network implementations

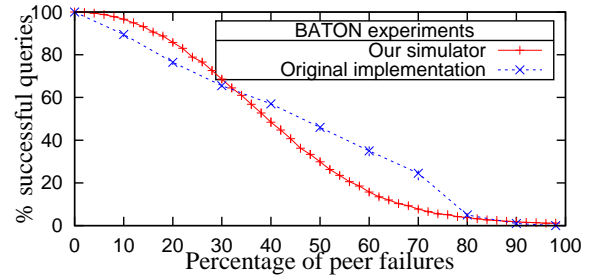


Figure 10: Experiments on the BATON network

on the Internet¹ to test our simulator against the network’s execution on PlanetLab. For the latter, we have used 8 PlanetLab nodes, each running up to 500 peers and an additional PlanetLab node to launch and collect information from the peers. Figure 10 shows the results of these experiments. The simulator is able to provide a good approximation, with an apparent underestimation of the performance when more than a third of the nodes fail. This is likely due to additional fault tolerance measures implemented later in the BATON protocol.

Other networks, like Pastry, Tapestry or those based on deBruijn graphs [24] show similarities with Cayley graphs and are the focus of current research.

8. CONCLUSIONS

We started this work with the aim of identifying a set of measures to follow when choosing or designing a structured overlay for a PDMS. Our strategy was not only to look at graph-theoretic measures, but also to learn from the experience of the past half decade by matching the proposed measures with the existing overlays to discover which of them are indeed significant. We identified four categories of measures:

1. **Always true:** *vertex transitivity, optimal fault tolerance, hierarchy:* all networks display these properties and we can conclude that these are required, but do not differentiate between the networks.
2. **External:** *node degree:* networks can have either a fixed or a variable number of neighbors. This decision

¹<http://www.comp.nus.edu.sg/~bestpeer/resource.html>

is generally taken even before the design process starts, subject to external constraints.

3. **Misleading:** *edge transitivity, edge bisection width:* though claimed to be an indication of bandwidth load balancing, our experimental results disprove this claim.
4. **Proper:** *diameter resilience, node bisection width, hamiltonicity:* these measures are indicative of a networks capacity to maintain low latency and query completeness in the presence of failures.

The *proper* measures indicated above are not guarantees of quality, but prerequisites: when designing a network, we should look for a structure that is strongly resilient and has a very high node bisection width. Hamiltonicity comes with a small print: not only should the network have a hamiltonian cycle, but also every node should be able to identify its position on this cycle and also identify its neighbors efficiently.

So which overlays may claim stardom in PDMS? The Star network had been suggested as a good candidate in [28] because it is able to maintain a low diameter and at the same time a small routing table. Our analysis shows that it also displays good performance against all measures except hamiltonicity. Yet there is no PDMS using the Star as an underlying overlay. This is an indication that 1. Hamiltonicity is important and 2. The Star may be simply too complex and counter-intuitive for its adoption in real systems.

Chord seems to be the most entitled to be the reference network. Here again, hamiltonicity plays a key role: since all these networks have a hamiltonian cycle, all of them can be arranged as a chordal ring and thus provide support for Chord's claim to fortune. As we have seen experimentally (Figure 6c), a chordal ring performs very well even with a fixed number of neighbors, reducing the need for more complex structures such as the butterfly.

Finally, in the process of analyzing all these overlays, we have created a framework by which a designer can implement and simulate easily different network structures. Within this framework, we can go from idea to simulation results within a day - a significant aid for a system designer. The simulations can either use a default routing protocol, based on Dijkstra's algorithm to identify the potential of the network, or specify their own routing protocol. While the former is a sort of upper bound on its performance, the latter provides a lower bound, given that no non-standard fault tolerance measures are used, such as caching or replication. Both of these bounds give the designer a significantly better perspective over the capability of the structured overlay to withstand failures.

Acknowledgments

The research of Mihai Lupu and Beng Chin Ooi was in part funded by ASTAR SERC Grant 072 101 0017. The authors would also like to thank Quang Hieu Vu and Cristina David for their help in running parts of the experiments and for their valuable comments regarding this work.

Repeatability assessment result

Figures 5, 6, 9 and 10 have been verified by the SIGMOD repeatability committee.

9. REFERENCES

- [1] K. Aberer. P-Grid: A Self-Organizing access structure for P2P Information Systems. In *Procs. of ICCIS*, 2001.
- [2] K. Aberer, L. O. Alima, A. Ghodsi, S. Girdzijauskas, S. Haridi, and M. Hauswirth. The Essence of P2P: A Reference Architecture for Overlay Networks. In *Proc. of P2P*, 2005.
- [3] S. B. Akers and B. Krishnamurthy. On Group Graphs and Their Fault Tolerance. *IEEE ToC*, 36(7), 1987.
- [4] M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani. The Price of Validity in Dynamic Networks. In *Proc. of SIGMOD*, 2004.
- [5] J.-C. Bermond, E. Darrot, O. Delmas, and S. Perennes. Hamilton Circuits in the Directed Wrapped Butterfly Network. *Discrete Applied Mathematics*, 84(1-3), 1998.
- [6] S. R. Blackburn. Node Bisectors of Cayley Graphs. *Theory of Computing Systems*, 29(6), 1996.
- [7] G. Cong, W. Fan, and A. Kementsietsidis. Distributed Query Evaluation with Performance Guarantees. In *Proc. of SIGMOD*, 2007.
- [8] A. Crainiceanu, P. Linga, A. Machanavajjhala, J. Gehrke, and J. Shanmugasundaram. P-Ring: An Efficient and Robust P2P Range Index Structure. In *Proc. of SIGMOD*, 2007.
- [9] F. Dabek, B. Zhao, P. Druschel, J. Kubiatowicz, and I. Stoica. Towards a Common API for Structured Peer-to-Peer Overlays. In *Proc. of IPTPS*, 2003.
- [10] C. D. Godsil. Connectivity of Minimal Cayley Graphs. *Archiv der Mathematik*, 37(1), 1981.
- [11] K. Gummadi, R. Gummadi, S. Ratnasamy, S. Shenker, and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. In *Proc. of SIGCOMM*, 2003.
- [12] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman. SkipNet: A Scalable Overlay Network with Practical Locality Properties. In *Proc. of USITS*, 2003.
- [13] D. F. Hsu. On Container Width and Length in Graphs, Groups and Networks. *IEICE TFECCS*, E77-A(4), 1994.
- [14] R. Huebsch, B. Chun, J. M. Hellerstein, B. T. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. R. Yumerefendi. The Architecture of PIER: an Internet-Scale Query Processor. In *Proc. of CIDR*, 2005.
- [15] H. V. Jagadish, B. C. Ooi, and Q. H. Vu. BATON: A Balanced Tree Structure for P2P Networks. In *Proc. of VLDB*, 2005.
- [16] H. V. Jagadish, B.C. Ooi, K.L. Tan, Q.H. Vu, and R. Zhang. Speeding up Search in Peer-to-Peer Networks with a Multi-way Tree Structure. In *Proc. of SIGMOD*, 2006.
- [17] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr. Overcast: Reliable Multicasting with an Overlay Network. In *Proc. of SOSDI*, 2000.
- [18] J.-S. Jwo, S. Lakshminarayanan, and S.K. Dhall. Embedding of Cycles and Grids in Star Graphs. In *Proc. of SPDP*, 1990.

- [19] A. Kementsietsidis, M. Arenas, and R. J. Miller. Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues. In *Proc. of SIGMOD*, 2003.
- [20] F. T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. Morgan Kaufmann Publishers, 1992.
- [21] J. Li, J. Stribling, R. Morris, M.F. Kaashoek, and T.M. Gil. A Performance vs. Cost Framework for Evaluating DHT Design Tradeoff Under Churn. In *Proc. of INFOCOM*, 2005.
- [22] P. Linga, A. Crainiceanu, J. Gehrke, and J. Shanmugasundaram. Guaranteeing Correctness and Availability in P2P Range Indices. In *Proc. of SIGMOD*, 2005.
- [23] B. T. Loo, T. Condie, M. Garofalakis, D. E. Gay, J. M. Hellerstein, P. Maniatis, R. Ramakrishnan, T. Roscoe, and I. Stoica. Declarative networking: Language, Execution and Optimization. In *Proc. of SIGMOD*, 2006.
- [24] E. K. Lua, J. Crowcroft, M. Pias, S. Ravi, and S. Lim. A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. *IEEE Comm. Surveys and Tutorials*, 7(2), 2005.
- [25] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A Scalable and Dynamic Emulation of the Butterfly. In *Proc. of SPDC*, 2002.
- [26] I. Pak and R. Radoicic. Hamiltonian paths in Cayley Graphs. Preprint, <http://www-math.mit.edu/~pak/hamcayley8.pdf>.
- [27] E. Pitoura, S. Abiteboul, D. Pfoser, G. Samaras, and M. Vazirgiannis. DBGlobe: A Service-Oriented P2P System for Global Computing. *SIGMOD Rec.*, 32(3), 2003.
- [28] C. Qu, W. Nejdl, and M. Kriesell. Cayley DHTs - A Group-Theoretic Framework for Analyzing DHTs Based on Cayley Graphs. In *Proc. of ISPA*, 2004.
- [29] D. Ratajczak and J. M. Hellerstein. Deconstructing DHTs. Technical Report IRB-TR-04-042, Intel Research Berkeley, Nov. 2003.
- [30] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content Addressable Network. In *Proc. of SIGCOMM*, 2001.
- [31] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl. *HyperCup - Hypercubes, Ontologies, and Efficient Search in Peer-to-Peer Networks*, volume 2530 of *LNCS*. Springer Berlin / Heidelberg, 2003.
- [32] H. Shen, C.-Z. Xu, and G. Chen. Cycloid: a Constant-Degree and Lookup-Efficient P2P Overlay Network. *Perform. Eval.*, 63(3), 2006.
- [33] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable P2P Lookup Service for Internet Applications. In *Proc. of SIGCOMM*, 2001.