DHT-assisted Probabilistic Exhaustive Search in Unstructured P2P Networks

Xucheng Luo, Zhiguang Qin
School of Computer Science and Engineering University of Electronic Science and Technology of China Chengdu, 610054, China
{*xucheng, qinzg*}*@uestc.edu.cn*

Jinsong Han
Department of Computer Science and Engineering Hong Kong University of Science and Technology Kowloon, Hong Kong
jasonhan@cse.ust.hk

Hanhua Chen
School of Computer Science and Technology Huazhong University of Science and Technology Wuhan, 430074, China
chenhanhua@hust.edu.cn

Abstract

Existing replication strategies in unstructured P2P networks, such as square-root principle based replication, can effectively improve search efficiency. How to get optimal replication strategy, however, is not trivial. In this paper we show, through mathematical proof, that random replication strategy achieves the optimal results. By randomly distributing rather small numbers of item and query replicas in the unstructured P2P network, we can guarantee perfect search success rate comparable to exhaustive search with high probability. Our analysis also shows that the cost for such replication strategy is determined by the network size of a P2P system. We propose a hybrid P2P architecture which combines a lightweight DHT with an unstructured P2P overlay to address the problems of network size estimating and random peer sampling. We conduct comprehensive simulation to evaluate this design. Results show that our scheme achieves perfect search success rate with quite small overhead.

1. Introduction

Since the emergence of peer-to-peer (P2P) file sharing systems, such as Napster [20] and Gnutella [2], millions of people have utilized P2P tools to share information on the Internet. Current P2P systems mainly employ three search schemes: flooding, DHT-based, and Hybrid P2P. In the first type, queries are flooded in unstructured P2P networks. The second method employs a *Distributed Hash Table* (DHT) to maintain a global index for locating items. DHT based scheme guarantees perfect successful rate but suffers from the problem of "exact match". The Hybrid P2Ps[13, 25, 3] combine an unstructured P2P overlay with a global index and items can be searched either by flooding or DHT looking up according to the item popularity (in this paper, we use "file", "file metadata", and "item" interchangeably).

Unstructured P2Ps commonly utilize replication strategies to improve the search performance. The replication strategies can be abstracted as the replication of queries or data items. Existing approaches can be divided into two categories. The first kind strategy tries to achieve a matching by issuing a large number of query replicas. Flooding, percolation search [22], and hybrid search [7] fall into this category. The other kind strategies, such as random walk based probabilistic search (RWPS) [6] and BubbleStorm [24], replicate both queries and items to achieve a better search performance.

The key issues of replication strategies in unstructured P2P networks are to estimate how many replicas of items and queries should be generated and determine how to distribute the replicas to achieve good search efficiency. Existing replication strategies for item search use query-rate based mechanism, such as square-root principle [4, 5, 27], to achieve better performance. The items with high query rate are highly replicated (for example, on the query path) for future query searching. However, the unpopular queries may have poor search efficiency because of rare item replicas. To address this problem, in this paper we show, through mathematical proof, that random replication strategy achieves the optimal search efficiency and the cost for the random replication strategy is determined by the network size of a P2P system.

We propose RandRep, a hybrid P2P architecture, to achieve the probabilistically exhaustive search. RandRep combines a lightweight DHT with an unstructured P2P overlay to support network size estimation. By nominating only a small fraction of nodes as DHT nodes, we achieve a lightweight node sampling protocol. We conduct comprehensive simulation to evaluate this design. Results show that our scheme achieves perfect search success rate with quite small overhead.

978-1-4244-1694-3/08/$25.00 ©2008 IEEE

The remainder of the paper is organized as follows. In Section 2, we discuss related work. Section 3 presents the design of RandRep. We evaluate RandRep's performance in Section 4. The conclusion is given in Section 5.

2. Related Work

Many schemes have been proposed to improve the search efficiency in decentralized P2P systems, such as [14, 10, 11, 12], and so on. Without centralized index servers, nodes cooperate with each other to perform search for desired data items. They can be categorized into four types, flooding based, DHT based, Hybrid P2P, and probabilistic search.

Flooding is commonly used in unstructured P2P networks. Flooding achieves low query latency, but is not scalable since it propagates a large number of redundant messages. Several alternatives have been proposed to address the scalability of flooding-based unstructured P2Ps. Qin Lv et al. [14] propose k-random walk to replace the flooding and significantly reduce the query replicas. Gia [2] is another promising alternative. Current search schemes in unstructured P2Ps are not exhaustive search, due to the unguaranteed success rate.

In structured P2Ps, a distributed hash table, DHT [19, 23, 26, 15, 17], is used for item locating, where an item is hashed to a node. The query is also hashed to the same node to retrieve the result. DHT based search is exhaustive because it can guarantee perfect success rate by hash an item and a related query to a rendezvous node. DHT-based schemes, however, suffer from the problem of "exact match", without support of versatile query languages, which is critical for sharing information applications in P2P networks.

Recently, Hybrid P2P architecture [13, 25, 3] attracted much attention. Based on the observation that flooding is efficient for highly replicated items while not valid for rare items, a Hybrid P2P network combines an unstructured flooding network with a structured DHT-based global index. Searches are performed either by flooding the unstructured network, or by looking up in the DHTs, according to the popularity of items. Although Hybrid P2P network improved search efficiency, how to efficiently differentiate rare items and popular items remains a challenging issue.

Probabilistic search is referred to the replication based probabilistic search schemes, in which queries and items are replicated to some nodes respectively to improve search performance. The replication strategy is crucial to this kind of search schemes. Flooding can not support exhaustive search even with heavy overall cost incurred by long Time To Live (TTL). Existing replication based search methods use replication strategies such as the square-root principle to achieve better performance. Although simulation results show the search performance is improved, how to get an optimal replication strategy is not trivial. Random walk based probabilistic search (RWPS) [6] deploys replicas by random walk. This scheme is not fault-tolerant. Bubblestorm [24] employs random multi-graph and its specific message propagating algorithm to distribute replicas. There is no explicit formal proof of the randomness of the replica deployment.

Unstructured P2Ps are the most promising scheme for content based information sharing applications, such as blog search, due to the fact that they can easily support versatile query languages. The matching operation for the query can be executed on the nodes who store the items. Therefore, any query language, such as keywords matching, XQuery [1], and so on, can be performed. In this paper, we show that random replication strategy in unstructured P2Ps achieves the optimal results using analytical method. By randomly distributed item and query replicas in unstructured P2P networks, we can guarantee exhaustive search with high probability. We also show that the cost for replication strategy is determined by the network size of P2P systems.

3. RandRep Design

In this section, we will present the design of RandRep. Before going to the details of RandRep, it is worthy to specify the network model of RandRep. Then we will present the detail designs in the estimation of the number of replicas and strategy for dispersing replicas.

3.1. Network Model

As a hybrid P2P system, RandRep maintains a lightweight DHT subsystem over unstructured P2P networks. This DHT structure makes the node selection and query (or data item) replication processed more efficiently.

In unstructured P2Ps, searching a file can be analogous with a Balls-and-Bins model [18]. Each node is a bin. For each search, queries and data items are two sets of balls with different colors. Finding the desired item in P2P is similar to a procedure that the query and data item balls are tossed into bins with a specific strategy, respectively. In large-scale P2P systems, it is not easy for requesters to know which bin contains the desired item balls and thereby toss the query ball into this bin. Intuitively, if the number of data item balls or queries is large enough, according to the Balls-and-Bins model, a collision of two kinds of balls in some bin can be guaranteed with high probability. This process is illustrated in Fig. 1.

The core of RandRep is to disperse reasonable number of replicas in the system randomly. In replication based search mechanisms, the success of a search depends on the number of replicas and the distribution of replicas. We still use Balls-and-Bins model to present the RandRep model.

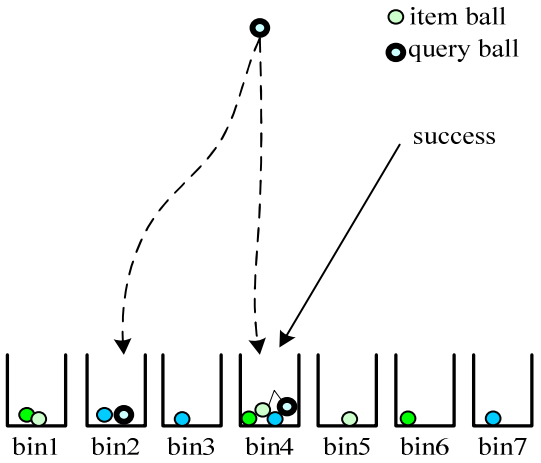

Figure 1. File search resembles a "Balls and Bins" model

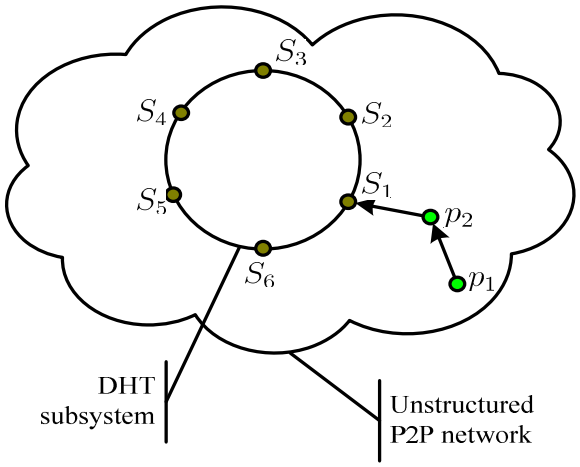

Figure 2. Model of RandRep

Suppose the topology of an unstructured P2P system is a random graph $G = (V, E)$, where V is the set of nodes and $|V| = N$. We can conceive the N nodes as N bins. Queries and data items are two kinds of colored balls for each search. According to the Balls-and-Bins model, our object is to guarantee at least one node having the query and data item simultaneously with high probability, while only a small fraction of nodes, say m in N and $m << N$, are required to be randomly chosen for two replications. For a specific data item, we replicate it to s ($s < m$) random different nodes. Correspondingly, query replicas for this data item are distributed to $q = m - s$ random different nodes. Since the data item replicas and query replicas encounter at least one node with high probability, the arbitrary matching can be executed by that node. To apply this model, three key issues should be addressed. First we must approximately estimate the network size, N. In the Balls-and-Bins model, the m and s are functions of N and thereby determined by the network size N. Second we need to choose a reasonable value of the number of data item balls or query balls for guaranteeing a search successful with high probability. The last issue is how to select the subsets of queries or data items, i.e., toss those balls to randomly chosen nodes. In the Balls-and-Bins model, randomly tossing balls is a fundamental requirement. Similarly, the replicas must be randomly distributed. Otherwise, the success of search can not be guaranteed with the given probability.

RandRep adopts hybrid architecture to address the three problems. We introduce a lightweight DHT subsystem to the unstructured P2P system. We leverage two attributes of DHT. The first is that DHT supports exact matching. The second is that DHT has a good structure for estimating the network size. The measurement on Gnutella P2P networks shows that the average lifetime of about ten percent nodes is more than 80 minutes [21]. RandRep utilize a small number of such relatively stable nodes to construct the lightweight DHT subsystem. Thus, the impact of network churn to DHT structure can be effectively constrained. Figure 2 illustrates this model. Node S_1, S_2, S_3, S_4, S_5, and S_6 are relatively stable. These nodes serve in DHT subsystem. Note that RandRep only needs a small fraction of nodes to maintain the DHT subsystem. All nodes in the P2P system register their state information in the DHT subsystem. Nodes obtain the network size estimation and random node selection service via the DHT subsystem. The replication can be also done based on the services provided by DHT subsystem. For an arbitrary node, several approaches can be used to access the DHT subsystem. Since the DHT nodes are distributed in the unstructured P2P system randomly, the messages from a node can always encounter a DHT node by random walk. For example, a message from node p_1 reaches a DHT node S_1 by two random walk steps in Fig. 2. To reduce the latency of accessing DHT subsystem, nodes cache some DHT nodes. By means of probing these nodes randomly, nodes can access the DHT rapidly. Another method is to utilize the DNS to map the DHT nodes.

The functionality of DHT in RandRep is different from other Hybrid P2Ps. In previous Hybrid P2Ps, DHT is used for indexing and querying rare items. In RandRep, DHT stores the state information of nodes, such as the degree, the size of the shared storage, the number of shared files, and the like. These data are generally rare items such that the effects of exact matching and versatile query are similar. It is reasonable to store these data in the DHT subsystem for query efficiency.

3.2. Network Size Estimation

The DHT subsystem provides a network size estimation service. Through query DHT, nodes can get the network size estimate. If the DHT nodes are distributed in the ID space uniformly random, each DHT node has equal length ID partition. This length is denoted as α. Thus, the number of DHT nodes is $n = L/\alpha$, where L is the length of ID space, for example, $L = 2^{160}$. If all nodes in the P2P system are also mapped onto the ID space uniformly random, each DHT node obtains the same number of normal P2P nodes, which is denoted as ω. Therefore, the network size can be estimated as $N = n\omega$. However, the lengths of ID partitions may not be the same in practice, The longest partition is $\Theta(\log L/L)$ [16] and the shortest partition is $\Theta(1/L^2)$ [8]. This large diversity may incur inaccurate estimations on the network size.

RandRep estimates the network size through the information aggregated from several DHT nodes. For a node i in DHT, it has local observation $< l_i, x_i >$, where the length of its ID partition is l_i and the number of data items stored on node i is x_i. If we collect a number of DHT nodes and aggregate their $< l_i, x_i >$, we have $l = \sum l_i$ and $x = \sum x_i$. The network size thereby can be estimated as $\hat{N} = \frac{L}{l}x$. To bound $\hat{N}$, the following theorem is given.

Theorem 1. *For any* $\sqrt{\frac{4L\varepsilon \ln N}{lN}} \leq \delta \leq 2e-1$ *, the probability of* $(1-\delta)N \leq \hat{N} \leq (1+\delta)N$ *is* $1-2N^{-\varepsilon}$*, where* $\varepsilon > 0$ *is a constant.*

Proof. The size of ID space is L. The size of aggregated ID subspace of a DHT node is l. We use an Indicator Variable Z_i to mark a node mapped to the ID subspace or not.

$$Z_i = \begin{cases} 1 & \text{hit an ID in the subspace} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Since nodes randomly choose ID from the ID space, the probability that a node is mapped into the aggregated ID subspace is $P[Z_i = 1] = \frac{l}{L}$.

We denote the random variable $Z = \sum_{i=1}^{N} Z_i$. Since Z_i is independent, the expectation of Z is given by

$$E[Z] = \sum_{i=1}^{N} E[Z_i] = \frac{l}{L}N \tag{2}$$

According to the Chernoff inequality [18], for arbitrary $0 < \delta \leq 2e-1$

$$P\left[Z < (1-\delta)E[Z]\right] < e^{-\frac{\delta^2}{4}E[Z]} \tag{3}$$

$$P\left[Z > (1+\delta)E[Z]\right] < e^{-\frac{\delta^2}{4}E[Z]} \tag{4}$$

We assume $e^{-\frac{\delta^2}{4}E[Z]} \leq N^{-\varepsilon}$, where $\varepsilon > 0$. Then,

$$\delta \geq \sqrt{\frac{4L\varepsilon \ln N}{lN}} \tag{5}$$

Thus, if $\sqrt{\frac{4L\varepsilon \ln N}{lN}} \leq \delta \leq 2e-1$, we have $P\left[(1-\delta)E[Z] \leq Z \leq (1+\delta)E[Z]\right] \geq 1-2N^{-\varepsilon}$. According to Eq.(2), replacing $E[Z]$ in it, we get $P\left[(1-\delta)N \leq \frac{L}{l}Z \leq (1+\delta)N\right] \geq 1-2N^{-\varepsilon}$. That is, $P\left[(1-\delta)N \leq \hat{N} \leq (1+\delta)N\right] \geq 1-2N^{-\varepsilon}$ □

To estimate the network size more accurate, we expect to minimize the δ according to the required accuracy and probability, and further determine the value of l . According Eq.(5), the minimum value of δ is $\delta = \sqrt{\frac{4L\varepsilon \ln N}{lN}}$. Thus, the value of l is $l = \frac{4\varepsilon \ln N}{\delta^2 N}L$. For example, if $\delta = 1/20$, $\varepsilon = 1/2$, and $N > 90,000$, $l = L/10$ is sufficient. We present the algorithm for estimating the network size in Algorithm 1. In RandRep's DHT, the successor of node i is $successor(i)$.

Algorithm 1 Network Size Estimation

```
1: Initialization: l ← 0, x ← 0
2: SizeEstimation(i, < l, x >)
3: l ← l + l_i
4: x ← x + x_i
5: if l ≥ L/10 then
6:     return x × L/l
7: else
8:     return SizeEstimation(successor(i), < l, x >)
9: end if
```

In this way, each DHT node can estimate the network size from its aggregated $< l, x >$. The aggregation can be achieved via DHT routing. The message overhead and query latency are reasonable due to the small amount of nodes required to communicate.

3.3. Random Node Subset Selection

Random node subset can also be obtained through DHT subsystem. If a node wants to select a random node, this node firstly generates a random ID in ID space. Then, the node queries DHT for this ID. If this ID has been assigned to some node, this node is selected and DHT returns the node's IP address. Otherwise, the first node after this ID is selected. Note that the probability of selecting one node more than once is negligible. This is guaranteed by the feature of collision resistance of hash functions. If the size of random node subset is s, s random IDs should be generated. In the DHT subsystem, the message overhead of each query is $O(\log n)$, where n is the number of nodes in

the DHT subsystem. Correspondingly, to select s random nodes, the message overhead is $O(s \log n)$ and the latency is $O(s \log n)$.

To further reduce the message overhead and latency, we optimize the random node selection process. Firstly, the requested DHT node generates a random ID set. Then, this node sorts these IDs to get an ordered ID list. Since the IDs are generated randomly, these IDs are distributed on the DHT uniformly. The requested DHT node divides the ordered ID list into $\log n$ sections according to the neighbor list. Each neighbor takes charge one section for query. By this way, each message can solve one random node selection. Suppose the size of the DHT subsystem is the same as the size of random node subset, each DHT node should generate one random node to construct the random node subset. Through the DHT ring, this operation can be performed. If the size of DHT subsystem is larger than the random node subset, through the finger table, one message can also solve one random node. In short, to select s random node collectively, about s messages are sufficient. The parallel queries by utilizing the neighbor list can further reduce the query latency.

3.4. Deployment Replicas

To guarantee the search success with high probability, the number of replicas must be carefully determined. Let r and q denote the numbers of item replicas and query replicas, respectively. We propose the following theorem for choosing r and q.

Theorem 2. *If* $rq \geq N\left((1+\varepsilon \ln N) + \sqrt{(1+\varepsilon \ln N)^2 - 1}\right)$, *the probability of two kinds of replicas encountering in at least one node is* $P \geq 1 - N^{-\varepsilon}$, *where* $\varepsilon > 0$ *is a constant.*

Proof. For each query replica, the probability of encountering an item replica is $p = r/N$. Let Indicator Variable X_i denote the result of the ith search. Then

$$X_i = \begin{cases} 1 & \text{hit a resource replica} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Therefore,

$$P[X_i = 1] = \frac{r}{N} \tag{7}$$

We denote random variable X as $X = \sum_{i=1}^{q} X_i$. Since X_i is independent, the expectation of X is given by

$$E[X] = \sum_{i=1}^{q} E[X_i] = \frac{rq}{N} \tag{8}$$

According to Chernoff inequality [18], for $0 < \varepsilon \leq 1$,

$$P\left[X < (1-\delta)E[X]\right] < e^{-\frac{\delta^2}{2}E[X]} \tag{9}$$

To meet the design target, the following equations are given.

$$\begin{cases} e^{-\frac{\delta^2}{2}E[X]} = N^{-\varepsilon} \\ (1-\delta)E[X] = 1 \end{cases} \tag{10}$$

where $\varepsilon > 0$ is a constant. When $rq \geq N\left((1+\varepsilon \ln N) + \sqrt{(1+\varepsilon \ln N)^2 - 1}\right)$, we have $P[X < 1] < N^{-\varepsilon}$. That is, if $rq \geq N\left((1+\varepsilon \ln N) + \sqrt{(1+\varepsilon \ln N)^2 - 1}\right)$, $P[X \geq 1] \geq 1 - N^{-\varepsilon}$.

□

Parameters r and q have impact on the overhead of item replication and query replication. If r and q are all larger than the given thresholds, the search success can be obviously guaranteed. Enlarging them, however, will increase the replication cost. To reduce the overhead, the rq should be minimized. Since $2N(1+\varepsilon \ln N)$ approximates $N\left((1+\varepsilon \ln N) + \sqrt{(1+\varepsilon \ln N)^2 - 1}\right)$, the minimum of rq is $2N(1+\ln N)$. The value of rq is also determined by the parameter ε, which also impacts the search success rate. Usually, the network size is larger than 1, 000. For $\varepsilon = 1$, the search success probability is more than 0.999. For example, if $N = 1,000,000$ and $\varepsilon = 1$, $P[X \geq 1] \geq 0.999999$. P2P systems are dynamic and nodes join and leave frequently. If nodes with replicas leave the system, the number of replicas is decreased. Correspondingly, the probability of search success is reduced. To constrain the decrease caused by such a network churn, replicas should be re-distributed to random nodes periodically to keep the required amount. The amount of added replicas is closely related to the amount of disappeared replicas caused by the network dynamics and the current network size. It is obvious that the replica maintenance overhead is highly related to the network dynamics. Assume the lifetime of an item is T, the number of additional replicas is a function of T, which is denoted as $f(T)$. If the bandwidth overhead of a resource replica is C_r, the overhead of resource replication is $C_r(r + f(T))$. Query overhead, on the other hand, is related to the query frequency in the resource lifetime. This frequency is denoted as $h(T)$. If the bandwidth overhead of a query replica is C_q, the query overhead for this resource is $C_q q h(T)$. The sum of all overhead is given by

$$C = C_r(r + f(T)) + C_q q h(T) \tag{11}$$

The problem of determining r and q converts to the optimization problem as below.

$$\begin{aligned} &\min C = C_r(r + f(T)) + C_q q h(T) \\ &s.t.\ rq = 2N(1+\varepsilon \ln N) \end{aligned} \tag{12}$$

Then, we have

$$C = rC_r + \frac{2N(1+\varepsilon \ln N)C_q h(T)}{r} + C_r f(T) \quad (13)$$

The derivative of C is given below.

$$C' = C_r - \frac{2N(1+\varepsilon \ln N)C_q h(T)}{r^2} \quad (14)$$

Let C' equal to zero. We have $r = \sqrt{2N(1+\varepsilon \ln N)C_q h(T)/C_r}$. Under this condition, the second order derivative of C is given below.

$$C'' = \frac{4N(1+\varepsilon \ln N)C_q h(T)}{r^3}\bigg|_r > 0 \quad (15)$$

where $r = \sqrt{2N(1+\varepsilon \ln N)C_q h(T)/C_r}$. Therefore, when $r = q = \sqrt{2N(1+\varepsilon \ln N)C_q h(T)/C_r}$, the total overhead is minimized. The minimum of the total overhead is $2\sqrt{2N(1+\varepsilon \ln N)C_r C_q h(T)} + C_r f(T)$.

The specific solution of this problem depends on the real applications. If the resource replica is the metadata of the resource, it is reasonable that $C_r \approx C_q$. For a simple case, if $f(T) = 0$ and $h(T) = 1$, we have $r = q = \sqrt{2N(1+\varepsilon \ln N)}$. Under this parameter setting, the minimum overhead is obtained, which is $2\sqrt{2N(1+\varepsilon \ln N)C_r C_q}$.

After determining the parameters r and q, the replicas should be distributed to random nodes. For item replication, r random nodes are selected through the DHT subsystem and the item replicas are distributed to these nodes. The query node selection and replication is performed by the requesting node in same way. According to Eq. 8, the expected value of collisions is $E[X] = 2(1 + \ln N)$. However, only one hit is needed. All other hits are redundant. To reduce the redundancy, a requesting node divides the q chosen random nodes into subgroups and the query is replicated in multiple rounds. In each round, query is only replicated within one subgroup. If there are no answers, the querying node chooses another untouched subgroup for replication in the next round. After all q query replicas have distributed, if there is still no answer, the probability of no item matching the query is very high. The size of each subgroup is g and the size of the last subgroup may be smaller than g.

To reduce the query overhead, we expect to minimize the number of collisions of item replica and query replica (one is enough). It is reasonable that the size of subgroup (or number of query replicas in each round) is the average of query replica amounts for first collision with the item replica. Random variable Y denotes the number of query replicas being distributed when first hit occurs. Y satisfies the Geometric distribution.

$$P[Y = k] = p(1-p)^{k-1} \quad (16)$$

where $p = \frac{r}{N}$. Therefore, $E[Y] = \frac{1}{p} = \frac{N}{r}$.

For query replication, it is reasonable to set the size of the subgroup $g = \lceil E[Y] \rceil$. By this way, network overhead can be effectively reduced. For a P2P network with 1, 000, 000 nodes, the values of r and g is $r = 3,977$, $g = 252$, respectively.

4. Performance Evaluation

In this section, we present the metrics, experiment setup, and performance evaluation of our design.

4.1. Methodology

The main metrics for evaluating search performance include success rate, query latency, and message overhead. The search success rate represents the ratio between the number of nodes with successfully returned results and the number of nodes which issue the query. If there are y nodes query a specific item and x nodes obtain the answer, the success rate is $h = x/y$. The success rate is the most important measurement of search performance, especially for unpopular items. We consider the success rate in two scenarios. The first has no node error in the system. In the second scenario, the node failure appears according to a probability. This scenario evaluates the fault-tolerance of search schemes. Query latency is measured by the maximum hops required to propagate query messages to a node with a replica. As to replication based schemes, the latency is composed of node selection latency and query replication latency. Message overhead is the sum of messages propagated in the system. In replication based schemes, this message overhead is composed of node selection message overhead, item replication message overhead, and query replication message overhead. As for the entire system, high success search rate, short query latency, and low message overhead are desired.

Simulation setup mainly includes the topologies and item placement. The topology of unstructured P2P systems was reported to be a power law random graph [9]. In simulation, we used power law random graphs with 50, 000 nodes, 100, 000 nodes, 150, 000 nodes, 200, 000 nodes, 250, 000 nodes, respectively. Since the DHT subsystem is a lightweight DHT, in our simulation, five percent nodes are selected as the DHT nodes. The Chord [23] protocol is used to connect the DHT nodes. Each item has only one copy shared by some random node.

4.2. Results

In unstructured P2P systems, two major algorithms with exhaustive search and versatile query language support are flooding and RWPS [6]. For comparison, we implement the

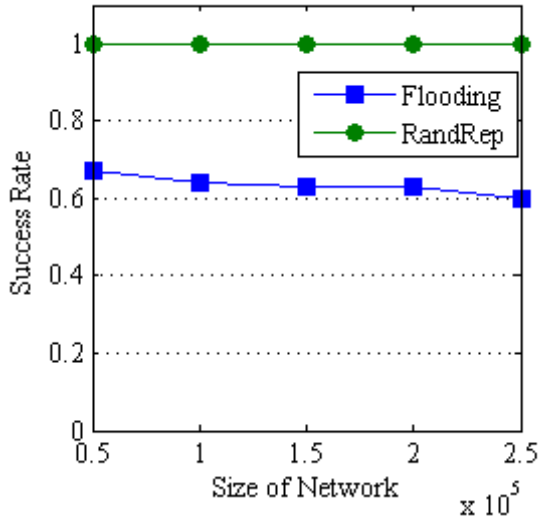

Figure 3. The search success probability of RandRep approximates 100%, while that of Flooding search scheme is a little more than 60%

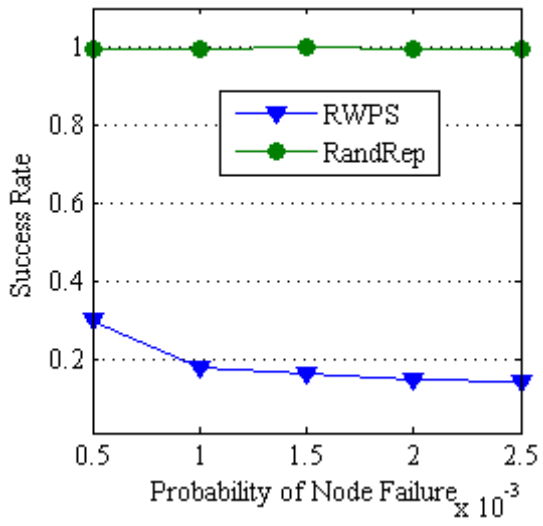

Figure 4. If nodes fail probabilistically, search success rate of RandRep is far higher than that of RWPS

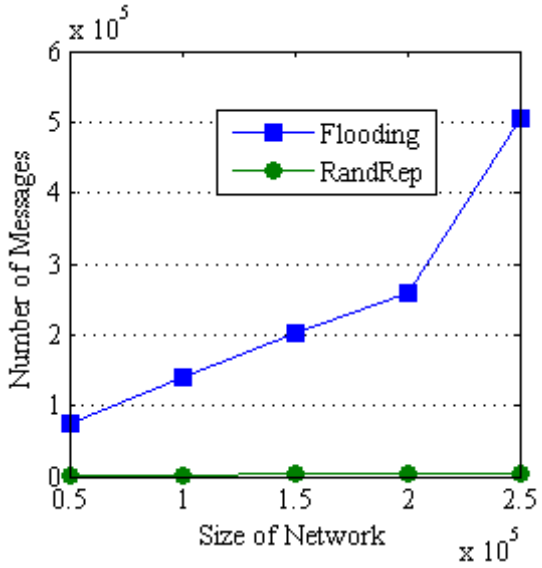

Figure 5. The number of messages of Flooding search is far larger than that of RandRep

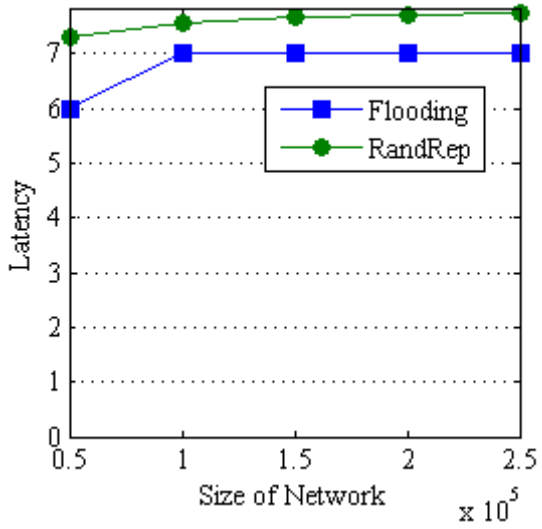

Figure 6. The average query latency of RandRep is little more than that of Flooding

two algorithms in our simulator. Each simulation is executed 10, 000 times and the averages are counted.

We first investigate search success rate in two scenarios. Figure 3 plots the success rate in the scenario where there is no node failure. RandRep and flooding are evaluated in this scenario. Since RandRep and RWPS employ similar principle, the search success rate of RWPS in this scenario resembles that of RandRep. The success probability of RandRep approximates 100%. The success rate of Flooding search is a little more than 60%. Figure 4 shows the search success rate in scenario where nodes fail probabilistically. We compare RandRep with RWPS in this scenario. The search

success rate of RandRep approximates 100%, while that of RWPS is less than 40%. As the failure probability increases, the search success rate of RWPS is decreased. This result shows that RandRep is fault-tolerant.

The query message overhead is given in Fig. 5. Since RandRep and RWPS employ similar principle, we only compare RandRep with Flooding search. RandRep significantly reduces the message overhead compared with Flooding search. More messages incur the congestion in message propagation. Some messages are failed to be forwarded. Therefore, the search success rate is decreased. According to the aforementioned, RandRep is more practical.

Figure 6 plots the query latency. Since RWPS employ random walk to replicate the items and queries, the latency is very high. We only compare latencies of RandRep and Flooding search. The query latency of RandRep is a little more than that of Flooding search. In real applications, this latency is acceptable.

According to the above results, RandRep achieves the high success rate and low message overhead. The search latency of RandRep is a little more than that of Flooding search, but this latency is acceptable in real applications.

5. Conclusion

In this paper, we propose a hybrid P2P system, RandRep, to provide both the versatile query language support and probabilistically exhaustive search. A small fraction of stable nodes are selected to construct a lightweight DHT subsystem. This subsystem enables a simple and effective way for the network size estimation and random node selection. The well designed parameters of RandRep guarantee perfect search success rate. The simulation results show that RandRep outperforms existing approaches with a significant improvement in terms of search success rate and message overhead.

6. Acknowledgements

This work was supported by NSFC grant No. 60473090 and No. 60573129. We thank anonymous reviewers for their valuable comments.

References

[1] D. Chamberlin. Xquery: a query language for xml. In *Proceedings of ACM SIGMOD 2003*, pages 682–682, San Diego, California, USA, 2003. ACM.

[2] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like p2p systems scalable. In *Proceedings of ACM SIGCOMM 2003*, pages 407–418, Karlsruhe, Germany, 2003. ACM.

[3] H. Chen, H. Jin, Y. Liu, and L. M. Ni. Difficulty-aware hybrid search in peer-to-peer networks. In *Proceedings of the ICPP 2007*, page 6, Xi'An, China, 2007. IEEE.

[4] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proceedings of ACM SIGCOMM 2002*, pages 177–190, Pittsburgh, PA, USA, 2002. ACM.

[5] B. F. Cooper. Quickly routing searches without having to move content. In *Proceedings of IPTPS 2005*, pages 163–172, Ithaca, NY, USA, 2005. Springer.

[6] R. A. Ferreira, M. K. Ramanathan, A. Awan, A. Grama, and S. Jagannathan. Search with probabilistic guarantees in unstructured peer-to-peer networks. In *Proceedings of P2P 2005*, pages 165–172, Konstanz, Germany, 2005. IEEE.

[7] C. Gkantsidis, M. Mihail, and A. Saberi. Hybrid search schemes for unstructured peer-to-peer networks. In *Proceedings of IEEE INFOCOM 2005*, pages 1526–1537, Miami, FL, USA, 2005. IEEE.

[8] V. King and J. Saia. Choosing a random peer. In *Proceedings of ACM PODC 2004*, pages 125–130, St. John's, Newfoundland, Canada, 2004. ACM.

[9] J. Kleinberg. The small-world phenomenon: an algorithmic perspective. In *Proceedings of ACM STOC 2000*, pages 163–170. ACM, 2000.

[10] Y. Liu, X. Liu, L. Xiao, L. M. Ni, and X. Zhang. Location-aware topology matching in p2p systems. In *Proceedings of IEEE INFOCOM 2004*, pages 2220–2230, Hong Kong, 2004. IEEE.

[11] Y. Liu, L. Xiao, and L. M. Ni. Building a scalable bipartite p2p overlay network. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 18(9):1296–1306, 2007.

[12] Y. Liu, Z. Zhuang, L. Xiao, and L. M. Ni. a distributed approach to solving overlay mismatching problem. In *Proceedings of IEEE ICDCS 2004*, pages 132–139. IEEE, 2004.

[13] B. T. Loo, R. Huebsch, I. Stoica, and J. M. Hellerstein. The case for a hybrid p2p search infrastructure. In *Proceedings of IPTPS 2004*, pages 141–150, La Jolla, CA, USA, 2004. Springer.

[14] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proceedings of ACM ICS 2002*, pages 84–95, New York,USA, 2002. ACM.

[15] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: a scalable and dynamic emulation of the butterfly. In *Proceedings of ACM PODC 2002*, pages 183–192, Monterey, California, USA, 2002. ACM.

[16] G. S. Manku. Routing networks for distributed hash tables. In *Proceedings of ACM PODC 2003*, pages 133–142, Boston, Massachusetts, USA, 2003. ACM.

[17] P. Maymounkov and D. Mazieres. Kademlia: a peer-to-peer information system based on the xor metric. In *Proceedings of IPTPS 2002*, pages 53–65, Cambridge, MA, USA, 2002. Springer.

[18] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, Cambridge, 1995.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. a scalable content-addressable network. In *Proceedings of ACM SIGCOMM 2001*, pages 161–172, San Francisco, California, USA, 2001. ACM.

[20] S. Saroiu, K. Gummadi, and S. Gribble. Measuring and analyzing the characteristics of napster and gnutella hosts. *Multimedia Systems Journal*, 9(2):170–184, 2003.

[21] S. Saroiu, P. K. Gummadi, and S. D. Gribble. a measurement study of peer-to-peer file sharing systems. In *Proceedings of MMCN 2002*, pages 156–170, San Jose, USA, 2002. SPIE.

[22] N. Sarshar, P. O. Boykin, and V. P. Roychowdhury. Percolation search in power law networks: Making unstructured peer-to-peer networks scalable. In *Proceedings of IEEE P2P 2004*, pages 2–9, Zurich, Switzerland, 2004. IEEE.

[23] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup service for internet applications. In *Proceedings of ACM SIGCOMM 2001*, pages 149–160, San Diego, California, USA, 2001. ACM.

[24] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann. Bubblestorm: Resilient, probabilistic, and exhaustive peer-to-peer search. In *Proceedings of ACM SIGCOMM 2007*, pages 49–60, Kyoto, Japan, 2007. ACM.

[25] M. Zaharia and S. Keshav. Gossip-based search selection in hybrid peer-to-peer networks. In *Proceedings of IPTPS 2006*, Santa Barbara, CA, USA, 2006.

[26] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: a resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22(1):41–53, 2004.

[27] M. Zhong and K. Shen. Popularity-biased random walks for peer-to-peer search under the square-root principle. In *Proceedings of IPTPS 2006*, Santa Barbara, CA, USA, 2006.