Contention-Based Performance Evaluation of Multidimensional Range Search in Peer-to-peer Networks

Spyros Blanas
Technical University of Crete
smplanas@softnet.tuc.gr

Vasilis Samoladas
Technical University of Crete
vsam@softnet.tuc.gr

ABSTRACT

Performance evaluation of peer-to-peer search techniques has been based on simple performance metrics, such as message hop counts and total network traffic, mostly disregarding their inherent concurrent nature, where contention may arise. This paper is concerned with the effect of contention in complex P2P network search, focusing on techniques for multidimensional range search. We evaluate peer-to-peer networks derived from recently proposed works, introducing two novel metrics related to concurrency and contention, namely responsiveness and throughput. Our results highlight the impact of contention on these networks, and demonstrate that some studied networks do not scale in the presence of contention. Also, our results indicate that certain network properties believed to be desirable (e.g. uniform data distribution or peer accesses) may not be as critical as previously believed.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Modeling techniques; H.4 [**Information Systems Applications**]: Miscellaneous—*peer-to-peer networks, multidimensional search*

1. INTRODUCTION

Structured peer-to-peer (P2P) networks can organize thousands of Internet nodes efficiently, fault-tolerantly and without centralized administration, allowing large populations of users to search for information efficiently. Since the introduction of Distributed Hash Tables (DHTs), simple lookups have been supported well. As applications often require more sophisticated search (low-dimensional range search, multi-attribute range queries etc.), there have been numerous recent efforts to develop solutions for these problems. With respect to multidimensional range search, a number of techniques proposed for one-dimensional data can be extended to support low-dimensional data by mapping to one dimension through space-filling curves. Space-partitioning techniques, as well as techniques adapting hierarchical index structures to P2P networks have also been proposed. However, these techniques have not been specifically developed or extensively studied for low-dimensional range search, nor have they been evaluated comparatively. Subsequently, it is not clear what techniques are best suited for the important problem of low-dimensional search.

Only few comparative studies of P2P networks are available, which we partially attribute to the lack of established performance models for P2P network search. This is in contrast to other types of decentralized asynchronous systems, e.g., client-server systems and communication networks, whose fundamental properties are well-studied. Queueing theory [21, 22] has proved an important analytical tool for the study of such systems, but to our knowledge it has not been applied to P2P network search so far.

Evaluation of P2P networks is usually done via simulation, or by implementing and deploying onto Internet-wide experimental platforms, notably PlanetLab. Simulation, although it only offers crude measures of performance, can support P2P network evaluation over a broad range of parameters (large number of nodes, different workloads). However, the conclusions from simulation results can be misleading, unless there is a robust modeling framework within which these results can be interpreted. To our knowledge, such a framework does not exist for search in P2P networks. On the other hand, as [14] eloquently discusses, PlanetLab deployments only involve (at best) a few hundred machines; what's more, these are relatively powerful nodes with good network connectivity. This platform is not very representative of the Internet user community, which involves massive numbers of nodes, of varying processing and network capabilities. Although very valuable for studying the proper engineering of P2P networks, PlanetLab deployments may not accurately reflect the realistic performance of P2P solutions, unless PlanetLab's size (and, perhaps more crucially, its diversity) increases by an order of magnitude.

Usually P2P search performance is evaluated in terms of latency (number of network hops performed by the search) and network traffic (number of messages). For DHT lookups these metrics are satisfactory, but for more complex search problems, such as range search, they are less informative.

In DHT lookups, the last routing hop is the one which discovers the result of the query. In range search, results may be found from the early stages of the search. Moreover, users generally cannot monitor the state of the search, i.e., whether a search has terminated, and are likely to wait for results only for a limited time, or until "enough" answers arrive. Thus, for range search performance, it is important

Conference InfoScale '07 June 6-8, 2007, Suzhou, China
Copyright 2007 ACM 978-1-59593-757-5 ...$5.00.

to measure the *responsiveness* of a network, that is, the percentage of the search completed as a function of time.

Two desirable design goals for P2P networks are low network traffic and uniform data distribution. The former is thought to relate to the scalability of the network in the presence of contention. The latter is believed to induce a more uniform *peer load* distribution, avoiding "hotspot" peers in the presence of contention. However, counter to expectations, these two factors are often antagonistic; an even data distribution induces higher network traffic, as demonstrated by our experiments. We demonstrate that none of these metrics is able to capture contention-related aspects accurately.

1.1 Related work on P2P network search

Research in indexing on P2P networks started with the introduction of Distributed Hash Tables (DHTs) [25, 27, 26, 24], but soon extended to complex problems, such as multi-attribute indexing, e.g. [5, 15], nearest-neighbor and similarity indexing, e.g. [18, 13, 28], and range search in one dimension, e.g. [16, 4, 9, 20, 8] or in multiple dimensions, e.g. [17, 6, 10, 3, 30].

Peer-to-peer range search in one dimension can be done by modifying DHTs to use an order-preserving hash function [9, 20], or by fault-tolerant, distributed variants of skip lists or trees with sideways search [4, 3, 16, 8]. In multiple dimensions, there is relatively less, more recent work than for one dimension.

In principle, any network for 1-d range search can be adapted to k-d range search, at least for the important case of small k, by employing space-filling curves. Yet, there is little investigation on the merits and drawbacks of this approach. In [6], the authors propose PHT, a trie-like structure using z-order space-filling curve, layered on an underlying DHT infrastructure which is used essentially as distributed storage. Ganesan et al. [10] propose two structures, SCRAP, based on space-filling curves over a skip graph-like network, and MURK, a CAN-derived network which partitions space in a manner similar to k-d trees. Arge et al. [3] propose skip-webs, an extension of skip graphs to a class of problems admitting certain theoretical properties related to space partitioning.

Efforts to adapt hiererchical search structures, such as search trees, to P2P networks, must overcome a challenging load balancing problem: avoid overloading the higher levels of the structure. Recently, Jagadish et al. [17] proposed VBI-tree, a P2P network that can adapt many data partitioning schemes, where sideways search is employed to reduce the number of accesses to the higher levels of the hierarchy.

Although the literature on search in P2P networks has grown significantly, there are few comparative studies of different proposals. Comparative studies exist for search in unstructured networks (e.g. [29]) and file sharing (e.g. [11]) where real data is also available (e.g. [12]).

1.2 Our contributions

This paper presents an extensive comparative simulation study of the performance of three P2P networks for low-dimensional range search, from the recent literature: PGrid, MURK and VBI-tree. Our results quantify the behavior of these networks across a variety of parameters, such as network size, data skew, query result size, data clustering quality, and network throughput. Our main conclusion is that some established design principles from the area of multidimensional indexing (such as good locality and uniform space partitioning), although still relevant, do not affect the quality of P2P network search to the extent they affect indexing quality.

Our main contribution is a novel approach to P2P search performance evaluation, in which the emphasis is on contention as a network serves multiple concurrent requests. We claim that network traffic is insufficient for evaluating P2P network performance, and we propose to replace it by *maximum throughput*. Intuitively, maximum throughput is the maximum rate of queries that a P2P network can sustain without any peer becoming overloaded. An appealing feature of maximum throughput is that it only depends on the distribution of message traffic among peers, and does not require expensive computation (such as detailed event-driven simulation). Although in this paper we focus on range search, our work is straightforwardly applicable to different P2P applications.

The importance of our novel metric is validated, as we demonstrate that some of the studied networks do not scale well in the presence of contention. In particular, we show that, for certain networks, maximum throughput does not increase when the size of the P2P network exceeds a few hundred peers. Such P2P networks are not scalable; peers joining a P2P network offer resources in exchange for service, thus as a network grows, service demand, and thus, throughput, is expected to grow.

2. MODELING P2P NETWORK PERFORMANCE

2.1 P2P network processes

Generally speaking, peers communicate to execute *processes*, i.e., independent invocations of some P2P protocol. In this paper, a process is a multidimensional query posed to some peer. Processes may include requests from peers to enter or leave the network, updates in the data stored in the network, etc.

Each process p can be represented by a tree whose nodes are peers, and whose edges correspond to messages sent from parents to children. The root of the tree is the peer where a process is enacted. When a process visits a peer v, (either by originating there, or by receipt of a message), that peer processes it locally and forwards it to a number of neighbors by sending to each of them a message. These peers then become children of v in the process tree. Peers may appear in the tree multiple times.

Each node in the tree of a process p is labeled with a weight, and all weights sum to 1. Each weight corresponds to the fraction of process "work" performed at that particular step. What constitutes work, is problem-specific; in this paper, each node weight is the fraction of result tuples reported to the user. An example of a process tree is shown in Fig. 1. Note that peer P2 appears twice, but only the first visit advances the computation.

Various performance metrics related to processes can be defined by process trees, such as the following (in parentheses, the value of each metric for the process tree of Fig. 1):

Messages: The number of nodes in the tree minus one (4).
Accesses: The number of different peers in the tree (4).
Latency: The height of the tree (3).

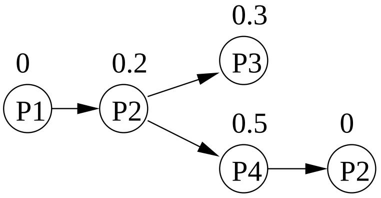


Figure 1: A sample process tree.

Responsiveness: A sequence $\{f_n\}$ where f_n is the sum of all weights of nodes at distance at most n from the root ($f = [0, 0.2, 1, 1, \ldots]$).

Of the above measures, latency and responsiveness relate to the delay of performing a process and the rest relate to the expended resources, such as network bandwidth and node processing.

2.2 Synchronous P2P network operation

Consider a *synchronous analysis*, where multiple processes operate independently in a network of peers. In this analysis, peers operate in cycles using a global clock. At each cycle, peers receive messages from other peers, where each message belongs to some process. Each peer processes all incoming messages and produces a number of outgoing messages, that is, peers have infinite processing capacity. In such a model, there is no contention for peer resources by the different processes executing concurrently, thus each process tree execution will proceed level by level. Such analysis approximates the case where the peers have relatively equal CPU speed and network connectivity, and also there are very few concurrently executing processes. This type of performance analysis is quite popular in the literature, mostly because of its simplicity. Each process execution can be studied in isolation and statistics of process-related metrics can be easily computed.

In real P2P networks, peers can have heterogeneous processing and networking capabilities [12]. Also, executing concurrent processes may create contention in peers. Thus, in reality, P2P networks become inherently asynchronous systems, whose exact behavior becomes much harder to study. In the general case, asynchronous analysis is *much* harder, most often requiring nothing short of fully detailed event-driven simulation. In what follows, we attempt to compromise the ease of synchronous analysis with the study of some performance aspects of P2P networks relating to asynchrony.

2.3 Asynchronous analysis

In this section, a peer is modeled as a service center, where customers (messages) arrive for service. A message "arrives" at node v at the instance where the transfer of the message from the sender to v commences. The "service" required for each message includes the bandwidth occupied until it is fully received and the CPU cost of processing by the P2P protocol, which results in preparing a number of new messages for transmission. An alternative definition would be to consider the "arrival" of a message at the instance where the message is fully received, and consider its "service" to include CPU processing and transmission of its resulting messages to their destinations. We use the first definition in this paper, although a similar analysis can be performed if the second definition is adopted.

Incoming messages arrive to peer j with arrival rate λ_j (we make no additional assumption on the distribution of arrival times). The service demands of individual messages are independent random variables, following the same arbitrary distribution. If peer j has a maximum service rate γ_j (depending on its processing and network capabilities), and an incoming message has service demand s, then the time required to serve this message is s/γ_j, if no other message is being served. When multiple messages arrive concurrently, various service policies are possible (e.g. First-Come-First-Served or Processor-Sharing). The time from a message's arrival to the instance where it has been fully served, is the *response time* for this message.

According to the above, in queueing theory parlance, a peer can be modeled by an GI/GI/1 queue, where the Gs denote arbitrary but independent message arrival and service demand probability distributions. This type of stochastic model has been well studied [21]. Let S be a random variable distributed according to the distribution of service demand of incoming messages. As usual, let $E[X]$ denote the expected value of random variable X. For briefness, define $S_j = S/\gamma_j$. The *traffic intensity* at peer j is defined as

$$\rho_j = \lambda_j E[S_j]. \tag{1}$$

A fundamental property of GI/GI/1 systems implies that peer j will be *stable* only if $\rho_j < 1$, else (when $\rho_j \geq 1$) service demand arrives at a rate greater or equal to service rate γ_j, which implies that the *response time* r_j of peer j to incoming messages (a random variable) will increase over time without bound, in which case the peer is *overloaded.*

2.4 Asynchronous P2P network operation

Consider a P2P network of n peers, where the service rate γ_j is given for each peer $j = 1, \ldots, n$. Also, consider a set $\mathcal{P}$ of processes, and associated popularities ϕ_p for each process $p \in \mathcal{P}$. Processes are enacted at a rate Λ, called the *process throughput.* Let $\Lambda_p = \phi_p \Lambda$ denote the rate of enactments of process p. Let $m_j(p)$ be the number of messages in process $p \in \mathcal{P}$ received by peer j. The quantities μ_j, where

$$\mu_j = \frac{\sum_{p \in \mathcal{P}} \phi_p m_j(p)}{\gamma_j}$$

constitute the *message distribution* of our network. The quantity $\gamma_j \mu_j$ is the expected number of messages received by peer j by a process enactment. Thus, $M = \sum_{j=1}^{n} \gamma_j \mu_j$ is the average per-process message traffic.

The rate of arrival of messages at peer j is

$$\lambda_j = \sum_{p \in \mathcal{P}} m_j(p)\, \Lambda_p = \Lambda \sum_{p \in \mathcal{P}} \phi_p m_j(p) = \gamma_j \mu_j \Lambda \tag{2}$$

If for some throughput Λ, every peer in the network is stable, we have for every peer j,

$$\rho_j = \lambda_j \frac{E[S]}{\gamma_j} = \mu_j \Lambda E[S] < 1$$

and we have our definition of *maximum throughput* Λ_{max}:

$$\Lambda_{\max} = \frac{1}{E[S] \cdot \max_j \mu_j} \tag{3}$$

Intuitively, $E[S] \max_j \mu_j$ is the average per-query processing time of the "most loaded" peer in the network. It is convenient to choose the units of service demand so that $E[S] = 1$, in which case $\Lambda_{\max} = 1/\max_j \mu_j$.

2.4.1 Relation to per-query network traffic

Let us consider a P2P network of n identical peers (γ_j are all equal) and an average per-query message traffic of M. Ideally, this traffic would be spread roughly equally among the peers, i.e., $\max_j \mu_j = \Theta(M/n)$, and thus, ideally, $\Lambda_{\max} = \Theta(n/M)$. For example, in a DHT lookup case, $M = \Theta(\log n)$, thus we would expect that $\Lambda_{\max}$ should scale almost linearly with the size of the network.

In more complex searches, such as range search, M may grow faster, e.g., as $\Theta(\sqrt{n})$, so we could still expect $\Lambda_{\max}$ to increase with n. Surprisingly, as our experiments will show later, this is not the case for some of the networks we study.

2.4.2 Expected hop latency

It is also desirable to study the effect of process throughput on the latency and responsiveness of a network, that is, take into account the actual delays that are caused by congestion as Λ ranges from 0 to just below $\Lambda_{\max}$. Unfortunately, this is hard to characterize accurately, without knowledge of the actual distributions of response times at individual peers.

3. P2P NETWORKS FOR MULTI-DIMENSIONAL RANGE SEARCH

In this section we present four P2P networks for multidimensional range search, derived from recent works in the literature.

3.1 Studied networks

3.1.1 PGrid

Our work extends [9], which studies one-dimensional range search on PGrid. The basic intuition behind the work in [9] is to view the set of peer IDs (which are strings of fixed size L), organized as a binary trie. Leaves of the trie correspond to actual peers, whereas internal nodes correspond to subsets of peers whose IDs have a common prefix. Let s_p be the prefix of peer p's ID, which corresponds to the path from the root of the trie to peer p. Peer p's range contains all L-bit strings with prefix s_p, which forms an interval in one dimension. Data keys are hashed to peers using an order-preserving hash function.

Two protocols are proposed in [9] for 1-d range search. The evaluation in [9] indicates that their so-called *shower* protocol improves latency significantly with only a modest increase in network traffic. In this protocol, a query is routed to an arbitrary peer whose range intersects it, and from there it is split into residual subqueries and forwarded in parallel recursively.

We adapted the network of [9] to multidimensional search, by using the z-order space-filling curve to hash multidimensional points in one dimension, and modifying the shower protocol appropriately. We refer to this network as PGrid-Z.

3.1.2 CAN and MURK

The CAN topology [25] is inherently multidimensional, with peers forming a d-dimensional toroid grid; thus, it is straightforward to associate a space partition with a CAN network. In low dimensions, two obstacles to high search performance must be overcome. First, routing in CAN requires $O(n^{1/d})$ hops, which is expensive for small d. Second, when the distribution of the data is skewed, the CAN grid may exhibit inferior data balancing. To overcome these obstacles, Ganesan et al. [10] proposed MURK. In MURK, the routing cost in low dimensions is improved by augmenting each peer's routing table with an additional set of $O(\log n)$ peer IDs, called *skip pointers* randomly selected across the network. To improve load balancing when the data is skewed, a MURK network differentiates node joins from CAN: when a new node joins, an existing node is split into two regions of equal number of data items (instead of equal volume, as in CAN). Thus, the overall space partition does not resemble a quadtrie (as in CAN), but rather that of a (possibly unbalanced) k-d tree.

In our experiments, we study MURK and also a variant of standard CAN (which we refer to as CAN-SP), augmented with randomly selected skip pointers. By contrasting them we can quantify the effect of data balancing. Also, CAN-SP supports the same set of possible space partitions as PGrid-Z, thus we can contrast these two protocols using identical partitioning of the dataset among peers.

3.1.3 VBI-tree

Recently, Jagadish et al. [17] proposed a novel technique for transferring hierarchical tree structures on P2P networks, introducing the first P2P range search technique which is not based on space partitioning or space-filling curves. Tree structures are hard to implement in P2P networks, since accessing the tree by following paths induces uneven load on nodes at the higher levels of the tree. The novel feature in VBI-tree (which inherits from BATON [16]), is that each tree node n has multiple so-called *left* and *right* pointers to nodes in the same level. By utilizing these additional pointers, query routing may avoid visiting nodes in higher levels of the tree. The experimental study in [17] demonstrates that this strategy is largely successful.

A plethora of well-known tree structures from the literature can be embedded in VBI-tree. The authors of [17] performed their experimental evaluation on an embedding of the M-tree [7], an index more suitable for high-dimensional queries. In our experiments, we embedded the Hilbert R-tree [19], which is more suitable for low-dimensional queries. We will refer to the resulting network as VBI-HR.

3.2 Dynamics of P2P networks

In reality, P2P networks are in a continuous state of flux, as peers and data enter and leave the network, concurrently with the execution of searches. These network dynamics affect search performance adversely. For example, when a peer leaves the network (esp. if it fails), search processes involving this peer have to be re-routed to other peers, delaying overall execution. Also, updates involve additional network traffic (e.g. transferring data to a peer entering the network), contending for network resources with search protocols.

In this work, we have taken a simplified approach to modeling network dynamics. We construct P2P network topologies by simulating a dynamic process of building the network. However, once we obtain a network of desired size (number of peers), we freeze the network in order to evaluate the performance of searches. This is justified on grounds of separation of concerns; the performance of a static network can be attributed to the quality of search protocols only. In particular, the non-scalability results that will be presented in our experiments are thus clearly attributed to deficiencies in the search protocols.

3.2.1 Dynamic topology construction

For VBI-HR, the manner in which nodes enter the network does not affect the final topology significantly (due to the nature of both VBI-tree and the Hilbert R-tree). For PGrid-Z, CAN-SP and MURK, this is not so, as an entering node must select an existing node, whose area will split. In the case of PGrid-Z, when a new node enters a network, it does not immediately occupy a fixed location in the topology, but instead it changes location over time, as the network reorganizes along the lines of [2].

We generate PGrid-Z, CAN-SP and MURK topologies in a similar manner: when a new node enters a network, it selects an existing node and splits its area, updating routing tables appropriately. In the case of PGrid, routing tables are then re-populated by randomly selecting appropriate nodes network-wide, in an attempt to emulate the effect of "node exchanges" (cf. [2]) over a long period of time.

The strategy via which an entering peer selects an existing peer to split, may result in topologies with different space partitions and data distributions. We considered two alternatives:

Volume-balanced selection: An entering node picks uniformly at random a point in the multidimensional search space, and then selects the node already in the network which is responsible for this point. Thus, existing nodes are selected for splitting with probability proportional to the volume of their assigned space partition. This strategy tends to equalize the volume of space assigned nodes, and can be easily implemented in practice.

Data-balanced selection: An entering node chooses uniformly at random a point *from the indexed dataset* and then selects the node responsible for this point. Thus, existing nodes are selected for splitting with probability proportional to the data they store. This strategy tends to equalize the amount of data assigned to nodes. Implementing this strategy in practice is not straightforward, as the whole indexed dataset (or its distribution) may not be accessible.

4. EXPERIMENTAL RESULTS

We now describe our extensive experimental comparison of seven P2P topologies via simulation, and present both the traditional and our new performance metrics. We do not consider heterogeneity in the network as it would not provide additional insights in our case. Thus, in all workloads, $\gamma_j = 1$ and $E[S] = 1$. Additionally, to make comparisons between CAN-SP and PGrid-Z even stronger, we constructed the corresponding networks with *identical* space partitions.

All networks were evaluated on datasets of 100,000 points, with query workloads constructed by (hyper-)cubic queries—all sides having equal length, each centered around a randomly chosen point in the dataset. In order to evaluate the effects of different point distributions, we used the following point sets:

R2 A 2-d point set on real data, obtained from [1], representing points along roads in Greece.

Hd For $d = 2, \ldots, 6$ these point sets were constructed by selecting random points close to the surface of a d-dimensional hyper-sphere. This dataset has intrinsic dimension one less than the nominal, and is hard to split evenly by quadtree-like space partitions.

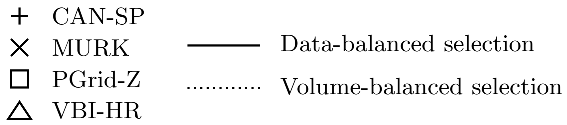


Figure 2: Legend for all graphs.

All graphs compare 7 P2P network topologies; for each of PGrid-Z, CAN-SP and MURK, we study two topologies, constructed by volume-balanced or data-balanced selection respectively. In all graphs, the curves are marked with the legend of Fig. 2. The caption of all graphs is marked with n for network size and $|q|$ for query size range.

We will first present the *responsiveness* metric, which shows that in the absence of contention responsiveness is low and remains logarithmic to network size. We shall then focus on *maximum throughput*, which reveals hitherto unquantified performance aspects of the studied P2P networks. Finally, we will demonstrate that some popular and widely used metrics, *message traffic* and *data and load balancing*, do not accurately reflect network performance in the presence of contention.

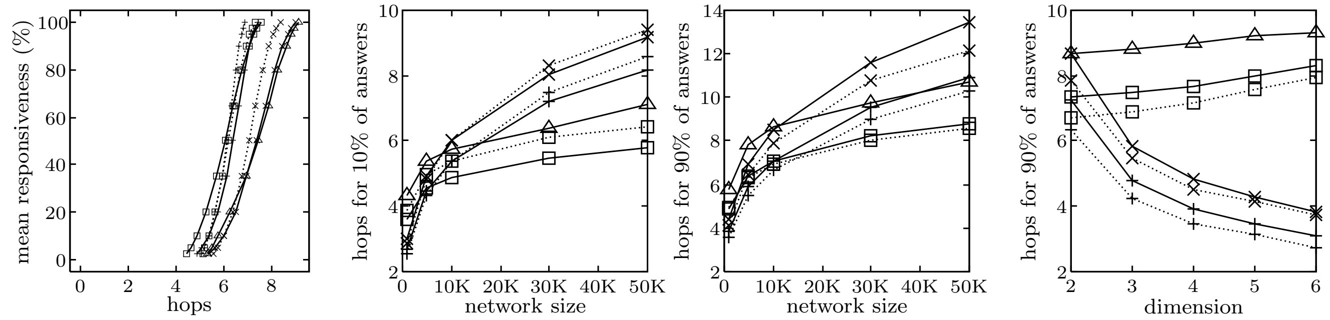
4.1 Responsiveness

Query latency and responsiveness are probably the most studied metrics in structured P2P networks. As already discussed, responsiveness is often more appropriate if complete answers are not required. Fig. 3(a) displays the average responsiveness of all networks for the R2 point set. As can be seen, every network routes queries for a few hops and then quickly reports answers. This shape of curve is typical, so in Figs. 3(b,c) we only show the hop count at which responsiveness attains specific values. Fig. 3(b) presents responsiveness for dataset R2 and varying network sizes. As can be seen, these curves grow roughly logarithmically with network size. Also, as the dimension increases (Fig. 3(c)), MURK and CAN-SP attain 90% responsiveness much faster, consistent with [10]. The most interesting observation is that, in all cases, the effect of data balancing on responsiveness is negligible.

4.2 Maximum Throughput

We now turn our attention to the contention-related metrics of our model. Fig. 4(a) shows the maximum throughput for the R2 point set over network size, for queries of size 40–60. A striking conclusion is that in two dimensions, except for PGrid-Z, the maximum throughput of other networks *does not scale with network size.* VBI-HR has the lowest maximum throughput ≈ 7. In fact, all data balanced networks, including PGrid-Z, have reduced maximum throughput as the network size increases (for MURK it actually decreases).

In Fig. 4(b) the effect of dimensionality on maximum throughput is depicted, for networks of 10,000 peers and queries of 40–60 points on the Hd point sets. Although PGrid-Z maintains the highest maximum throughput, its performance decreases significantly. Even more dramatic

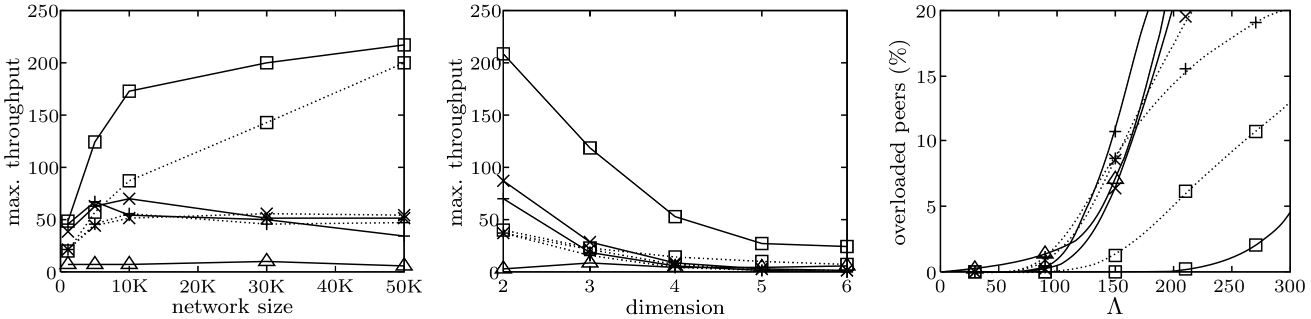


(a) Responsiveness for R2 point set. (n=10K, $|q|$=40–60)

(b) R2 point set, 10th and 90th percentiles.

(c) Hd point set. (n=10K)

Figure 3: Responsiveness percentiles. ($|q|$=40–60)



(a) R2 point set.

(b) Hd point set. (n=10K)

(c) R2 data set. (n=10K)

Figure 4: Effect of throughput on stability. ($|q|$=40–60)

is the decrease for MURK and CAN-SP, whose maximum throughput drops below 1. This implies that it is preferable to store the whole dataset on a single peer.

Our metric of maximum throughput may be criticized as being too pessimistic, as it only depends on the single most loaded peer in a P2P network. It may be argued that just one overloaded peer will not cause too much trouble for the rest of the network in practice. To counter this argument, we present in Fig. 4(c) the percent of overloaded peers (those for which ρ_j exceeds 1) in all networks, as the throughput increases. It can be seen that less strict definitions of maximum throughput would yield similar qualitative results.

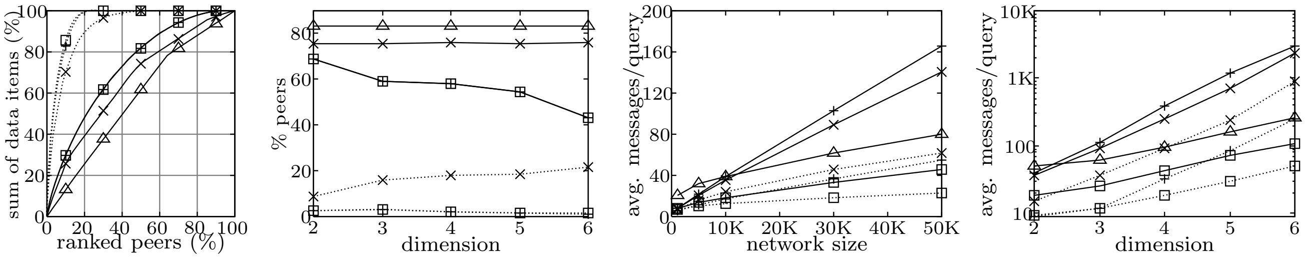
4.3 Data and load distribution, network traffic

We now turn our attention to some typical metrics appearing in the literature, related to contention. The data distribution for each network is shown in Figs. 5(a,b). Fig. 5(a) depicts the percent of the R2 dataset stored in the most loaded x% of the peers for $x = 0 \ldots 100$. A totally even distribution is denoted by the diagonal, while curves further away denote more uneven distributions. It is observed that volume-balanced selection results in uneven data distribution, with 10% of the peers storing 70–85% of the dataset. In particular, PGrid-Z exhibits quite bad balancing, although Fig. 4(a) reveals that it scales well under contention. Similar remarks hold in higher dimensions, as can be seen in Fig. 5(b).

As might be expected, uneven data balancing induces uneven distribution of query accesses (that is, the percentage of query processes in which a peer participates). In Fig. 6 we plot the distribution of query accesses throughout the network, for the R2, H2, H4 and H6 point sets. Query accesses are relatively balanced in VBI-HR and all data-balanced topologies, while for all volume-balanced topologies, 20% of the peers is accessed by approximately 70% of the queries. Yet, this discrepancy does not correlate to the actual scalability of these networks.

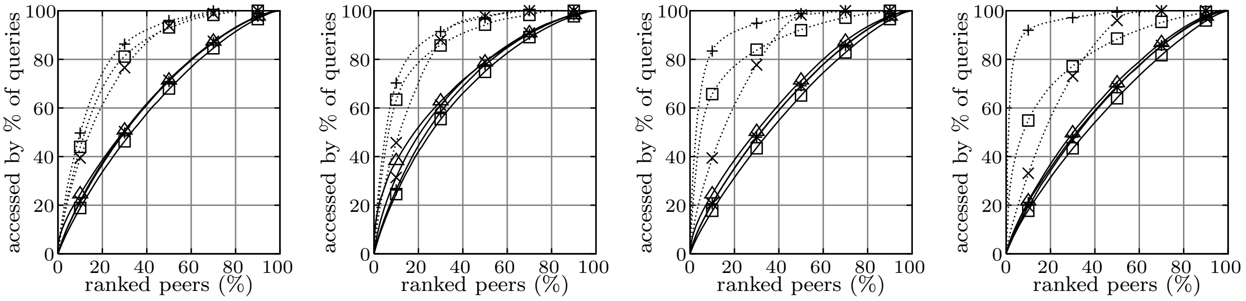
We now turn our attention to average per-query network traffic M. Intuitively, one expects that a lower value of M implies better overall performance under high throughput. Figs. 5(c,d) depict M as it varies with network size for the R2 dataset, and with the dimension, for the Hd dataset respectively. It can be seen that PGrid-Z exhibits lower M, and is indeed the most scalable network. Yet, the space-balanced MURK and CAN-SP networks have also low traffic for the R2 dataset, as the network scales. Also, the least scalable network, VBI-tree, is in fact not the most expensive in network traffic. Another unexpected observation is that, in general, better load balancing implies higher network traffic! In particular, the volume-balanced topologies all have lower network traffic than the corresponding data-balanced ones. This partially explains the fact that CAN-SP and MURK performance decreases in Fig. 4(a) as the network becomes larger, since the higher per-query traffic will eventually counteract their more even message distribution.

CAN-SP and MURK have up to 1-2 orders of magnitude higher traffic as the dimension increases, despite the fact that the number of peers accessed per query is relatively low. The cause of this inflation in traffic is in the forward-



(a) R2 point set. (b) % of peers storing 90% of Hd dataset. (c) R2 point set. (d) Hd point set. (n=10K)

Figure 5: Data distribution and per-query network traffic. ($|q|$=40–60)



(a) R2 point set. (b) H2 point set. (c) H4 point set. (d) H6 point set.

Figure 6: Distribution of query accesses. (n=10K, $|q|$=40–60)

ing of a query to all relevant neighbors. If a query has k relevant peers, it will require $\Omega(k \min\{k, 2^d\})$ messages. Furthermore, k also increases with the dimension. In high dimensions, this results in exorbitant message traffic.[1] For the 6-d dataset, M becomes comparable to network size, explaining the drop of $\Lambda_{\max}$ below 1, observed previously. On the other hand, this extravagant expenditure of network resources yields impressive responsiveness (cf. Fig. 3(c)).

The overall conclusion from the results in this section supports our thesis, that network traffic and even balancing do not correlate to scalability in the presence of contention, and they do not affect responsiveness noticeably, when throughput is low.

5. CONCLUSIONS

This paper introduced a novel performance analysis for search protocols on structured P2P networks, with an emphasis on the effects of contention on scalability. Our analysis is based on well-understood principles of queueing theory and makes minimal assumptions regarding the homogeneity of peers, the nature of protocols or the structure of the underlying communication network. The associated performance metrics we propose depend only on the distribution of messages to peers.

We analyzed a number of recent works on low-dimensional range search, and revealed performance aspects of these works, which were previously unsuspected. We demonstrated that some widely used performance metrics, such as per-query traffic, locality, data distribution, etc., do not relate directly to performance aspects of interest, namely query latency and system throughput.

For multi-dimensional range search in particular, our results reveal a number of intriguing aspects. In most works so far, it was assumed implicitly—if not explicitly—that many well established principles from the area of multidimensional indexing, such as even space partitioning or data distribution, would carry over in P2P networks. Our results indicate that, although these aspects play a role, they are not what determines the quality of a solution. For example, our results on PGrid-Z vs. CAN-SP demonstrate that two networks with identical data and space partitioning, and equally reasonable topologies, can have very different performance with respect to throughput.

Regarding the three networks we have studied in our experiments, we can broadly say that their search performance in the absence of load is generally good, but, for all but one, scalability under contention is bad.

5.1 Future work

This paper has not taken into account a number of important aspects of P2P networks related to their dynamic nature, most prominently peer failures, churn, and dynamic adaptation. Advances in characterizing such dynamic behavior have been made only for DHTs (e.g. [23]). Our techniques, applied to these problems, may be able to characterize more accurately the quality of dynamic adaptation protocols proposed in the literature.

Another important aspect, unstudied in this paper, is the effect of contention on responsiveness. Similar problems

[1] The authors of [10] do not provide measurements of the per query traffic, but only of the number of relevant peers per query.

have long been studied by the networking community, with very few analytical results. As P2P network protocols are typically more complicated than network routing protocols, it seems unlikely that purely analytical results will be found in the near future. As fully detailed simulation of P2P networks is very resource demanding, it seems important to seek ways to approximate the problem to the best possible accuracy, using less detailed simulations.

Last, but certainly not least, our results indicate that some proposed search techniques for P2P networks may not scale well under contention. Therefore, an immediate research direction should be to evaluate other existing techniques and improve their scalability. Of particular interest would be analytical guarantees of scalability for particular networks. Our maximum throughput metric has a simple enough definition to be amenable to such results.

Acknowledgments

This research was co-funded by the European Social Fund and National Resources–EPEAEK II–PYTHAGORAS. We would like to thank Stavros Christodoulakis for valuable discussions.

6. REFERENCES

[1] The R-tree portal, http://www.rtreeportal.org.

[2] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-Grid: A self-organizing structured P2P system. *SIGMOD Record*, 32(3), 2003.

[3] L. Arge, D. Eppstein, and M. T. Goodrich. Skip-webs: Efficient distributed data structures for multi-dimensional data sets. In *PODC*, pages 69–76, 2005.

[4] J. Aspnes and G. Shah. Skip graphs. In *SODA*, pages 384–393, Philadelphia, PA, USA, 2003.

[5] A. R. Bharambe, M. Agrawal, and S. Seshan. Mercury: Supporting scalable multi-attribute range queries. In *SIGCOMM*, pages 353–366, 2004.

[6] Y. Chawathe, S. Ramabhadran, S. Ratnasamy, A. LaMarca, S. Shenker, and J. Hellerstein. A case study in building layered DHT applications. In *SIGCOMM*, pages 97–108, 2005.

[7] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB*, pages 426–435, 1997.

[8] A. Crainiceanu, P. Linga, J. Gehrke, and J. Shanmugasundaram. Querying peer-to-peer networks using P-trees. In *WebDB*, pages 25–30, 2004.

[9] A. Datta, M. Hauswirth, R. John, R. Schmidt, and K. Aberer. Range queries in trie-structured overlays. In *P2P*, pages 57–66, 2005.

[10] P. Ganesan, B. Yang, and H. Garcia-Molina. One torus to rule them all: Multidimensional queries in P2P systems. In *WebDB*, pages 19–24, 2004.

[11] Z. Ge, D. R. Figueiredo, S. Jaiswal, J. F. Kurose, and D. F. Towsley. Modeling peer-peer file sharing systems. In *INFOCOM*, 2003.

[12] S.-T. Goh, P. Kalnis, S. Bakiras, and K.-L. Tan. Real datasets for file-sharing peer-to-peer systems. In *DASFAA*, pages 201–213, 2005.

[13] A. Gupta, D. Agrawal, and A. el Abbadi. Approximate range selection queries in peer-to-peer systems. In *CIDR*, 2003.

[14] A. Haeberlen, A. Mislove, A. Post, and P. Druschel. Falacies in evaluating decentralized systems. In *IPTPS*, 2006.

[15] R. Huebsch, J. M. Hellerstein, N. Lanham, B. T. Loo, S. Shenker, and I. Stoica. Querying the internet with PIER. In *VLDB*, pages 321–332, 2003.

[16] H. V. Jagadish, B. C. Ooi, and Q. H. Vu. BATON: A balanced tree structure for peer-to-peer networks. In *VLDB*, pages 661–672, 2005.

[17] H. V. Jagadish, B. C. Ooi, Q. H. Vu, R. Zhang, and A. Zhou. VBI-tree: A peer-to-peer framework for supporting multi-dimensional indexing schemes. In *ICDE*, page 34, 2006.

[18] P. Kalnis, W. S. Ng, B. C. Ooi, and K.-L. Tan. Answering similarity queries in peer-to-peer networks. *Inf. Syst.*, 31(1):57–72, 2006.

[19] I. Kamel and C. Faloutsos. Hilbert R-tree: An improved R-tree using fractals. In *VLDB*, pages 500–509, 1994.

[20] D. R. Karger and M. Ruhl. Simple efficient load balancing algorithms for peer-to-peer systems. In *SPAA*, pages 36–43, 2004.

[21] S. S. Lavenberg. *Computer Performance Modeling Handbook*. Academic Press, Inc., Orlando, FL, USA, 1983.

[22] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik. *Quantitative system performance: Computer system analysis using queueing network models*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1984.

[23] D. Liben-Nowell, H. Balakrishnan, and D. Karger. Analysis of the evolution of peer-to-peer systems. In *PODC*, pages 233–242, 2002.

[24] P. Maymounkov and D. Mazières. Kademlia: A peer-to-peer information system based on the XOR metric. In *IPTPS*, pages 53–65, 2002.

[25] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *SIGCOMM*, pages 161–172, 2001.

[26] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Distributed Systems Platforms (Middleware)*, pages 329–350, 2001.

[27] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM*, pages 149–160, 2001.

[28] C. Tang, S. Dwarkadas, and Z. Xu. On scaling latent semantic indexing for large peer-to-peer systems. In *SIGIR*, pages 112–121, 2004.

[29] D. Tsoumakos and N. Roussopoulos. Analysis and comparison of P2P search methods. In *InfoScale*, page 25, 2006.

[30] C. Zheng, G. Shen, S. Li, and S. Shenker. Distributed segment tree: Support of range query and cover query over DHT. In *IPTPS*, 2006.