

Service Differentiation in Peer-to-Peer Networks Utilizing Reputations

Minaxi Gupta, Mostafa Ammar
College of Computing, Georgia Institute of Technology
Atlanta, GA, U.S.A.
{minaxi, ammar}@cc.gatech.edu

Abstract

As the population of P2P networks increases, service differentiation issues become very important in distinguishing cooperating peers from free-loaders. The basis for service differentiation could either be economic or the fact that the peers differ from each other in the type of services and resources they contribute to the system. Taking the latter approach, this paper makes three contributions: 1) it defines parameters that are suitable for service differentiation in P2P networks, 2) it proposes SDP, a protocol to accomplish service differentiation, and 3) it identifies a set of features that are necessary in a reputation system that measures the contributions of individual peers in the system.

1 Motivation

Many flavors of peer-to-peer (P2P) networks are in wide use today to access a variety of content. All P2P networks fall in one of the three major categories: 1) centralized P2P networks like Napster, 2) decentralized unstructured networks like Gnutella and Kazaa, and 3) decentralized structured networks like CAN [1] and CHORD [2]. The prominent centralized P2P application, Napster was forced to shut down due to legal issues and the structured P2P networks are still at a research stage. However, many flavors of unstructured P2P networks are currently in use. The popularity of these networks can be judged from the fact that Kazaa (www.kazaa.com) routinely supports of the order of two million simultaneous peers and transports more than 300 million files.

Each peer is both a client and a server in P2P networks. Perhaps this inherent notion of equality of peers is one of the reasons why service differentiation issues have not received much attention from the research community. However, as the population of these networks increases, issues like free-loading behavior¹ make it necessary to differentiate among the peers. Various measurement studies of existing P2P systems like Napster and Gnutella have observed a widespread presence of free-loading in these networks. For example, it is reported in [3] that at the time of the study, nearly 70% of Gnutella users shared no files, and nearly 50% of all responses were returned by the top 1% of sharing hosts. Another study in [4] reports the free-loaders to be about 25% in Gnutella but much less in Napster. In order to motivate peers to contribute more resources for the common good of the system than they take away from it, Kazaa defines the notion of *participation level*. This participation level is then used to prioritize among the peers under heavy access conditions.

There are two main directions one can pursue in order to provide service differentiation in P2P networks. First, economic factors can be used to create different classes of peers. However, due to the absence of the traditional client-server model of delivery in P2P networks, pricing issues in these networks are yet to be resolved. Second, the fact that the peers differ from each other in the quality of services and resources they provide (or are able to provide) can be exploited. A measure of the latter is often referred to as the *reputation* of the peer.

Research in designing *reputation systems* to gauge peers' contribution to the P2P network has gained momentum [5, 6, 7, 8, 9, 10]. The basic idea behind these reputation systems is to assign a reputation to each peer based on the satisfaction experienced by the other peers from the services it provides to them. While being able to find reputable peers is beneficial from the perspective of a peer wishing to retrieve content, it may serve as a disincentive for peers acting as servers to have a good reputation for the fear of being overwhelmed. An effective service differentiation scheme could serve as a motivation for peers to possess good reputations.

This paper takes the approach of using peer reputations to address service differentiation issues in P2P networks. The goal of service differentiation is not to provide hard guarantees but to create a distinction among the peers based on their contributions to the system. The basic idea being, the more the contribution, the better the relative service.

¹Free-loaders are peers who only download content but do not serve it to the other peers.

While service differentiation parameters are well understood and studied in the context of the Internet (e.g. delay, jitter, bandwidth), they are still to be defined for the P2P networks. With special focus on decentralized unstructured P2P networks, this paper makes three main contributions. First, it defines a set of parameters that can be used to create service differentiation in P2P networks. Second, it proposes a service differentiation protocol SDP, to accomplish service differentiation. Third, it identifies a set of features that are necessary in a reputation system to be able to use it for service differentiation.

The rest of this paper is structured as follows. Section 2 defines the parameters for service differentiation in P2P networks. Section 3 describes SDP in detail and section 4 discusses the requirements from the underlying reputation system. A discussion of security and participation issues in SDP is presented in 5. Finally, sections 6 and 7 present the evaluation and the conclusion respectively.

2 Service Differentiation in P2P Networks

In this paper, we assume that the peer reputation scores are used to map them into various LoSs. The parameters that can be used to define each LoS are discussed in section 2.2. Assuming that a service differentiation scheme consisting of 3 levels of service (LoS) is used, figure 1 shows an example of how the peer reputation scores (RSs) can be mapped to various LoSs using parameters a and b . In this scheme, peers for whom $RS < a$ are eligible for *basic LoS*. $a \leq RS < b$ provides *enhanced LoS* to the peers. Peers with $RS > b$ receive *premium LoS*. The parameter a and b are expected to be known to each peer in the P2P network.

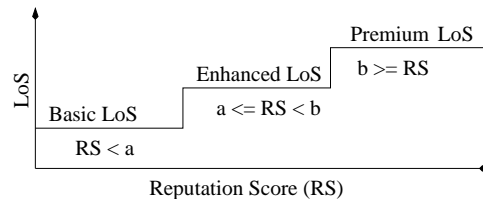


Figure 1: An example of 3 levels of service in a P2P network.

2.1 Main P2P Functions

Peers carry out three main functions in a P2P network. Cooperation among the peers is required for all these functions. The overall experience of a peer in a P2P network depends on the network conditions and the services and resources provided by the other peers during each of these functions. The network conditions depend on many factors that may not be controllable within a P2P overlay topology and as a result are not considered in this paper. Focusing on decentralized unstructured P2P networks, this section describes the typical functions peers carry out in a P2P network.

- **Bootstrapping:** Bootstrapping is required to allow peers to join the network. Most unstructured P2P networks use something similar to a *GWebCache*. GWebCache servers (typically several hundred in number) are web servers running a special module that allows the Gnutella *servents*² to query them to find the addresses of other servents. These servers comprise a distributed system to solve the bootstrapping problem.
- **Content search:** The content retrieval process in all the existing flavors of P2P networks involves a content search phase before it can be downloaded. In unstructured P2P networks like Gnutella, to search for the desired content, querying peer generates a query with appropriate *keywords* and sends it to all the peers that it is directly connected to in the overlay topology. The peers who process this query reply back if they have the content in their shared directory and forward the request to the peers they are directly connected to depending on the *hop-count* (or the TTL) of the query. This forwarding continues until the TTL specified by the querying peer is exhausted.
- **Content download:** The second phase in content retrieval involves selecting a peer to download the content from. The querying peer selects a peer after receiving all the replies from the content search phase. At that point, the content download typically uses a HTTP or a TCP connection with the selected peer.

²Gnutella peers are called *servents* because they serve both as a server and a client

2.2 Parameters for Service Differentiation

The set of parameters that can be mapped to each LoS are guided by the factors that provide service differentiation during the bootstrapping, content search, and content download functions in a P2P network; and hence the peer's perception of service quality. These factors are based on the salient features of the widely deployed unstructured P2P networks and the results of the current research on unstructured P2P networks.

2.2.1 Factors Affecting Bootstrapping

During the bootstrapping process, the type of peers a peer directly connects to in the overlay topology play an important role in its overall satisfaction from content search and download functions later on. For example, apart from how cooperative the connecting peers are, their actual network distance, processing power, memory, bandwidth, and storage capacity are important factors.

2.2.2 Factors Affecting Content Search

The following factors can be used as a basis for service differentiation because they impact the perception of service for a peer during the content search:

- **Number of hops:** To search content, the querying peer sets the maximum number of hops in the overlay topology its query would take, by denoting a *hop-count*. While the success of the content search phase depends on many other factors as well, the number of hops plays an important role. Hence, setting a *hop-limit* could act as a component of the service differentiation scheme.
- **Premium content:** Peers can choose to classify some of the content they share as premium content, which they can make available only to peers eligible for certain minimum LoS. This classification can be done through some system wide guidelines.
- **Hard to find content:** A special utility of the P2P networks for many peers comes from being able to access *hard-to-find* content. Although classifying content as hard-to-find may be based on subjective criteria, peers can potentially reserve the hard-to-find content only for peers with a certain minimum LoS.
- **Query caching:** Sripanidkulchai [11] found that the popularity of search strings in Gnutella follows a Zipf-like distribution and that caching a small number of queries results in a significant decrease in the traffic in the Internet. In order to distinguish among the peers with various LoSs, the outcome of caching queries may be made available only to peers with a certain minimum LoS eligibility.
- **Cached content:** Kazaa distinguishes between the functionality of *supernodes* and the rest of the peers in its P2P network. Peers with higher bandwidths can choose to become supernodes in a Kazaa P2P network. During idle periods, the supernodes actively query other peers in the network and cache the content so retrieved. This gives the supernodes access to additional content and when queries for the cached content arrive at the supernodes, they can get served faster. For faster retrieval of content in unstructured P2P networks, Cohen et. al. [12] have also proposed caching strategies. Due to its potential to improve the peer experience, caching could be used to distinguish among the service provided to peers with various LoSs.
- **Interest-based locality:** By exploiting interest-based locality, Sripanidkulchai et. al. [13] have proposed an efficient content search solution for unstructured P2P networks. The basic idea is for peers that share similar interests to create *shortcuts* to each other. These shortcuts can then be used to locate content faster. The basic Gnutella content search paradigm remains as a backup mechanism. In creating such shortcuts, peer LoS eligibility may be used as an additional deciding factor.
- **Load balancing:** An enhancement to maintaining a shortcut to peers that share common interests (as described above) would be to maintain the most recent load for those peers as well. Such an information can help in avoiding the already overwhelmed peers and potentially get content faster. The availability of such information however would require a periodic protocol to assess the load for the peers with similar interests. But if this information is available, it can be provided to peers eligible for certain minimum LoS during content search phase.

2.2.3 Factors Affecting Content Download

During the content download phase, following factors can be used as a basis for service differentiation because they impact a peer's overall experience:

- **Rate of transfer:** During content download from the chosen peer, the rate of transfer may be dependent on the LoS the downloading peer is eligible for. The basic idea is to restrict the portion of capacity used to serve the peers with less than a certain LoS. These restriction may either be in effect all the time or may be used only during periods of heavy loads.
- **Scheduling policy:** During periods of heavy load or even otherwise, the peers may use various scheduling policies in order to give priority in serving content to the peers with with premium LoS over the peers with enhanced LoS and to enhanced LoS over basic LoS.

3 Service Differentiation Protocol (SDP)

This section describes SDP, a protocol to accomplish service differentiation in P2P networks.

3.1 Basic Assumptions

SDP enhances the basic functionality of Gnutella-like unstructured P2P protocols to include the service differentiation functionality. It assumes that the peers have immediate access to their own reputation scores. Assuming that the network stores reputations in a decentralized manner, one way to accomplish this is through local storage of reputations. However, security issues in such a storage need to be carefully addressed. Another alternative is to compute reputations *just-in-time* from a distributed storage. These issues are described in detail in section 4.

SDP is flexible about the structure of peer reputations. The reputation score could be a *scalar* or a *vector*. There is no requirement on how the freshness of the reputation score is guaranteed. If the underlying reputation system wishes to guarantee freshness, it can assign a time-stamp to the reputation scores. SDP only requires that the structure of the reputation scores be known to all peers. Several examples of peer reputations exist in the literature today [5, 6, 7, 8, 9, 10].

Further, SDP assumes that the mapping of the reputation scores to the LoSs is known to all the peers in the P2P network. Such a mapping could be statically configured by being a part of the software itself or could be downloaded from the GWebCache servers during the bootstrapping process. Also, the peers are expected to know what values of the parameters described in section 2.2 are mapped to each LoS.

The anonymity issues in SDP are dealt with in a manner similar to those in the popular unstructured P2P protocols. The details of this and other security issues are discussed in section 5.

3.2 SDP Details

During the bootstrapping process, most popular unstructured P2P protocols provide an option to connect only to *high-capacity* (in terms of bandwidth, processing power, memory, and storage) peers. Such high-capacity peers are referred to as *supernodes* in Kazaa and *ultrapeers* in Limewire (www.limewire.com). Additionally, *binning* scheme of the type proposed in [14] can be used to allow peers to connect to peers close-by in the Internet topology. These factors impact the QoS perceived by the peers and can be incorporated in a service differentiation scheme. In this paper, we focus only on the service differentiation during the search and download process of the content retrieval.

For the subsequent SDP description, we assume that a unstructured P2P protocol like the one described in the Gnutella specification [15] is being used. The terminology used is the same as used in that specification and all enhancements proposed by SDP are on top of such a protocol.

3.2.1 Content Search

This section describes how the content search part of unstructured P2P protocols can be modified to incorporate the service differentiation functionality. The parameters used by SDP to provide service differentiation during this phase are the same as those described in section 2.2.2.

Search phase 1: The peer who initiates the search request sends its reputation score along with the *Query* message. We refer to this enhanced query as *Query-SDP*. This requires enhancing the Gnutella query to include the peer's reputation score in addition to the standard fields like TTL, hops, search criteria etc. The basic idea behind *Query-SDP* is to allow each that processes the query to have access to the querying peer's reputation score.

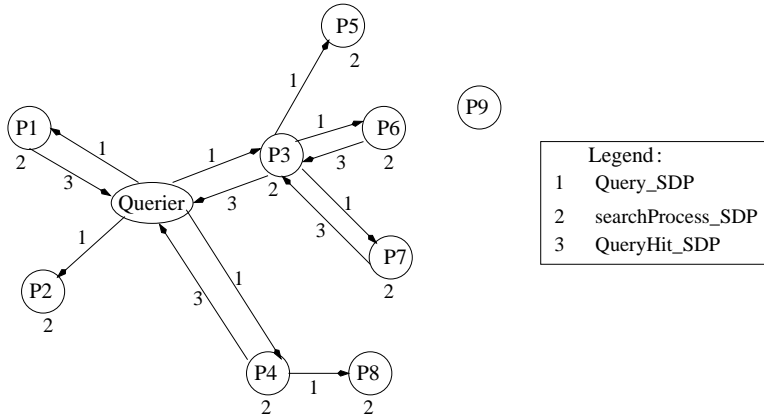


Figure 2: SDP during content search for a peer eligible for basic LoS

Search phase 2: Each peer who receives *Query_SDP* extracts the reputation score. This score is used to map the peer to the LoS it is eligible for and process the query accordingly. The LoS specific processing is referred to as the *searchProcess_SDP*. Since the processing is dependent only on the reputation score, SDP does not require any identification for the querying peer. Also, the peers who process the query do not have to cache the reputation scores for any other peer in this scheme. This is because the reputation scores may change over time.

In order to map the reputation score to a LoS, parameters a and b of the type described in section 2. As described before, if the system supports 3 LoSs, peer with $reputationscore < a$ would be eligible for a *basic LoS*. $a \leq reputationscore < b$ would be used to provide *enhanced LoS* to the peers. Peers with $reputationscore > b$ would receive *premium LoS*.

Examples of functions that would be a part of *searchProcess_SDP* are shown in figure 3. These functions would provide appropriate LoS using the parameters described in section 2.2.2. Assuming 3 LoSs implies that there are 3 separate functions, one for each LoS. The functions in figure 3 assume that the number of allowable hops for basic, enhanced, and premium LoS are given by $hops_basic$, $hops_enhanced$, and $hops_premium$ respectively. The basic idea behind these functions is the following. If a peer's query has already traversed more hops than it was eligible for, it is dropped immediately. However, if it has not traversed extra hops but would go farther than should (based on the $TTL+Hops$), then appropriate value for the TTL needs to be set. Further, for enhanced and premium LoS peers, additional lookups are needed for service differentiation.

After the LoS specific processing, the query continues to be processed per Gnutella guidelines. This is denoted by *process Query* at the end of each of the functions in 3. Since *interest-based locality* and *load balancing* parameters from section 2.2.2 require additional protocols to be run, we eliminate them from these functions. However, if such information is available, it can be incorporated easily. As a result, the functions currently use query caching results, cached content, premium content and hard-to-find content for differentiating among the peers.

Search phase 3: Notice from the functions in figure 3 that the LoS specific query processing may amount to dropping the query. But if a query is not dropped during search phase 2, the peer forwards it on its outgoing interfaces according to Gnutella specifications. In case a response is to be sent back to the querying peer after processing the query, *QueryHit_SDP* is sent. *QueryHit_SDP* is an enhancement to the Gnutella *QueryHit* message. *QueryHit_SDP* allows peers who reply to optionally put their reputation scores in the response. This is to help the querying peer make a decision about who to download the content from based on the reputation of the responders.

An example: Figure 2 shows an example of a section of an unstructured P2P network running SDP during the search process. Assume that the querying peer's reputation score makes it eligible for basic LoS which is mapped to $hops_basic = 2$. In the first hop, *Query_SDP* is forwarded to peers P1, P2, P3, and P4. Upon finding relevant content, P1 replies to the querying peer using *QueryHit_SDP*. Because one more hop is allowed, in addition to responding to the querying peer using *QueryHit_SDP*, peers P3 and P4 forward *Query_SDP* to P5, P6, P7, and P8. Peers P6 and P7 respond back using *QueryHit_SDP* but peer P6 does not forward the *Query_SDP* further because the hop count is exhausted and drops the query.

3.2.2 Content Download

This section describes how SDP enhances the content download process in unstructured P2P protocols to incorporate support for service differentiation. SDP uses the parameters described in section 2.2.3 for this phase.

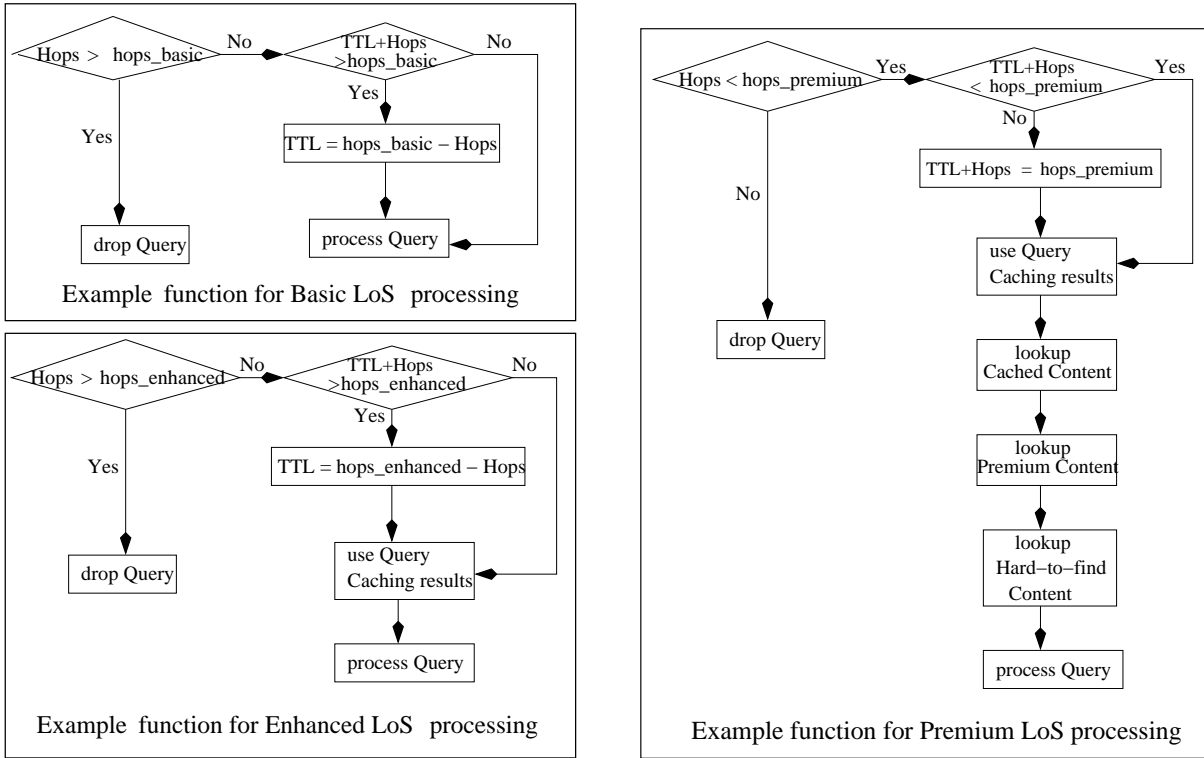


Figure 3: Functions for service differentiation during content search.

Download phase 1: After selecting a peer to download from, the querying peer connects to the selected peer using a TCP or HTTP connection. Just as in search phase 1, this phase also requires the querying peer to send its reputation score while establishing the connection for downloading.

Download phase 2: Before serving the content, the sender peer maps the reputation score to the LoS the requester peer is eligible for. Once the LoS is decided, the sender peer picks the appropriate rate of transfer and scheduling policy for the LoS.

The topic of what transfer rates to use and the particular scheduling policies employed needs more research and is beyond the current scope of this paper.

4 Reputation Scores

This section describes how the SDP requirements translate into guidelines for a reputation system. It also compares the existing reputation systems for their suitability to accomplish service differentiation using SDP.

To be able to provide service differentiation, SDP requires reliable reputation scores to be available to the peers who process the content search and download requests. Instead of having to deduce the reputation scores by each peer who needs it, SDP assumes that they are sent by the requester peers along with the request. As described in section 3.2, SDP is flexible about the structure of the reputation scores.

Ebay (www.ebay.com), Advogato (www.advogato.org), and Slashdot (www.slashdot.org) centrally track the reputations of their users. Centralization is a reasonable choice for the purposes these systems are used for. However, having to retrieve reputations from a central location is not a scalable choice for SDP because even one search/download can lead to many peers attempting to retrieve the reputation of the requester peer.

Various reputation systems that store reputations in a distributed form have been proposed in the recent research literature [5, 6, 7, 8, 9] for the purposes of helping peers avoid transactions with malicious peers. Although these proposals differ from each other in many ways, the following common features make them unsuitable for SDP:

- *Reputation inference:* Although these reputation systems store reputations in a distributed fashion, irrespective of the particular scheme chosen to store the data, they all require reputations to be computed *on-demand* which requires cooperation from the peers in performing computations. For SDP, this would introduce additional latency when the reputation scores are needed. Also, similar to the case when reputations are stored centrally,

this solution will not be scalable because each search/download may require many peers to have access to the reputation scores.

- *Subjectiveness*: In order to keep the reputation computations distributed, all existing reputation systems are based on a subjective assessment of reputations. Subjectiveness is part of the reason to why reputation inference overheads are necessary. An *objective* reputation computation is important not only to remove the need for on-demand reputation inference, but to incorporate a detailed set of rules to judge a peer's cooperation in the P2P network.
- *Range of values*: Although several reputation systems have a limited range of values that peer reputations can assume, reputation values in most of the proposed reputation systems are *non-decreasing*. It is a relatively minor inconvenience for the mapping of reputation scores to various LoSs in SDP. This is because as the reputation values increase, this mapping may have to be changed. Thus, a desired feature would be to have a fixed range of reputation values.

A Partially Distributed Reputation System: The reputation system proposed in [10] satisfies the requirements of SDP. This partially distributed solution uses a trusted reputation computation agent (RCA) to compute scalar and objective reputations. The basic idea behind ensuring reliable reputation computations is for the peers to collect the credit for their work in the system and periodically contact the RCA to have that credit converted into a reputation equivalent. The RCA encrypts the reputation score by its public key. Since RCA's key is assumed to be known to all the peers, any peer can decrypt the reputation score. Peers store their reputations locally for fast retrieval in this proposal.

To respect the privacy concerns of peers, the solution requires the peers to explicitly *enroll* with the RCA to participate in the reputation computations. This is a function that needs to be carried out only when a peer joins the P2P network for the first time. The new peers and the peers who choose not to enroll in the reputation computations always have a default reputation score of 0 in the system. By earning a good reputation score, they can upgrade their reputation score. The reputations in this system either expire or are *used-up* by the peers. Hence, they do not always increase. Thus, in order to maintain a good reputation, peers need to continually participate in the system. The reputations are saved across sessions, so peers with good reputations can continue to benefit from them.

5 Discussion of Security and Participation Issues

Decentralization is one of the core strengths of the P2P networks because it makes them robust to failures. However, the same decentralization makes it hard to enforce rules in any light-weight manner. As a result, any modification to the P2P system could only motivate peers by providing them incentives.

Security Issues: We have so far not discussed any security issues in ensuring the functioning of the proposed service differentiation. Malicious peers can thwart SDP in two primary ways. First, they may not give better LoS to peers who are eligible for enhanced or premium LoS. Second, they may collude with other peers and give each other a better LoS than the points justify.

While performing bootstrapping, search, and download functions in a P2P network, peers rely on the presence and services of other peers. Current P2P networks do not have any provisions for isolating misbehaving peers and work on goodwill of majority of the peers. Since the success of bootstrapping, search, and download functions is not reliant on the behavior of any particular peer, the P2P networks leverage redundancy for their successful operation.

Unless the underlying functionality of the P2P network is guaranteed to be secure, security of the proposed service differentiation can not be guaranteed. In fact, tracking the behavior of each peer in the system in a manner that does not compromise the scalability of the system does not seem possible. The reputation system assumed by SDP ([10]) guarantees the security and correctness of the reputation scores. And SDP functions on the belief that if a peer trusts the reputation of the other peer, it will be willing to provide them with appropriate LoS. Moreover, the rewards from malicious behavior while providing service differentiation are limited unlike the one's that are possible by being able to assume a good reputation score without earning it.

Further, Gnutella like protocols do not address anonymity issues. SDP does not change the anonymity characteristics of the unstructured P2P protocols. As a result, it does not provide anonymity. However since it does not require any identifier for the reputation scores, it is independent of whether or not the underlying P2P network provides anonymity. SDP makes the service differentiation decisions based only on the reputation scores and not who sent it.

Participation Issues: The lack of participation may not necessarily arise out of malicious behavior. It may also be because of a peer running a version of the P2P software which is not modified with SDP enhancements.

The proposed service differentiation scheme is relatively insensitive to a small percentage of peers not providing the adequate LoS. During the search phase, the redundancy aspect is likely to offset the effect of peers not providing appropriate LoS. And peers who do not send their reputation scores may be avoided for content download. The effect of participating is demonstrated by the evaluation results presented in section 6.

6 Evaluation

SDP can be evaluated along the following 4 dimensions:

- **Effectiveness:** The actual differentiation in services provided to peers belonging to different LoSs during the content search and download phases is an important measure of effectiveness of SDP.
- **Sensitivity to participation:** Expecting that all peers in the system run SDP for it to be effective is not possible due to various reasons. Peers could be running different versions of the the underlying Gnutella protocol and also could be malicious. As a result, it is important to gauge the sensitivity of SDP to the extent of participation required from peers in implementing the SDP functionality.
- **Overheads:** SDP enhances the Gnutella *Query* and *QueryHit* messages to include the reputation scores. It also involves extra processing on the part of peers who process the content search and download requests. Though the processing as well as the bandwidth overheads of SDP are expected to be minimal, an estimation of these overheads is another dimension of evaluation.
- **Impact of parameter values:** The exact values of parameters used while mapping them to various LoSs is another important consideration. For example, the percentages of premium and hard-to-find content, the number of cached files, and the number of cached queries are useful factors in the deployment of SDP.

Our simulations focus on the first two parameters from the above list. We generated connected topologies with peer populations ranging from 5000 to 25,000. Compared to the actual populations of widely deployed P2P networks, these topologies are small. However, our main goal in the preliminary evaluation presented in this paper is to observe the general trends in the effectiveness of SDP.

Each peer in the simulation topologies is connected on an average to about 4 other peers in the system. We assume that the total number of files in each topology is the same as the peer population. The file popularities are Zipf distributed with a parameter of 1.0. At the beginning of the simulation, each peer possesses one unique file. As the peers access more files, they cache them and serve them to other peers. This leads to the file propagation. The file sizes are uniformly distributed between 08MBytes. Also, each peer has a *capacity-limit*, implying that it can not cache any more than a certain number of files.

The requests for files in this system are uniformly distributed among all the peers with an exponential inter-arrival time of 50 requests/second. The simulation logs are 1 hour long for each topology. Further, the simulations assume that 3 LoS (basic, enhanced, premium) are in use and that all the peers stay in the system for the entire duration of the logs and that the peer reputations do not change during that time period.

To estimate the effectiveness of SDP in providing service differentiation, we focus on the search aspect. Further, we assume that parameters like premium content, hard-to-find content, cached queries, and cached files are constant across all the LoSs. The only parameter differentiating among the peers with different LoSs is the number of hops their queries are allowed to go. These simulations assume that all the peers are running SDP and that they exhibit greedy behavior in that the number of hops specified by the peers in their searches exceed the eligibility of their reputation scores.

We also estimate the sensitivity of SDP to the participation of peers during content search. While lack of participation could also imply various kinds of malicious behavior, we only consider the lack of participation to mean that the peers are not SDP enhanced. As a result, they process the queries as they would in the case of Gnutella.

For the above evaluations, we experimented with 3 population sizes of 5000, 10,000, and 25,000 peers in the system. We assumed three combinations of the percentage of peers eligible for basic, enhanced, and premium LoS respectively. These combinations were (20, 20, 60) (40, 40, 20), and (60, 20, 20). We refer to each of them as the percentage-tuple for simplicity. The first, second, and third entries of each percentage-tuple correspond to the percentage of peers eligible for basic, enhanced, and premium LoS respectively. For each population size and each percentage-tuple, we experimented with various cache sizes (501000 files) and various sets of hop-tuples. Each hop-tuple (i, j, k) corresponds to the maximum number of hops the queries for basic, enhanced, and premium LoS peers are allowed to go. With all other parameters staying constant, the effect of varying population sizes was that the

smaller the peer population in the topology, the more the successes during content search (and hence lesser failures). For each topology, the results across various cache sizes were identical. We now present the results for the topology with a population size of 10,000 and a cache size of 1000 files.

Figures 4(a), 4(b), and 4(c) show the effect of hop-tuple and percentage-tuple on the success of queries for the basic, enhanced, and premium LoS peers respectively. Irrespective of the particular hop-tuple, the success rates for premium LoS peers are the highest. The success rates for enhanced LoS peers are better than those of basic LoS but worse than the premium LoS peers. This shows that for the topologies tested, SDP is able to provide service differentiation for content search using just the number of hops. Comparing across graphs for peers belonging to the same LoS respectively, it becomes clear that for basic and enhanced LoS peers, as the percentage of peers eligible for the same service increases, the success rate deteriorates. This is expected because the presence of more basic and enhanced peers implies fewer premium peers and less successes during content search. The total effect is that of lesser propagation of files. Also, as expected, as the number of hops for individual LoS increase, the success rates improve as well.

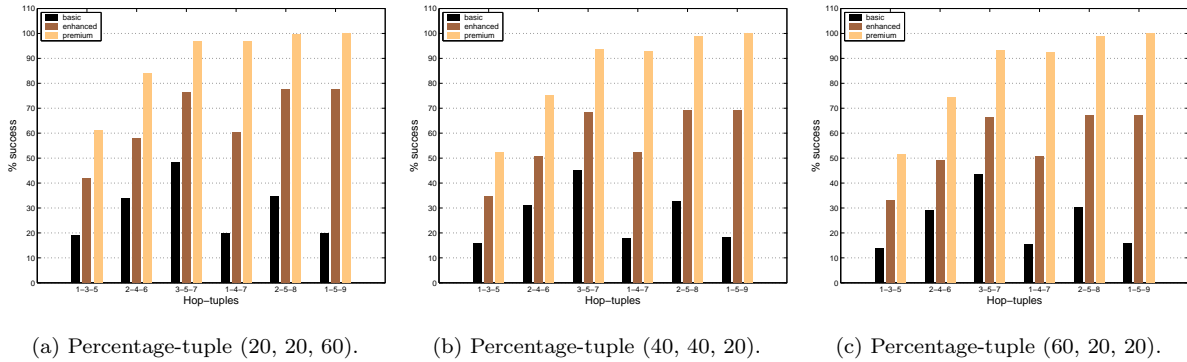


Figure 4: Effectiveness of SDP during search.

Figures 5(a) and 5(b) show the effect of lack of participation on the effectiveness of SDP during content search for hop-tuples (1, 3, 5) and (2, 4, 6) respectively. The trends for hop-tuples (1, 4, 7) and (1, 5, 9) were similar to that of (1, 3, 5), the only difference being the increased % successes for the former hop-tuples as expected. Similarly, the trends for hop-tuples (2, 5, 8) and (3, 5, 7) were similar to that of (2, 4, 6). The graphs in figure 5(a) and 5(b) are for percentage-tuple (40, 40, 20). The graphs for percentage-tuples (20, 20, 60) and (60, 20, 20) were almost identical. Since querying peers are assumed to be greedy, as fewer percentage of peers implement SDP, the overall trend in both the graphs is that of decreased service differentiation among the basic, enhanced, and premium LoS peers. This is expected because the presence of more non-SDP peers means that more querying peers will get content search replies from farther away than their LoS justifies. For figure 5(b), when the percentage of peers not participating increases beyond 30%, the difference in service diminishes quicker than for figure 5(a).

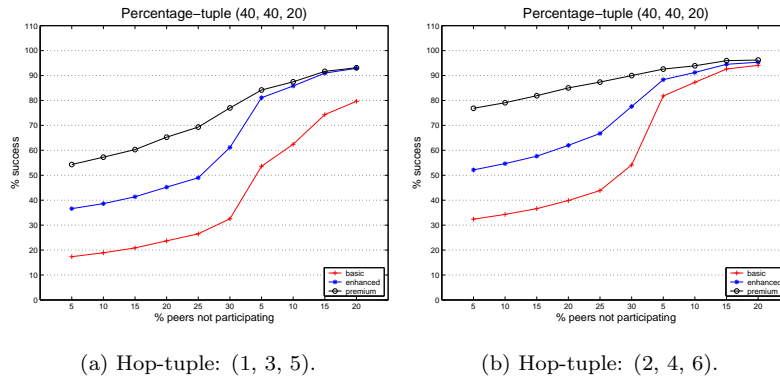


Figure 5: Effect of participation on SDP during search.

7 Conclusion

We presented SDP, a protocol for service differentiation in decentralized unstructured P2P networks. SDP uses parameters that affect the content search and download functions to provide different LoSs. It accomplishes this by enhancing the Gnutella specification. Though SDP is independent of the underlying reputation system used to decide on the LoS a peer is eligible for, a system where peers store their reputations locally is the most efficient. A preliminary evaluation of the service differentiation achieved during the search phase shows the promise of the approach.

8 Acknowledgment

The authors will like to thank Paul Judge for many useful insights during the initial discussions on the topic.

References

- [1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content addressable network,” in *ACM SIGCOMM*, Aug. 2001.
- [2] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, “Chord: A scalable Peer-To-Peer lookup service for internet applications,” in *ACM SIGCOMM*, Aug. 2001, pp. 149–160.
- [3] E. Adar and B. A. Huberman, “Free riding on Gnutella,” Tech. Rep., Xerox PARC, 2000.
- [4] S. Saroiu, P. K. Gummadi, and S. D. Gribble, “A measurement study of peer-to-peer file sharing systems,” in *SPIE Conference on Multimedia Computing and Networking (MMCN)*, Jan. 2002.
- [5] K. Aberer and Z. Despotovic, “Managing trust in a peer-2-peer information system,” in *Ninth International Conference on Information and Knowledge Management (CIKM)*, Nov. 2001.
- [6] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, “A reputation-based approach for choosing reliable resources in peer-to-peer networks,” in *9th ACM Conference on Computer and Communications Security*, Nov. 2002.
- [7] S. D. Kamvar, M. Schlosser, and H. Garcia-Molina, “Eigenrep: Reputation management in p2p networks,” Unpublished work, 2003.
- [8] S. Lee, R. Sherwood, and B. Bhattacharjee, “Cooperative peer groups in nice,” in *IEEE INFOCOM*, Apr. 2003.
- [9] L. Xiong and L. Liu, “Building trust in decentralized peer-to-peer communities,” in *International Conference on Electronic Commerce Research (ICECR-5)*, Oct. 2002.
- [10] M. Gupta, P. Judge, and M. H. Ammar, “A reputation system for peer-to-peer networks,” in *NOSSDAV*, June 2003.
- [11] K. Sripanidkulchai, “The popularity of gnutella queries and its implications on scalability,” White Paper Featured on O’Reilly’s website <http://www.openp2p.com/>, Feb. 2001.
- [12] E. Cohen and S. Shenker, “Replication strategies in unstructured peer-to-peer networks,” in *ACM SIGCOMM*, Aug. 2002.
- [13] K. Sripanidkulchai, B. Maggs, and H. Zhang, “Efficient content location using interest-based locality in peer-to-peer systems,” in *IEEE INFOCOM*, Apr. 2003.
- [14] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, “Topologically-aware overlay construction and server selection,” in *IEEE INFOCOM*, Apr. 2002.
- [15] “Gnutella protocol specification,” http://www9.limewire.com/developer/gnutella_protocol.0.4.pdf/.