Hybrid Client-Server and Peer-to-Peer Caching Systems with Selfish Peers

Youngmi Jin and Yung Yi  
EE Dept., KAIST, Korea  
\{youngmi_jin,yiyung\}@kaist.ac.kr

George Kesidis and Fatih Kocak  
EE and CSE Depts., The Penn. State Univ  
\{gik2,fwk5027\}@psu.edu

Jinwoo Shin  
CS Dept., Georgia Tech.  
jshin72@cc.gatech.edu

Abstract—This paper considers a hybrid peer-to-peer (p2p) system, a dynamic distributed caching system with an authoritative server dispensing contents only if the contents fail to be found by searching an unstructured peer-to-peer (p2p) system. We study the case when some peers may not be fully cooperative in the search process and examine the impact of various non-cooperative behaviors on the querying load on the server as the peer population size increases. We categorize selfish peers into three classes: impatient peers that directly query the server without searching the p2p system, non-forwarders that refuse to forward query requests, and non-resolvers that refuse to share contents. It is shown that in the hybrid p2p system, impatient and/or non-forwarding behaviors prevent the system from scaling well because of the high server load, while the system scales well under the non-resolving selfish peers. Our study implies that the hybrid p2p system does not mandate an incentive mechanism for content sharing, which is in stark contrast to unstructured p2p systems, where incentivizing peers to share contents is known to be a key factor for the system’s scalability.

I. INTRODUCTION

In a simplified peer-to-peer (p2p) search scenario, peers and known service/data objects are given geospatial coordinates. Queries to (presumed) known coordinates are then forwarded to neighboring peers that are closest. Under certain topological conditions (e.g., the presence of long distance neighbors [1]), it can be shown that forwarding is efficient. Typically based on consistent hashing, distributed hash table (DHT) coordinate systems also have good forwarding-delay properties, e.g., [2]; peers are expected to be able to resolve queries for data/service objects which are proximal to themselves in (hash) key space. In an unstructured search, as considered in this paper, the contact/resolution time in hops of a single-threaded or limited-scope flooded query (including an anycast) has been studied using techniques from the spectral theory of Markov chains and random graphs, e.g., [3]–[7].

In this paper, we consider the impact of selfish behaviors on content distribution using local search for a hybrid unstructured p2p and client-server framework consisting of an authoritative server and unstructured peers [8]–[11]. The interest on a hybrid p2p system is increasing since it can provide scalable content distribution and overcome weaknesses of purely unstructured p2p systems, including illegal content disseminations. Recently, the authors in [6] showed that hybrid p2p systems have good scalability under the assumption that all peers are cooperative, implying that both obtaining scalability and preventing illegal content distribution are feasible. They proved that a hybrid p2p system can alleviate the server load significantly for a random walk based content search, under some conditions on the structures of the peer-connectivity graph family and time to live for query resolution.

However, it has been observed that a considerable portion of peers in practice do not cooperate mainly in content sharing and possibly search query forwarding, and such selfish behaviors significantly degrade the performance of content distribution [12], [13]. Thus, it is of critical importance to study how the performance of hybrid p2p systems will be in presence of selfish peers, which has been under-explored to the best of our knowledge. Note that peers can have various selfish behaviors and scenarios in terms of what and how. Thus, in this paper we classify selfish behaviors into three categories: (i) impatience meaning direct access to the server without searching other peers, (ii) non-forwarding that refuses to forward queries, and finally (iii) non-sharing that refuses to share contents. In our model, the system consists of a server, selfish peers and altruistic (hence cooperative) peers; some fraction of peers are selfish. Also, depending on how selfish peers behave, we consider two scenarios: static one in which selfish peers always act selfishly when handling queries during the whole stay period in the system, and probabilistic one in which selfish peers opt to act selfishly with some probability when handling queries. The probabilistically selfish peers correspond to those who may want to hide their non-cooperativeness to avoid being detected. This paper investigates how each selfish behavior or their multiple combinations affect on the scalability of a hybrid p2p system measured by the server load as the number of peers increases.

The main contributions of the paper are as follows:

1) We provide mathematical and numerical analysis of the impacts of various selfish behaviors on the server load for both static and probabilistic scenarios. In both scenarios, we prove that the scalability of the hybrid p2p system is preserved in presence of non-resolving (i.e., non-content-sharing) peers under the same conditions for the scalability of fully cooperative hybrid p2p systems. This is in stark contrast to that in purely unstructured p2p systems. However, the scalability does not hold any more in presence of non-forwarding or impatient peers.

2) For the static scenario, we obtain closed forms of the server loads and the probabilities of query resolution by the server under various selfish behaviors. This leads to our main result about the scalability of the hybrid p2p system under the static scenario. These studies additionally offer more accurate scalability properties, and they can also be independently useful to other analytical studies of hybrid systems.
p2p systems. The main novelty of this analysis lies in a
definition of “contact set”, that is an extension of the set of
peers with the content in the fully cooperative p2p system.
This allows us to make mathematical connections between
fully cooperative studies in [6] and partially cooperative
ones pursued in this paper.
3) The analysis of the probabilistic scenario is much more chal-
lenging than that of the static one since the peer selfishness
is intermittent and hence the contact set mentioned above
becomes highly dynamic. To overcome this issue, instead of
obtaining closed forms as in the static scenario, we
provide comparison results between the static scenario and
the probabilistic one, which suffice to determine scalability.
Our approach considers “virtual walks” in the query propa-
gation. This provides analytic separation between the query
propagation dynamics and the peers’ selfish behaviors.
Our results imply that for a hybrid p2p content distribution
(or caching) system, an incentive mechanism for content shar-
ing is not necessary, while an incentive mechanism for impa-
tient peers and/or non-forwarding peers is essential to guarantee
the scalability of the system. Note that incentive mechanisms
for content-sharing have been extensively studied, e.g., [14]
and less attention has been paid to other selfish behaviors than
non-sharing selfish behaviors [15]. We show that in hybrid
p2p systems, non-forwarding and unconditional access to the
server cause more dominant increase of the server load than
non-content-sharing. Our finding suggests two opposite and
arguable points. Since the impact of non-forwarding selfish
behaviors is critical, study on incentive mechanisms for query
forwarding may be important [16]. Searching information over
social networks could be achieved through query propagation in
p2p systems. The other aspect is that query forwarding cost can
be regarded negligible since the forwarding cost is much less
than content-sharing cost. If forwarding cost is small enough to
be considered negligible and peers are willing to forward the
queries, then in the hybrid p2p system, it suffices to consider an
incentive mechanism to prohibit peers from direct accessing the
server. Such an incentive mechanism seems much simpler than
that for content sharing, because an incentive mechanism for
content sharing requires a complicated design of fair rewarding
and implementation difficulties such as heavy communication
overhead load and reliability [14]. Our results show that a
hybrid p2p system can be a more practical scalable and efficient
content distribution architecture against selfish behaviors than
a fully distributed p2p system with an incentive mechanism for
content-sharing, when the forwarding cost is negligible.

II. Model
In this section, we describe our model of a hybrid p2p system
with a server and many peers. Among many content search
mechanisms, we consider a popular random walk based query propa-
gation. Our model is similar to that in [6] except that
some peers may be selfish while others are cooperative.

A. Network, Peer Churn, and Query Propagation
Network. We consider a hybrid p2p system that has a single
server and \( N \) peers. The peers form an unstructured p2p
network and all of them have direct connectivity to the server.
The peers constitute an undirected, connected graph \( G(V,E) \).
The graph \( G \) represents a p2p overlay network, where one hop
in the overlay may correspond to multiple “physical” hops.
Denote by \( d_i \) the degree of node \( i \), and let \( d := \max_{i \in V} d_i \)
be the maximum degree. Once the graph \( G \) is given, there is
an associated random walk, which is a discrete-time Markov
chain with transition probability matrix \( R \) whose entries are, for
all peers \( i, j \in V \).

\[
R_{ij} = \begin{cases} 
\frac{1}{d_i}, & \text{if } i \neq j, \quad (i, j) \in E, \\
0, & \text{otherwise,}
\end{cases}
\]  

Peer churn. Peers dynamically enter and leave the system. We
assume that as soon as a peer departs the system, a new peer
enters the system and replaces the departing peer (this modeling
assumption for peer churn is commonly made in literature, see,
e.g., Sec. 2.4 of [17]). Thus, neither the total number of peers
nor its graphical topology \( G \) changes. Peers stay in the system
for an independent and identically exponentially distributed
time with mean \( 1/\mu \). Peers can generate query requests for the
contents in the system. For simplicity, we consider the case
where the queries are generated for only one content and those
query requests are generated only by newly entering peers with
probability \( p \). The query request probability \( p = p(N) \) can be a
function of the total number of peers \( N \). Then, by Little’s
formula [18], the mean rate at which peers “arrive” is \( N/(1/\mu) \),
and so the mean load (query rate) at which queries are generated is:

\[
\text{mean load} = pN\mu. \tag{2}
\]

A new peer, who is cooperative, first sends a query to her
neighboring peers who further relay the query to other peers.
However, there may exist selfish peers who directly access the
server. After sending a query to the peers, the (cooperative) peer
initiating the query waits for a query resolution response until
the given time to live \( T_{\max} \). If the peer does not get the response
by \( T_{\max} \), it deems the process of searching the p2p network as
failed and sends a new request directly to the server. Let \( A = \Delta(t) \subset V \) be the set of peers possessing the content at time \( t \).
We assume the time-scale separation between churn dynamics
and content resolution, i.e., the maximum query response time
\( T_{\max} \) is negligible compared to the mean peer lifetime \( 1/\mu \).
Then, the churn state \( \Delta(t) \) forms a continuous-time Markov
process with transition rates depicted in Fig. 1. If a peer, \( x \in A \)
leaves the system (at rate \( \mu \)) and the new peer replacing \( x \)
does not send a request (with probability \( 1-p \)), then the state
changes from \( A \) to \( A - \{x\} \), i.e., at rate \( (1-p)\mu \). If \( y \in A^c \) leaves the system (at rate \( \mu \)) and a new peer replacing \( y \) sends
a request (and acquires the content, with probability \( p \)), then
state changes from \( A \) to \( A \cup \{y\} \) (at rate \( p\mu \)).

Query propagation. We assume that queries are propagated by a
continuous-time random walk which is a lazy version of
(1) where the holding time at each peer \( i \) is independently
exponentially distributed with mean \( 1/\delta \). A cooperative peer
issuing a query sends a query packet to one of its neighbors
which is chosen uniformly at random. A peer that receives the
query packet checks whether it has the requested content. If
of peers, we classify selfish behaviors in two categories: those who do not share contents and those who share contents with probability $\sigma$. To formally model selfishness, we assume that the new entering peer is selfish with probability $\sigma$ and the arriving peer is cooperative or vice versa. We further assume that the departing peer does not necessarily inherit the selfish property of the departing one: it is possible that the exiting peer is selfish and the arriving peer is cooperative or vice versa. We define $\beta \in (0, 1]$ as the probability that a peer acts selfishly when it handles a query request.

**Main Results**

This section summarizes and discusses the main results of the paper. It is assumed that all peers are cooperative, and the authors of [6] got the probability of query resolution by the server, $s(p)$, and showed that the average load on the server, $\mu N p s(p)$, is bounded as $N \to \infty$ regardless of $p(N)$ (note that $p$ is a function of $N$) when $T_{\text{max}}(N) = \Omega(N)$ and $\{G(N)\}_{N=2}^\infty$ is an expander family whose definition is as follows (see Corollary 1 in [6]). A sequence of graphs $\{G(N)\}_{N=2}^\infty$ is an expander family if

$$\hat{\tau} := \limsup_{N \to \infty} \tau_{G(N)} < \infty$$

(see also [19] and (3) of [6]), where

$$\tau_G := (1 - \lambda_2^R)^{-1}$$

for a graph $G = (V, E)$ with $|V| = N$ and $\lambda_2^R$ is the second largest eigenvalue of the transition probability matrix $R$ associated with $G$ (see (1)). It was evidenced that the overlay graphs of unstructured p2p systems are expanders (e.g., see [3]).

**B. Our Results**

Once there is a constant fraction of selfish peers, the selfish behaviors generally increase the server load and may compromise the scalability of the hybrid p2p system. The increment on the server load depends on selfish behaviors as stated in Theorem 1.

**Theorem 1:** For both static and probabilistic cases, as $N \to \infty$, the hybrid p2p systems have server load that is

(a) **bounded** under non-resolving peers, if $\{G(N)\}_{N=2}^\infty$ is an expander family and $T_{\text{max}}(N) = \Omega(N)$ regardless of $p(N)$;

(b) **unbounded** under impatient, blackhole, or completely selfish peers when $\lim_{N \to \infty} N p(N) = \infty$; and

(c) **unbounded** under non-forwarding peers when $\lim_{N \to \infty} p(N)$ exists and $\lim_{N \to \infty} p(N) < 1$, under the condition that $\lim_{N \to \infty} N p(N) = \infty$.

The condition that $\lim_{N \to \infty} N p(N) = \infty$ in (b) and (c) means that the demand for the content increases with the population size, that is, the content is still popular even the size of the p2p system gets large. Hence the results in (b) and (c)
imply that the server in the hybrid p2p system cannot support query requests for popular contents as the total number of peers is increasing.

Note that under the identical conditions for scalability of a fully cooperative hybrid p2p system, the non-resolving selfish behavior does not destroy the scalability of the p2p system. In the hybrid p2p system, a query request is always resolved either by a peer or by the server, once generated. Hence a peer generating a query request ultimately owns the content and there is a sufficient number of (cooperative) peers with the content. However, in a fully distributed p2p system without a server, some query requests may not resolved because of the absence of a server and selfishness of peers. The unresolved query requests may increase the number of peers without the content and as a result the performance of content distribution can be severely deteriorated by the non-resolving selfish behavior.

In the static case, we get the closed forms for the probability of query resolution by the server corresponding to the cases of various selfish behaviors and mathematically confirm the intuition that multiple selfish behaviors increase the server load more than a single selfish behavior. The closed forms of the probability of query resolution by the server enable us to examine the limit of the server loads for the various cases of selfish behaviors as \( N \) goes to \( \infty \).

**C. Implications**

We discuss the implications of Theorem 1 with focus on the relation between incentive mechanisms and the server loads. The fact that the server loads are unbounded as the system size increases provides some guidelines on how we should design hybrid p2p systems against selfish peers.

*Non-resolving peers.* As stated in Theorem 1, non-resolving peers do not have critical impact on the server load for any fixed selfish portion \( \sigma < 1 \). It implies that a mechanism to incentivize peers to share contents is not necessary in hybrid p2p systems. This is in contrast to the case of purely unstructured p2p systems which typically exert significant efforts to develop nice incentive mechanisms for content sharing [17], [20], [21]. The main reason for this difference lies in the fact that we have a dedicated server in hybrid p2p, which ensures to sustain a reasonable degree of content availability, whereas in an unstructured p2p the content availability can be worsened over time by non-resolving peers.

*Impatient peers.* It is intuitive that the system does not scale well to the increase of the impatient peers, because the server load increases in proportion to a \( \sigma \) portion of \( N \) peers. However, this unscalability can be easily solved by employing a simple incentive mechanism with help of the server in hybrid p2p, e.g., enforcing impatient peers to pay a small fee for their impatient direct server access.

*Non-forwarding peers.* In this case, the server load also blows up even for a small portion of non-forwarding peers. Thus, the key of hybrid p2p system with selfish peers lies in how to incentivize them to forward queries. Two opposite, arguable points can be discussed for non-forwarding peers. The first is that the cost of forwarding (short) query messages may be negligible, compared to that of sharing contents unless resource is scarce. Thus, one may connect Theorem 1 and simple payment-based incentivization for impatient peers, to the implication that hybrid p2p systems are generally scalable even in presence of selfish peers. Note that this contrasts with the case of wireless ad hoc networks with scarce battery and bandwidth resources, where an extensive array of research on forwarding incentivization has been studied e.g., [22]–[24]. Another point is that despite low query forwarding cost, there may still exist a non-negligible portion of non-forwarding peers, e.g., simply malicious peers or peers in the competitive p2p systems [25], in which case a scheme for forwarding incentive needs to be applied to the system. However, an incentive mechanism for non-forwarding peers seems to involve some degree of hardness and complexity, especially compared to that for impatient peers. Due to the dedicated server in hybrid p2p, developing an incentive mechanism in this case can be easier than that of wireless ad hoc networks, yet it may incur heavy message passing among the server and the peers. Incentive mechanisms for forwarding queries get important as the search for information over social networks becomes popular [16]. To answer more definitely, further experimental studies on existence as well as the portion of non-forwarding peers in hybrid and unstructured p2p systems are necessary.

**IV. Static Selfish Behaviors**

This section analyzes the effect on the p2p system performance of each type of static selfish behaviors in the aspect of the server load. To investigate the asymptotic server load, we will first find the probability that a query is resolved by the server and the average time to find a peer with the content by random walk for each selfish behavior. The performance from the querying peer’s point of view is captured by the latency.

Before starting our analysis on the p2p system under the selfish behaviors, we first introduce necessary preliminaries for the fully cooperative p2p system in Section IV-A. In the following sections, we will discuss the server load for the five selfish behaviors in Section II. From now on, a random walk always means the continuous time random walk in (3).

**A. Fully Cooperative Peers**

We state the probability of query resolution by the server and the average time to find a peer with the content in [6]. Let \( T \) be the average time to find a peer possessing the content by a random walk and \( s \) be the probability that a query is resolved by the server. Therefore, for a completely cooperative p2p system (i.e., \( \sigma = 0 \)), the probability that the p2p system resolves a query is \( 1 - s \), and by (2) the mean load (query rate) on the server is

\[
l(p) = spN\mu.
\]

The two quantities, \( s \) and \( T \), are obtained by conditioning the set of peers possessing the content, \( A \). The steady-state distribution of the churn process \( A(t) \) is

\[
\nu_A(N, p) := P(A(t) = A) = p^{|A|}(1 - p)^{N - |A|},
\]

for \( A \subseteq V \), which does not depend on \( \mu \) (peer churn parameter).
For a given set \( B \subseteq V \), \( E_i[T_B] \) denotes the mean hitting time to the set \( B \) from peer \( i \) by the (query propagation) random walk and \( P_i(T_B > t) \) denotes the probability that a random walk starting from the peer \( i \) hits the set \( B \) more than \( t \) (seconds).

The expressions for \( T \) and \( s \) are respectively (see equations (6)-(9) (proof of Prop. 1) and the proof of Prop. 2 of [6]).

\[
s(p) = \sum_{A \neq V} \nu_A(N-1,p)f_A, \tag{5}
\]
\[
T(p) = \sum_{A \neq V} \nu_A(N-1,p)g_A, \tag{6}
\]

where

\[
f_A := \frac{1}{N} \sum_{j \in A^c} P_j(T_A > T_{\max}),
\]
\[
g_A := \frac{1}{N} \sum_{j \in A^c} E_j\left[\min\{T_A, T_{\max}\}\right].
\]

Note that for a given set \( A \subseteq V \), \( \frac{N}{N-|A|} f_A \) is the probability that a randomly chosen query-issuing peer does not get the content until \( T_{\max} \) and \( \frac{N}{N-|A|} g_A \) is the average time to find a peer with the content within \( T_{\max} \). We emphasize the following facts:

- \( g_A \) and \( f_A \) do not depend on \( p \).
- \( f_A > f_{A \cup \{x\}} \) and \( g_A > g_{A \cup \{x\}} \) for \( x \notin A \) and \( A \subseteq V \).

Note that \( f_\emptyset = 1 \) and \( g_\emptyset = T_{\max} \) and that \( g_A \) and \( f_A \) depend on the p2p overlay graph structure and the dynamics of the random walk. In addition, we have found useful properties of \( s(p) \) and \( T(p) \) as follows. (see the Appendix for the proof).

**Lemma 1**: \( T(p), s(p) \) are decreasing convex functions of \( p \).

**B. Impatient Peers (S1)**

If a peer is impatient with probability \( \sigma \) (though always cooperatively relaying and resolving) and fully cooperative with probability \( 1 - \sigma \), then the probability that an “arriving” peer generates a query to the p2p system is simply reduced to \( (1 - \sigma)p \). But because the impatient peers are assumed to be cooperatively resolving and relaying, impatience does not have an effect on the probability of query resolution by the p2p system, \( 1 - s \), and the mean sojourn time of the query in the p2p system, \( T \).

Impatience does have an effect on the server querying load. By (2), the mean querying load to the server is increased to

\[
s^{\text{SIP}}(p) = \frac{pN\mu}{\kappa^\text{SIP}}, \tag{7}
\]

where \( s^{\text{SIP}}(p) \), the probability that the query is resolved by the server, is

\[
s^{\text{SIP}}(p) = \sigma + (1 - \sigma)s(p) \geq s(p)
\]

with \( s^{\text{SIP}}(0) \equiv s \). Similarly, the load per resolving peer is now \( \Pi^{\text{SIP}} = (1 - \sigma)\Pi(p) \), where \( \Pi^{\text{SIP}} \equiv \Pi \) is the load per resolving peer for fully cooperative p2p system (with \( \sigma = 0 \)).

**C. Non-resolving Peers (S2)**

Now assume \( \sigma \) is the probability that a peer is non-resolving (though patient and relaying), and a peer is cooperative with probability \( 1 - \sigma \). Here, a non-resolving peer (implicitly with content) acts just like a cooperative relaying peer (that does not possess the content).

For this subsection and the following subsection, we will use a contact set whose definition is as follows. A contact set \( C \) is a set of peers such that if a query request (random walk on the p2p overlay network) reaches any element \( j \in C \), then the query request is terminated.

When there are non-resolving peers, the contact set \( \hat{A} \) is a set of peers that have the content and are cooperative (resolving).

The p2p graph dynamics can be represented by a Markov process \( \hat{A}(t) \) with transition rates

\[
\hat{A} \rightarrow \hat{A} - \{x\} \quad \text{with rate } (1 - p + \sigma p)\mu \ orall x \in \hat{A},
\]
\[
\hat{A} \rightarrow \hat{A} \cup \{y\} \quad \text{with rate } (1 - \sigma)p\mu \ orall y \in \hat{A}^c.
\]

Hence, it follows that

\[
s^{\text{SNR}}(p) = \sum_{A \neq V} f_A\nu_A(N-1,(1-\sigma)p).
\]
\[
T^{\text{SNR}}(p) = \sum_{A \neq V} f_A\nu_A(N-1,(1-\sigma)p).
\]

Therefore, we have the following proposition.

**Proposition 1**: \( s^{\text{SNR}}(p) = s((1-\sigma)p) \) and \( T^{\text{SNR}}(p) = T((1-\sigma)p) \).

From the perspective of query resolution by the p2p system, the mean size of the contact set terminating (and here successfully resolving) the query has changed to \( \kappa^\text{SNR}\mu \) where

\[
\kappa^\text{SNR} := (1-\sigma)p, \tag{8}
\]

and, for the fully cooperative case,

\[
\kappa_0^\text{SNR} = p.
\]

In the following, \( \kappa \) is the probability that a peer belongs to the contact set \( \hat{A} \) that terminates a query before time to live \( T_{\max} \), whether or not contacting this set results in the query being successfully resolved. So we can relate the scenario with non-resolving peers to a fully cooperative one as

\[
s^{\text{SNR}}(p) = s(\kappa^\text{SNR}) > s(p) \quad \text{and} \quad T^{\text{SNR}}(p) = T(\kappa^\text{SNR}) > T(p),
\]

where the inequalities are by Lemma 1. Because we assume that the non-resolving peer is patient, the mean query rate to the server is \( s(\kappa^\text{SNR})pN\mu \).

**D. Blackhole Peers (S4)**

Now suppose that the probability that a peer is a blackhole is \( \sigma \), and that a peer is cooperative with probability \( 1 - \sigma \). The mean size of the random set of blackholes is \( E[A^{\text{SBH}}] = 1 - \sigma N \).

A query will “stop” when it either times out or contacts \( \tilde{A} := A^{\text{SBH}} \cup A = A^{\text{SBH}} \cup (A \setminus A^{\text{SBH}}) \), where

\[
E[\tilde{A}] = \kappa^{\text{SBH}}N \quad \text{and} \quad \kappa^{\text{SBH}} := \sigma + (1 - \sigma)p.
\]
The resulting p2p system dynamics can be represented by a Markov process \( \{A(t)\} \) with transition rates

\[
\begin{align*}
A &\rightarrow A - \{x\} \quad \text{with rate } (1 - p)(1 - \sigma)\mu \quad \forall x \in A \\
A &\rightarrow A \cup \{y\} \quad \text{with rate } (\sigma + p - \sigma\mu)\mu \quad \forall y \in \tilde{A}.
\end{align*}
\]

The closed forms of \( s_{\sigma}^{SBH}(p) \) and \( T_{\sigma}^{SBH}(p) \) are provided below.

**Proposition 2:**

\[
\begin{align*}
 s_{\sigma}^{SBH}(p) &= \frac{\sigma}{\kappa_{SBH}^{\sigma}} + s(\kappa_{SBH}^{\sigma}) \left(1 - \sigma\right)p \\
 T_{\sigma}^{SBH}(p) &= \frac{\sigma}{\kappa_{SBH}^{\sigma}} + T(\kappa_{SBH}^{\sigma}) \left(1 - \sigma\right)p.
\end{align*}
\]

**Proof:** By conditioning on \( \hat{A} = A^{SBH} \cup A \), we get

\[
1 - s_{\sigma}^{SBH}(p) = \sum_{\hat{A} \neq V} h_{\hat{A}}(\sigma, p)\nu_{\hat{A}}(N - 1, \kappa_{SBH}^{\sigma}).
\]

where \( h_{\hat{A}}(\sigma, p) = P(T_{\hat{A}} < T_{max}, X(T_{\hat{A}}) \in A \setminus A^{SBH}) \) (and we suppressed indication of conditioning on the initial (querying) peer on \( \hat{A}^c \)). Let \( X(t) \) be the peer handling the query at time \( t \), so that

\[
h_{\hat{A}}(\sigma, p) = P(T_{\hat{A}} < T_{max}, X(T_{\hat{A}}) \in A \setminus A^{SBH}) = \frac{1 - \sigma}{\kappa_{SBH}^{\sigma}} P(T_{\hat{A}} < T_{max}).
\]

Therefore,

\[
1 - s_{\sigma}^{SBH}(p) = \left(1 - \sigma\right)p \sum_{\hat{A} \neq V} P(T_{\hat{A}} < T_{max})\nu_{\hat{A}}(N - 1, \kappa_{SBH}^{\sigma})
\]

\[
= \left(1 - \sigma\right)p \left(1 - s(\kappa_{SBH}^{\sigma})\right).
\]

Using similar arguments, we have the result on \( T \). \( \square \)

Since blackhole peers have two selfish behaviors, non-resolving and non-forwarding, we intuitively expect the server load with blackhole peers is bigger than that with non-resolving peers. The following corollary shows that this intuition is true.

Recall that \( s_{\sigma}^{SNR}(p) = s((1 - \sigma)p) \) and \( T_{\sigma}^{SNR}(p) = T((1 - \sigma)p) \).

**Corollary 1:**

\[
 s_{\sigma}^{SBH}(p) \geq s_{\sigma}^{SNR}(p) \geq s(p)
\]

\[
 T_{\sigma}^{SBH}(p) \geq T_{\sigma}^{SNR}(p) \geq T(p).
\]

**Proof:** Let

\( E_1 = \{ \text{event such that } T_{\hat{A}} < T_{max} \text{ and } X(T_{\hat{A}}) \in A \setminus A^{SBH} \} \)

\( E_2 = \{ \text{event such that } T_{\hat{A} \setminus A^{SBH}} < T_{max} \} \).

Note that \( E_1 \subseteq E_2 \). Hence, \( P(T_{\hat{A}} < T_{max}, X(T_{\hat{A}}) \in A \setminus A^{SBH}) \leq P(T_{\hat{A} \setminus A^{SBH}} < T_{max}) \).

Therefore,

\[
1 - s_{\sigma}^{SBH}(p) \leq 1 - s((1 - \sigma)p)
\]

by (9), (10) and the fact that the mean size of \( A \setminus A^{SBH} \) is \((1 - \sigma)pN\). Finally, recall that \( s_{\sigma}^{SNR}(p) = s((1 - \sigma)p) \) by Proposition 1 and \( s((1 - \sigma)p) \geq s(p) \) by Lemma 1.

For \( T_{\sigma}^{SBH}(p) \), we can use the same argument since, \( E[\min\{T_{\hat{A} \setminus A^{SBH}}, T_{max}\}] \geq E[\min\{T_{\hat{A}}, T_{max}\}] \) together with Lemma 1 and Proposition 1 (again, suppressing indication of conditioning the initial peer in \( \hat{A}^c \) above). \( \square \)

We can similarly show that the mean query rate on the (non-blackhole) resolving peers satisfies

\[
\Pi_{\sigma}^{SBH}(p) = \frac{1 - s_{\sigma}^{SBH}(p)}{1 - s(p)} \Pi(p).
\]

**E. Non-forwarding Peers (S3)**

The effect of non-forwarding behavior is similar to that of blackhole behavior. Now suppose that the probability that a peer is non-forwarding is \( \sigma \), and that a peer is cooperative with probability \( 1 - \sigma \). The contact set that terminates a query is the set of peers that are selfish or that have the content. The mean size of the contact set is \( \kappa_{SNF}^{\sigma} N \), where

\[
\kappa_{SNF}^{\sigma} := \sigma(1 - p) + p = \kappa_{SBH}^{\sigma}.
\]

However, the probability that the query is resolved given that the contact set is reached is

\[
\frac{p}{\kappa_{SNF}^{\sigma}} = \frac{p}{\sigma(1 - p) + p},
\]

rather than \((1 - \sigma)p/\kappa_{SBH}^{\sigma}\) in Proposition 2. Hence (11) becomes

\[
1 - s_{\sigma}^{SNF}(p) = \frac{p}{\kappa_{SNF}^{\sigma}} \left(1 - s(\kappa_{SNF}^{\sigma})\right).
\]

Hence, we have the following proposition.

**Proposition 3:**

\[
 s_{\sigma}^{SNF}(p) = \frac{\sigma}{\kappa_{SNF}^{\sigma}} + s(\kappa_{SNF}^{\sigma}) \frac{p}{\kappa_{SNF}^{\sigma}}.
\]

Furthermore, we observe that

\[
 s_{\sigma}^{SBH}(p) \geq s_{\sigma}^{SNF}(p) \geq s(p)
\]

\[
 T_{\sigma}^{SBH}(p) \geq T_{\sigma}^{SNF}(p) \geq T(p).
\]

where we use \( s(0) = 1 \) for the first equality and the second inequality holds because of the convexity of \( s(p) \). Similarly, the analogous version of (12) holds as well.

**F. Completely Selfish Peers (S5)**

A completely selfish peer is both impatient and a blackhole. Considering the separate cases above, the probability that a query request is resolved by the server is

\[
 s_{\sigma}^{SCS}(p) = \sigma + (1 - \sigma) s_{\sigma}^{SBH}(p)
\]

\[
= s_{\sigma}^{SBH}(p) + \sigma(1 - s_{\sigma}^{SBH}(p)).
\]

Note that \( s_{\sigma}^{SCS}(p) \geq s_{\sigma}^{SBH}(p) \). To summarize,

\[
 s_{\sigma}^{SCS}(p) \geq s_{\sigma}^{SBH}(p) \geq s_{\sigma}^{SNR}(p) \text{ or } s_{\sigma}^{SNR}(p) \geq s(p).
\]

**G. Proof of Theorem 1 for Static Selfish Behaviors**

This subsection provides the proof of Theorem 1 under static selfish behaviors.

**S1.** Under impatient peers, by (7) the average load on the server is greater than \( \sigma p N \mu \) which diverges as \( N p \to \infty \).

**S2.** Under non-forwarding peers, the server load is

\[
pN\mu s(\kappa_{SNF}^{\sigma}) = pN\mu s((1 - \sigma)p)
\]

\[
= \frac{1}{1 - \sigma}(pN\mu s(p')).
\]
where \( p' = (1 - \sigma)p \) and the first equality is due to Proposition 1. By Corollary 1 in [6], \( p'N\mu s(p') \) is bounded if \( \{G(N)\}_{N=2}^\infty \) is an expander graph family and \( T_{\text{max}}(N) = \Omega(N) \) as \( N \to \infty \). Hence the server load \( s(\kappa^{\text{SNR}})pN\mu \) is bounded regardless of \( p \) with the assumptions.

**S3.** To derive a contradiction, suppose that the server load is bounded under non-forwarding peers. Namely, for some \( K < \infty \),

\[
\lim_{N \to \infty} \mu N ps^{\text{SNF}}(p) < K.
\]

Then, we observe that

\[
\mu N p \leq \frac{K}{s^{\text{SNF}}(p)} \leq K\frac{(1 - p) + p}{\sigma(1 - p)} = K\left(1 - \frac{1}{\sigma} + \frac{1}{\sigma(1 - p)}\right)
\]

(14)

where the second inequality holds by Proposition 3. Thus, when \( \lim_{N \to \infty} p(N) < 1 \), the right hand side of (14) has a finite limit, which contradicts that \( \mu N p \) is unbounded. Therefore, \( \lim_{N \to \infty} pNps^{\text{SNF}}(p) = \infty \).

**S4.** Under blackhole peers, the server load diverges as \( N \to \infty \) because

\[
\mu N ps^{\text{SBH}}(p) \geq \mu N p\frac{\sigma}{\sigma + (1 - \sigma)p} > \mu N p\sigma,
\]

where the first inequality is due to Proposition 2.

**S5.** Under completely selfish peers, the server load is unbounded simply because the completely selfish case is for the corresponding system with blackhole peers. \( \square \)

V. **Probabilistically Selfish Behaviors**

In this section, we consider probabilistically selfish behaviors. We remind the definition: an arriving peer is selfish with probability \( \sigma \) and will thereafter behave selfishly when handling queries only with probability \( \beta \). In other words, a selfish peer may behave differently when handling the same query more than once. Recall that our random walk is assumed memoryless. Note that this is another way to model how a node may (selfishly or maliciously) attempt deplete a query’s time to live \( T_{\text{max}} \). A motivation for acting selfishly in such a probabilistic manner is to avoid detection and classification as a non-cooperative peer.

A. **Proof of Theorem 1 for Probabilistically Selfish Behaviors**

We analyze the asymptotic server load for various probabilistically selfish peers which act selfishly (according to types of (S2)-(S4)) as in the previous section. A similar strategy to what we used to establish the asymptotic server load for static selfish peers does not work for probabilistically selfish peers. Instead, we show the following comparison results.

**Proposition 4:** Let \( \sigma_1\beta_1 = \sigma_2\beta_2 \) and \( \beta_1 \leq \beta_2 \). Then, it follows that

(a) \( s^{\text{SNR}}_{\sigma_1,\beta_1}(p) \leq s^{\text{SNR}}_{\sigma_2,\beta_2}(p) \leq s^{\text{SNF}}_{\sigma_1,\beta_1}(p) \)

(b) \( s^{\text{SNF}}_{\sigma_1,\beta_1}(p) \leq s^{\text{SNF}}_{\sigma_2,\beta_2}(p) \leq s^{\text{SBH}}_{\sigma_1,\beta_1}(p) \)

(c) \( s^{\text{SBH}}_{\sigma_1,\beta_1}(p) \leq s^{\text{SBH}}_{\sigma_2,\beta_2}(p) \leq s^{\text{FBH}}_{\sigma_1,\beta_1}(p) \)

Since we have

\[
s^{\text{SNR}}_\sigma(p) = s^{\text{SNR}}_{\sigma,1}(p), \quad s^{\text{SNF}}_\sigma(p) = s^{\text{SNF}}_{\sigma,1}(p), \quad s^{\text{SBH}}_\sigma(p) = s^{\text{SBH}}_{\sigma,1}(p)
\]

the conclusions of Theorem 1 for probabilistically selfish behaviors can be derived using Proposition 4 and Theorem 1 for static selfish behaviors. Now we present the proof of Proposition 4 in below.

**Proof of Proposition 4:** Let \( \mathcal{P} = i_1 \to i_2 \to \cdots \to i_L \) be the random walk generated by a query request (excluding the peer generating the request), and \( Y_n \in \{H,T\} \) be the random coin to decide whether peer \( i_n \) possesses the query content or not, i.e., \( P(Y_n = H) = p \). Similarly, we use \( Z_n \in \{H,T\} \) for \( i_n \) being selfish when entering the system, and \( W_n \in \{H,T\} \) for \( i_n \) acting selfishly when handling the query request. By considering “virtual” walks even after a query request is resolved (or non-forwarded), we assume that the random walk continues till time to live \( T_{\text{max}} \). Hence, \( P(\mathcal{P}) \) is independent of parameters \( \sigma, \beta \) and types of selfish behaviors, and only dependent on the underlying graph and holding times at peers. Using this notation, we have

\[
s(p) = \sum_{\mathcal{P}} P(\mathcal{P}) \cdot P(\text{A query is not resolved} | \mathcal{P}).
\]

Therefore, to prove part (a), (b) and (c), it suffices to study whether \( P(\text{A query is not resolved} | \mathcal{P}) \) decreases or increases when \( \sigma, \beta \) change.

For part (a), i.e., non-resolving peers, let \( B_n \) denote the event that \( Y_n = T \) or \( Y_n = Z_n = W_n = H \). Using this notation, we have

\[
P(\text{A query is not resolved} | \mathcal{P}) = P\left(\bigcap_{n=1}^L B_n | \mathcal{P}\right)
\]

\[(15)\]

where \( R(\mathcal{P}) \) and \( S_i(\mathcal{P}) \) denote the set of peers appearing in \( \mathcal{P} \) and the set of indexes for peer \( i \in R(\mathcal{P}) \) in \( \mathcal{P} \), respectively, i.e.,

\[
R(\mathcal{P}) = \{i_1, i_2, \ldots, i_L\} \quad S_i(\mathcal{P}) = \{n : i_n = i\}.
\]

For example, if \( \mathcal{P} = a \to b \to c \to b \to a \to d \), then \( R(\mathcal{P}) = \{a,b,c,d\} \) and \( S_{b}(\mathcal{P}) = \{1,5\}, S_{a}(\mathcal{P}) = \{2,4\}, S_{c}(\mathcal{P}) = \{3\}, S_{d}(\mathcal{P}) = \{6\} \). The second equality in (15) is from the independence between \( \bigcap_{n \in S_i(\mathcal{P})} B_n \) and \( \bigcap_{n \in S_i(\mathcal{P})} B_n \) if \( j \neq k \). Furthermore, we have

\[
P(\text{A query is not resolved} | \mathcal{P}) = \prod_{i \in R(\mathcal{P})} P(\bigcap_{n \in S_i(\mathcal{P})} B_n | \mathcal{P})
\]

\[(16)\]

where we use the fact that \( Y_{n_1} = Y_{n_2} \) and \( Z_{n_1} = Z_{n_2} \) (with probability 1) for \( n_1, n_2 \in S_i(\mathcal{P}) \). (16) increases as \( \beta \) increases and \( \sigma \beta \) is fixed.
For part (b), i.e., non-forwarding peers, note that
\[ P(\text{A query is resolved} \mid \mathcal{P}) = \sum_{n=1}^{L} P(\text{A query is resolved at the } n\text{-th peer } i_n \text{ of } \mathcal{P} \mid \mathcal{P}) \]
\[ = \sum_{n=1}^{L} P(E_n) \cdot P(\text{A query is resolved at } i_n \mid E_n, \mathcal{P}), \]
where we let \( E_n \) be the event that the random walk reaches \( i_n \) through \( \mathcal{P} \), i.e., a query is neither resolved nor non-forwarded till the \((n-1)\)-th peer of \( \mathcal{P} \). One can check that
\[ P(\text{A query is resolved at } i_n \mid E_n, \mathcal{P}) = \begin{cases} p & \text{if } i_n \text{ does not appear in } \mathcal{P}_n \\ 0 & \text{otherwise} \end{cases} \]
\[ P(E_n) = \prod_{i \in R(\mathcal{P}_n)} (1-p)(1-\sigma+\sigma \beta |S_i(\mathcal{P}_n)|) \]
where \( \mathcal{P}_n = i_1 \to ... \to i_{n-1}, \) i.e., the first sub-path of length \( n-1 \) in \( \mathcal{P} \). The part (b) of Proposition 4 follows by observing that \( P(E_n) \) increases as \( \beta \) increases and \( \sigma \beta \) is fixed.

For part (c), i.e., blackhole peers, we use a similar strategy to that used for the part (b) by using analogous definitions of \( E_n, \mathcal{P}_n \). In this case,
\[ P(\text{A query is resolved at } i_n \mid E_n, \mathcal{P}) = \begin{cases} (1-\sigma \beta)p & \text{if } i_n \text{ does not appear in } \mathcal{P}_n \\ 0 & \text{otherwise} \end{cases} \]
\[ P(E_n) = \prod_{i \in R(\mathcal{P}_n)} (1-p)(1-\sigma+\sigma \beta |S_i(\mathcal{P}_n)|). \]
As before, this establishes part (c) of Proposition 4. \( \square \)

VI. NUMERICAL STUDY

This section shows our numerical experiments. It is known that almost all \( d \)-regular graphs form an expander family if \( d > 2 \) [26]. To create the overlay graphs (of peers) of an expander family, we used \( d \)-regular graphs with \( d = 3 \). For comparison, 2-regular graphs are also used. For each plot, \( p = 0.1, \mu = 0.001, \) and \( T_{\text{max}}(N) = N/20. \) Due to space limit, we provide the experimental results of only non-resolving and blackhole peers over \( d \)-regular graphs with \( d = 2, 3 \). We also did numerical experiments with non-regular graphs (e.g., \( d \) is uniformly distributed in \( \{2, 3, 4, 5\} \)) and obtained similar results as the \( d \)-regular graph case for \( d = 3 \).

Figure 2 and 3 depict the bounded asymptotic server load of the p2p system with non-resolving peers for 3-regular graphs and 2-regular graphs, respectively. Figure 2 shows that as in Theorem 1, the server load decreases with \( N > 1000 \) and is bounded ultimately. Note that the boundedness of asymptotic server load is still observed in 2-regular graphs in Figure 3. However, as \( d \) changes from 3 to 2, the server load increases significantly both for static and for probabilistic cases. This can be explained by the property of expander graphs (or relaxation time). Recalling the analysis in Section IV, we know that the case with non-resolving peers is identical with the fully cooperative case with small query-generating probability (i.e., from \( p \) to \((1-\sigma)p\)). In a fully cooperative p2p system, the server load is directly related to how quickly a query-issuing peer searches a peer possessing the content. The search time is closely related to the second largest eigenvalue and random walk search is effective in an expander family [6]. With this rationale, in a hybrid p2p system with non-resolving peers, the server load drastically increases for 2-regular graphs, compared with that for 3-regular graphs [3]. We took \( \beta = 0.5 \) for the case of probabilistic (blackhole or non-resolving) peers for both cases. Figure 2 and Figure 3 also show that \( s^{PNR}_{\phi} \leq s^{SNR}_{\phi} \) for \( \phi = \sigma \times \beta \), which is consistent with Proposition 4.

Fig. 2: Load on the server with non-resolving peers \( d = 3 \)

Fig. 3: Load on the server with non-resolving peers \( d = 2 \)

In Figure 4, the total load on the server \( \mu N s^{PBH}_{\phi} \) and \( \mu N s^{SNR}_{\phi} \) are depicted for static and probabilistic blackhole peers, respectively. Here, we observe that the load on the server is unbounded in both plots as Theorem 1 suggests. We took \( \beta = 0.5 \) for the case of probabilistic (blackhole or non-resolving) peers. Figure 4 shows that \( s^{PBH}_{\phi} \geq s^{SNR}_{\phi} \) for \( \phi = \sigma \times \beta \) in Proposition 4. Under blackhole peers, there is little change on the server load as \( d \) changes from 3 to 2.

Fig. 4: Load on the server with blackhole peers \( d = 3 \)

VII. SUMMARY AND CONCLUSION

This paper has analyzed the impact of selfish behaviors on the performance of content distribution of an unstructured
hybrid p2p system, which exhibits good scalability for an expander graph family when all \( N \) peers are cooperative and the time to live of a query request is designed as \( \Omega(N) \). We classified different selfish behaviors and analyzed mathematically and numerically how the asymptotic server load changes by the selfish behaviors. Our analysis revealed that non-resolving selfish behavior does not compromise the scalability while selfish behaviors of non-forwarding and direct-accessing the server cause the server load of the hybrid p2p system to increase with the number of peers \( N \). These results suggest that a hybrid system can be designed to be scalable without an incentive mechanism for content sharing. But the system does need incentive mechanisms for query request forwarding and access mechanism for content sharing. However, the system can be designed to be scalable without an incentive mechanism.

The analysis of the average time to find a peer with the content incentive mechanisms for query request forwarding and access mechanism for content sharing. But the system does need incentive mechanisms for query request forwarding and access to peers for contents. We are currently working on spectral analysis of the average time to find a peer with the content within \( T_{\text{max}} \) for the probabilistically selfish scenario and some preliminary results can be found in [27].

**APPENDIX**

**Proof of Lemma 1:** We first show that \( s(p) \) and \( T(p) \) are decreasing in \( p \). By direct differentiation,

\[
s'(p) = \sum_{B : \sharp|B| \leq N-1} \frac{f_B |B| p^{|B|-1} (1-p)^{N-1-|B|}}{A : |A| = k} \sum_{A : |A| \leq N-2} (N-1-|A|) f_A |A| (1-p)^{N-2-|A|} = \sum_{k=0}^{N-2} p^k (1-p)^{N-2-k} H(k),
\]

where

\[
H(k) = \sum_{B : |B| = k+1} (k+1) f_B - \sum_{A : |A| = k} (N-1-k) f_A.
\]

To show that \( H(k) < 0 \) for \( 0 \leq k \leq N-2 \), first note that for any \( j \in A^c \), \( x \in A^c \), \( j \neq x \),

\[
P_j(T_A > T_{\text{max}}) \geq P_j(T_{A\cup x} > T_{\text{max}}).
\]

Then, for any \( x \in A^c \),

\[
Nf_A = \sum_{j \in A^c - \{x\}} P_j(T_A > T_{\text{max}}) - \sum_{j \in A^c - \{x\}} P_j(T_{A\cup x} > T_{\text{max}})
\]

Summing both sides over all \( x \in A^c \), we have

\[
\sum_{x \in A^c} Nf_A \geq Nf_A + \sum_{x \in A^c} Nf_{A\cup x}(x) \Rightarrow (N - |A| - 1) f_A \geq \sum_{x \in A^c} f_{A\cup x}(x) \Rightarrow \sum_{A : |A| = k} (N - k - 1) f_A \geq \sum_{B : |B| = k+1} (k+1) f_B (17)
\]

Similarly, by substituting \( f_A \) by \( g_A \) and \( P_j(T_A > T_{\text{max}}) \) by \( \mathbb{E}_j \min(T_A, T_{\text{max}}) \), we can show that \( T(p) < 0 \).

Now we show the convexity of \( s(p) \). Note that \( s(0) = 1 \). So, \( s(p) \) is a convex function of \( p \) because, for any \( p > 0 \) and \( \sigma > 0 \),

\[
s((1-\sigma)p) \leq s_{SPH}((1-\sigma)p) = s(0) \frac{\sigma}{\kappa_{SPH}} + s((1-\sigma)p) \frac{(1-\sigma)p}{\kappa_{SPH}}.
\]

A similar argument works for \( T \). This completes the proof of Lemma 1. \( \square \)

**REFERENCES**


