Random Choices for Churn Resilient Load Balancing in Peer-to-Peer Networks

Song Fu and Cheng-Zhong Xu
Dept. of Electrical & Computer Engineering
Wayne State University, Detroit, MI
{song, czxu}@eng.wayne.edu

Haiying Shen
Dept. of Computer Science & Engineering
University of Arkansas, Fayetteville, AR
hshen@uark.edu

Abstract

Peer-to-peer (P2P) networks based on consistent hashing functions have an inherent load uneven distribution problem. Things are even worse in unstructured P2P systems. The objective of load balancing in P2P networks is to balance the workload of the network nodes in proportion to their capacity so as to eliminate traffic bottleneck. It is challenging because of the dynamic nature of overlay networks and time-varying load characteristics. Random choices schemes can balance load effectively while incurring only a small overhead, making such schemes appealing for practical systems. Existing theoretical work analyzing properties of random choices algorithms can not be applied in the highly dynamic and heterogeneous P2P systems. In this paper, we characterize the behaviors of randomized search schemes in the general P2P environment. We extend the supermarket model by investigating the impact of node heterogeneity and churn to the load distribution in P2P networks. We prove that by using d-way random choices schemes, the length of the longest queue in P2P systems with heterogeneous nodal capacity and node churn for $d \geq 2$ is $c \log \log n / \log d + O(1)$ with high probability, where c is a constant.

Keywords: Randomized probing; Peer-to-Peer networks; Load balancing; Heterogeneous and bounded nodal capacity; Churn.

1 Introduction

Peer-to-peer (P2P) networks have become, in a short period of time, one of the fastest growing and most popular Internet applications. An important class of the P2P overlay networks is distributed hash tables (DHTs) that map keys to nodes of a network based on a consistent hashing function. Representatives of the DHTs include Chord [26], Pastry [21], Tapestry [32], CAN [19], and Cycloid [24]. In a DHT, each node and key has a unique Id, and a key is mapped to a node according to DHT definition. The Id space of a DHT is partitioned among nodes and each of them is responsible for those keys whose Ids are located in its space portion. An important goal in the design of DHTs is to achieve a balanced partition of the hash space among peer nodes. It is often desirable that each node assumes responsibility for a portion of the hash space that is proportional to its power, measured in terms of its processor speed, available bandwidth, and/or storage capacity, and that this property is maintained as nodes join and leave the system. A similar goal is desirable in unstructured P2P networks as well.

However, consistent hashing [26] produces a bound of $O(\log n)$ imbalance of keys between nodes, where n is the number of nodes in the system. Things are even worse in unstructured P2P systems, where no commonly-accepted load distribution mechanisms are built in. In addition, users may query geographically close nodes and those that have popular files. These lead to imbalanced distribution of workload among peer nodes. When a node become overloaded, it can not store any other files or respond to user queries any more, which affects system utilization and users' satisfaction. To balance load among peer nodes in a P2P network, lightly loaded nodes need to be selected to store files or service queries. Load balancing in P2P networks is an important topic and many research works have been conducted focusing on it recently [29, 22, 33, 9, 2].

It is well known that simple randomized load balancing schemes can balance load effectively while incurring only a small overhead, making such schemes appealing for practical systems. The paradigm of multiple random choices was used in [22, 9, 11, 34, 3]. Several peer nodes are probed before storing a file to or dispatching a user query to the least loaded one. Random choice algorithms are scalable and they require little control messages and data structures [22, 3]. More importantly, they work in P2P systems with churn, a situation where a large percentage of nodes join and leave continuously and rapidly, leading to unpredictable network size.

To theoretically analyze the effectiveness of random choices schemes in balancing load of distributed systems, researchers have proposed several techniques. Azar et al. [1] introduced the *layered induction* approach, where the random choice problem was modeled by balls-and-bins. It provides nearly tight results. The *witness tree* method was used by Cole et al. [6] to handle the random choices problem. The probability of the certain event can then be bounded by the probability of occurrence of a witness tree. Generally, witness tree arguments involves the most complexity, and they have proved to be the most challenging in terms of obtaining tight results. The *fluid limit* model [16, 17] characterize the system dynamics by constructing a family of differential equations. This approach is simple and flexible. When the system dynamics can be modeled by this method, the differential equations generally yield accurate numerical results.

However, these theoretical work analyzed a system where compute nodes have homogeneous and infinite capacities. Moreover, node churn, a defining characteristic of P2P systems, is not modeled by these approaches. As a result, we can not directly argue that the performance bounds derived in these work are still valid in the P2P networks. In this paper, we analyze the dynamic behavior of random choice paradigm in general P2P systems, where peer nodes join/leave at runtime and they have heterogeneous and bounded capacities.

We model dynamic P2P systems, where load queries arrive as a Poisson stream at a collection of n peer nodes, where n is a random variable reflecting nodal churn. Nodes are heterogeneous with bounded capacity. For each query, a number of d nodes are chosen independently and uniformly at random, and query is queued at the node currently containing the fewest queries. We refer to such multiple choices query as d-way probing. Queries arrive to peer nodes at rate λ relative to the node population. They are served according to the FIFO protocol, and the service time for a query is exponentially distributed with mean 1.

We extend the supermarket model [16] to formulate behaviors of the preceding dynamic system in the general case and quantify system properties. We are interested in characterizing the average response delay and the maximum load among active nodes. However, quantifying these metrics in a general P2P system is nontrivial. It's difficult to find closed-form solutions to the differential equations describing system dynamics, after we remove the restrictions of static system configuration, homogeneous and infinite node capacities. Instead of solving the equations directly, we analyze the lower and upper bounds of state variables at equilibrium points with reference to those in a special case. Based on these bounds, we quantify the average response delay and the maximum load, and come to the following conclusions of d-way probing in P2P networks:

Theorem 1 *For any fixed time interval I, the expected time that a query spends in the dynamic P2P system with* $d \geq 2$*, denoted by* $T_d(\lambda)$*, over interval* $[0,\ I]$ *satisfies that* $\frac{T_d(\lambda)}{\log T_1}$ *is close to* $\alpha\frac{1}{\log d}$*, for* λ *close to 1, where* α *is a constant depending on capacities of peer nodes and the change rate of the node population.*

Theorem 2 *For any fixed time interval I, the length of the longest queue in the dynamic P2P models with* $d \geq 2$ *over interval* $[0,\ I]$ *is* $c\log\log n\ + O(1)$ *with high probability* $(1-O(\frac{1}{n}))$*, where c is a constant depending on capacities of peer nodes, d and the arrival rate of queries, and the* $O(1)$ *term depends on I and some constants.*

These theorems show that two-way randomized probing is asymptotically superior to the one-way choice approach. However, by increasing the number of choices further, efficiency of the search algorithm does not improve significantly. They are consistent with the findings in the supermarket model [16], where the number of servers are fixed, their capacities are homogeneous and infinite. Simulation and experiment results confirm the correctness of our findings. Although the randomized probing algorithms for load balancing are designed and analyzed within the context of P2P networks, the results have wide applicability and are of interest beyond the specific applications.

The remainder of this paper is organized as follows: The basic supermarket model is briefly introduced in Section 2. Section 3 formulates the d-way randomized probing in P2P networks. We investigate the influences of nodal capacity in Section 3.1 and nodal churn in Section 3.2. By analyzing the equilibrium points of the system, we quantify the expected time of a query spending in the system and the length of the longest queue among peer nodes. Experimental results are shown in Section 4. Section 5 presents the related work. Conclusions are made in Section 6.

2 Basic Supermarket Model

A load balancing scheme distributes user requests or storage loads among compute nodes and avoids hot spots. In [16], Mitzenmacher presents a supermarket model based on differential equations to analyze both static and dynamic load balancing strategies. In this section, we briefly describe this model, and in the subsequent sections we will present our extension of the model to formalize randomized probing algorithms for load balancing in general P2P systems.

Supermarket model analyzes balancing workload in a special distributed environment. User requests arrive as a Poisson stream at a collection of servers. For each request, some constant number of servers are chosen independently and uniformly at random with replacement from the servers, and request waits for service at the server currently containing the fewest requests. Requests are served according to the FIFO protocol, and the service time for a request is exponentially distributed with mean 1. Three underlying assumptions were made: (*a*) unbounded server capacities, (*b*) static server configuration, and (*c*) homogeneous servers. The author derived the average time of a request staying in the system and the maximum workload of the servers by solving the differential equations of system states.

3 Randomized Probing in General P2P Systems

In large-scale P2P systems, a great number of nodes share resources and issue queries to each other. More often than not, they have heterogeneous configurations of storage capacity and processing speed. In addition, dynamics is a defining characteristic of P2P networks, with nodes joining and leaving frequently. Load balancing in such large-scale and dynamic distributed environments is challenging. Obtaining the capacity information of all active nodes before dispatching jobs to the most lightly-loaded nodes is expensive. Randomized probing is a remedy to this problem.

By applying randomized probing algorithms, we make dispatch decisions based on the load dynamics of a small number of nodes that are selected randomly. In this way, the number of load query messages that are exchanged is reduced significantly. The scalability of these algorithms is ensured because the number of control messages for each decision making is almost constant even when the system scale expands. However, theoretically analyzing behaviors of these algorithms in such general cases is challenging. Nodal heterogeneity and churn may make it intractable. In this section, we extend the supermarket model to formulate behaviors of randomized probing algorithms in general P2P environments and to quantify system dynamics with regards to the nodal workload and average response time based on our extended models. Our extension is made in two orthogonal dimensions: server capacity (from homogeneous and unbounded case to heterogeneous and bounded case) and node dynamics (from static configuration to dynamic configuration).

3.1 Heterogeneous and Bounded Node Capacity

In the basic supermarket model, as described in Section 2, servers are modeled as homogeneous with unbounded capacity. However, in practical P2P systems, peer nodes have limited and different storage capacity and processing speed. In this section, we analyze behaviors of random choice algorithms for load balancing in P2P networks where nodes have heterogeneous and bounded capacity. We extend the basic supermarket model in two steps: bounding node capacity in a homogeneous environment (Section 3.1.1) and then heterogenizing node capacity (Section 3.1.2). Here we assume the static composition of peer nodes. A P2P network consists of N nodes. Node churn or dynamic composition will be studied in Section 3.2.

3.1.1 Homogeneous and Bounded Case

In a homogeneous P2P system with bounded nodal capacity, we use C to denote the uniform capacity of peer nodes. C is measured as the maximum number of queries that a node can queue at runtime. When a node receives a query from a peer, it services the query if there are extra capacity to handle it. Otherwise, it drops this query by its admission controller. Therefore, we adopt the saturation policy as follows. A query is turned down when all of the d nodes, selected randomly and independently, are saturated, i.e. their load equals to their capacity.

Let $n_i(t)$ denote the number of nodes queuing i queries at time t; $m_i(t) = \sum_{k=i}^{C} n_k(t)$, i.e. the number of nodes queuing at least i queries at time t; $p_i(t) = n_i(t)/n$ be the fraction of nodes that have queues of size i; $s_i(t) = \sum_{k=i}^{C} p_k(t) = m_i(t)/n$ be the tails of the $p_i(t)$. We drop the reference to t in the notation where the meaning is clear. The s_i is more convenient to work with than the p_i. In an empty system, which corresponds to one with no query, $s_0 = 1$ and $s_i = 0$ for $1 \leq i \leq C$. The expected number of queries per node at any time t is $\sum_{i=1}^{C} s_i(t)$, and it's finite because each node can queue at most C queries at a time.

The state of the system at any given time can be represented by an finite vector $\vec{s} = [s_0, s_1, \ldots, s_C]$. It contains information regarding the number of nodes queuing each size of queries. We can derive the maximum load of peer nodes and the average response time of queries, based on the dynamics of this state information. This resulting model can be considered as a Markov chain on the above state space.

We now extend the basic supermarket model to formulate and analyze randomized probing among peer nodes with homogeneous and bounded capacity. The time evolution of the P2P system is specified by the following set of differential equations:

$$\left\{\begin{array}{l} \dot{s}_i \;=\; \lambda(s_{i-1}^d - s_i^d) - (s_i - s_{i+1}), \quad \text{for } \; i < C \\ \dot{s}_C \;=\; \lambda(s_{C-1}^d - s_C^d) - s_C \\ s_0 = 1 \end{array}\right. \tag{3.1}$$

where $\dot{s}_i$ denotes ds_i/dt.

Let us explain the reasoning behind the system (3.1). Consider a P2P network with N nodes, and determine the expected change in the number of nodes with at least i queries over a small period of time of length dt. The probability a query arrives during this period is $\lambda N dt$, and the probability an arriving query is dispatched to a node queuing $i-1$ queries is $s_{i-1}^d - s_i^d$, i.e. all of the d nodes chosen by the new query are of size at least $i-1$, but not all of size at least i. The probability a query leaves a node of size i in this period is $N(s_i - s_{i+1})dt$, for $i < C$. Because each node can server no more than C queries at a time, the probability a query leaves a node of size C in this period is $Ns_C dt$.

Next, we try to find the equilibrium points of (3.1). At the equilibrium points, the volume of incoming queries equals to the volume of outgoing queries, i.e. $\dot{s}_i = 0$. For a special case $d = 1$, system (3.1) becomes stable at states

$$\pi_i = \frac{\lambda^i - \lambda^{C+1}}{1 - \lambda^{C+1}}, \quad \text{for } 1 \le i \le C.$$

We denote the expected time a query spends in the P2P system with homogeneous and bounded-capacity nodes by $T_d(\lambda)$. As mentioned before, the probability that an incoming query arriving at time t becomes the ith query in the queue is $s_{i-1}(t)^d - s_i(t)^d$. Therefore, the expected time a query that arrives at time t spends in the system is $T_d(\lambda) = \sum_{i=1}^{C} i(s_{i-1}(t)^d - s_i(t)^d) = \sum_{i=0}^{C-1} s_i(t)^d - Cs_C(t)^d$. For $d = 1$, it's clear that at the equilibrium point,

$$T_1(\lambda) = \sum_{i=0}^{C-1} s_i - Cs_C = \frac{1 - (C+1)\lambda^C + C\lambda^{C+1}}{(1-\lambda)(1-\lambda^{C+1})}.$$

Then, we consider the convergence of sequence $\{s_i(t)\}_{i=0}^{C}$ for $d \ge 1$.

Definition 1 *A sequence $\{x_i\}$ is said to decrease doubly exponentially if and only if there exist positive constants N, $\alpha < 1, \beta > 1$, and γ such that for $i \ge N$, $x_i \le \gamma\alpha^{\beta^i}$.* □

Then, we show that every trajectory of the system (3.1) converges to a fixed point.

Corollary 2 *Suppose there exists some j such that $s_j(0) = 0$. Then the sequence $\{s_i(t)\}_{i=0}^{C}$ decreases doubly exponentially for all t≥0, where the associated constants are independent of t.*

Proof: According to the definition of sequence $\{s_i(t)\}_{i=0}^{C}$, it's clear that $s_0 \ge s_1 \ge \ldots \ge s_C$, i.e. a monotone decreasing sequence. Let's first increase $s_i(0)$ such that $s_i(0) = s_{i-1}(0) - \epsilon$, where ϵ is a small constant. Let $v = max(s_i(0) \cdot \lambda^{-\frac{d^i-1}{d-1}})^{\frac{1}{d^i}}$. In the original system $s_i(t) \le s_i(0)$ for all $t \ge 0$. Then, $s_i(t) \le \lambda^{\frac{d^i-1}{d-1}} \cdot v^{d^i}$. Based on the result in [15], we conclude that $\{s_i(t)\}_{i=0}^{C}$ decreases doubly exponentially for all t≥0. Next, we further find the upper and lower bounds of the equilibrium points. □

Sequence $\{s_i(t)\}_{i=0}^{C}$ decreases doubly exponentially to a fixed point. Let $\vec{\pi} = [\pi_0, \pi_1, \cdots, \pi_C]$ denote the equilibrium points of states $\{s_i\}$ in system (3.1). We now examine the expected time a query spends in the homogeneous and bounded-capacity P2P system.

Theorem 3 *For $\lambda \in [0,1]$ and $d \ge 2$, $T_d(\lambda) \le \alpha(\log T_1(\lambda))$, where α is a constant dependent only on d and C. Moreover,*

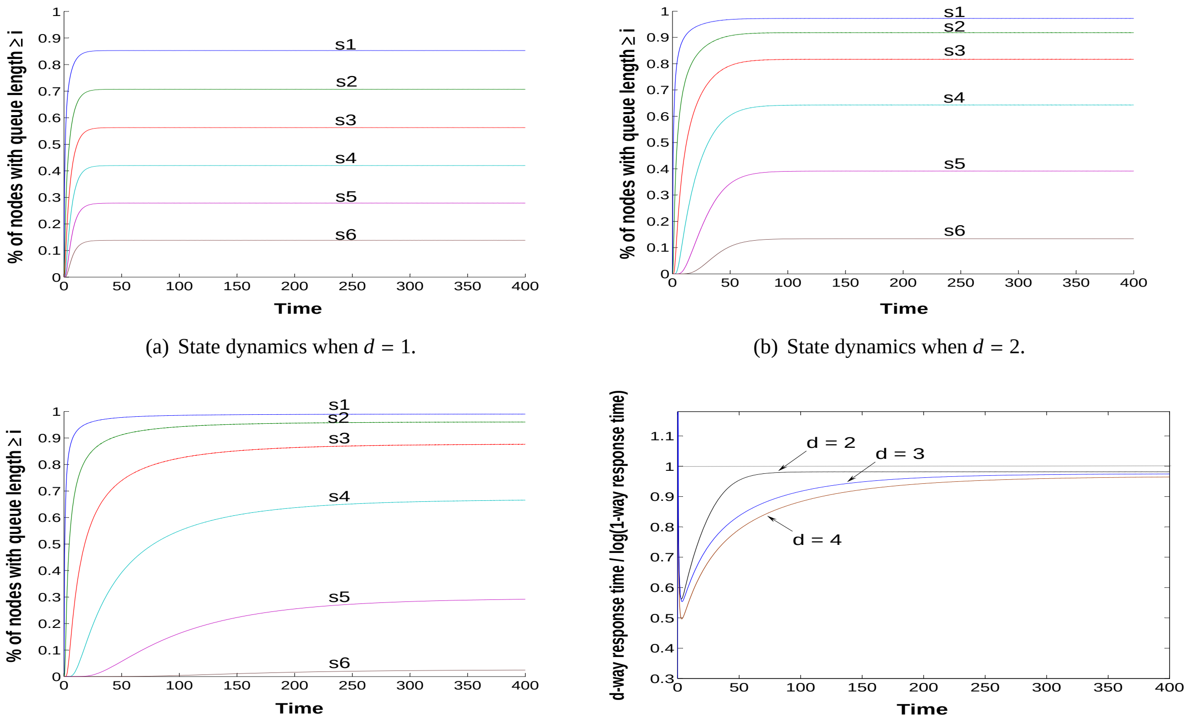
$$\lim_{\lambda\to 1^-} \frac{T_d(\lambda)}{\log T_1(\lambda)} = \frac{C}{\log\frac{C}{2}\,\log d}. \tag{3.2}$$

Proof: First, we prove $\pi_i \le \lambda^{\frac{d^i-1}{d-1}}$ by induction. For $i = 0$, $\pi_0 = \lambda^{\frac{d^i-1}{d-1}}\big|_{i=0} = 1$. For $0 \le i \le k$, assume $\pi_i \le \lambda^{\frac{d^i-1}{d-1}}$. Then for $i = k+1$, let's compare the equilibrium points of (3.1) and those of the following unbounded-capacity system

$$\left\{\begin{array}{l} \dot{s}_i \;=\; \lambda(s_{i-1}^d - s_i^d) - (s_i - s_{i+1}), \quad i \ge 1 \\ s_0 = 1 \end{array}\right. \tag{3.3}$$

Let $(\hat{\pi}_i)$ denote the equilibrium points of (3.3). In [15], it was proved that $\hat{\pi}_i = \lambda^{\frac{d^i-1}{d-1}}$. Then, we only need to prove that $\pi_i \le \hat{\pi}_i$ for $0 \le i \le C$, which are as follows. From (3.3), we have $\hat{\pi}_i = \lambda\hat{\pi}_{i-1}^d$, $i \ge 1$. According to (3.1), $\pi_i = \lambda(\pi_{i-1}^d - \pi_C^d)$, $1 \le i \le C$. Because π_i, $\hat{\pi}_i \ge 0$, $\pi_i \le \lambda\pi_{i-1}^d$. Based on the assumption $\pi_i \le \hat{\pi}_i$ for $i \in [0,k]$, we get $\pi_{k+1} \le \lambda\pi_k^d \le \lambda\hat{\pi}_k^d = \hat{\pi}_{k+1}$. Therefore, the equilibrium points $\pi_i \le \lambda^{\frac{d^i-1}{d-1}}$ for $1 \le i \le C$. On the other end, we prove $\pi_i \ge a\hat{\pi}_i^b$, where $a = \lambda^{\frac{1}{d-1}}$ and $b = 2 - \frac{\log(1+\lambda)}{\log\lambda}$. The induction step is as follows: assume $\pi_i \ge a\hat{\pi}_i^b$ for $0 \le i \le k-1$, and then according to (3.1) $\pi_k + \lambda\pi_C^d = \lambda\pi_{k-1}^d \Rightarrow \lambda\pi_{k-1}^d \le \pi_k + \lambda\pi_k^d \le (1+\lambda)\pi_k$. Thus, $\pi_k \ge \frac{\lambda}{1+\lambda}\pi_{k-1}^d \ge \frac{\lambda}{1+\lambda} \cdot a^d \cdot \hat{\pi}_{k-1}^{bd}$. Because $\hat{\pi}_k = \lambda\hat{\pi}_{k-1}^d$ from (3.3), we have $\pi_k \ge \frac{1}{1+\lambda} \cdot \frac{a^{d-1}}{\lambda^{b-1}} \cdot a\hat{\pi}_k^b = \frac{1}{(1+\lambda)\lambda^{b-2}} a\hat{\pi}_k^b = a\hat{\pi}_k^b$.

Based on the result $a\hat{\pi}_i^b \le \pi_i \le \hat{\pi}_i$ for $0 \le i \le C$, we can calculate the upper and lower bounds of $T_d(\lambda)$. $T_d(\lambda) = \sum_{i=1}^{C} i(\pi_{i-1}(t)^d - \pi_i(t)^d) = \sum_{i=0}^{C-1} \pi_i^d - C\pi_C^d = \sum_{i=1}^{C} \pi_i$. Therefore, $a\sum_{i=1}^{C} \hat{\pi}_i^b \le T_d(\lambda) \le \sum_{i=1}^{C} \hat{\pi}_i$. When $\lambda \to 1^-$, π_i tends to be $\lambda^{\gamma\frac{d^i-1}{d-1}}$, where γ is a constant within $[1, \frac{3}{2}]$. Therefore, $T_d(\lambda) = \sum_{i=1}^{C} \lambda^{\gamma\frac{d^i-1}{d-1}}$. Because



(a) State dynamics when $d = 1$.

(b) State dynamics when $d = 2$.

(c) State dynamics when $d = 3$.

(d) $\alpha T_d(\lambda)/\log T_1(\lambda)$ changes with time.

Figure 1. Dynamics of query response time and nodal queue length in a simulated P2P network with homogeneous and bounded-capacity nodes.

$\prod_{i=0}^{C-1}(1 + \lambda^{d^i} + \lambda^{2d^i} + \ldots + \lambda^{(d-1)d^i}) = \frac{1-\lambda^{d^c}}{1-\lambda}$, we get $\sum_{i=0}^{C-1}\log(1+\lambda^{d^i}+\lambda^{2d^i}+\ldots+\lambda^{(d-1)d^i}) = \log(1-\lambda^{d^c})-\log(1-\lambda)$

Based on the result in [15], we have (3.2). □

Adding a constraint of node capacity makes it difficult to find the closed-form solution to (3.1) with parameters λ and d. To analyze the dynamic behaviors of the P2P system with homogeneous and bounded nodal capacity, we conducted simulations to trace the changes of state variables in example systems. In the example system, we have 4 peer nodes, each of which can queue up to 6 queries at a time, and the arrival rate of queries $\lambda = 0.99$. The initial system is empty and queries comes and leaves the P2P system following the model described at the beginning of this section. Figures 1(a), 1(b) and 1(c) depict the dynamics of the nodal queue length when one, two and three choices are made by random in node search. It's clear that as d increases the number of nodes with the longest queues decreases, because incoming queries are more evenly distributed among nodes for larger d. Figure 1(d) presents the values of $\frac{C}{\log\frac{C}{2}\ \log d} \cdot \frac{T_d(\lambda)}{\log T_1(\lambda)}$ converges closely to 1 as time goes on. The small deviation from 1 is because the arrival rate of queries λ is set to 0.99 as to simulate $\lambda \rightarrow 1^-$. The system is stable and we have different trajectory for different d. From this figure, we can measure the expected time that a query stays in the P2P system, i.e. T_d.

We apply Kurtz's theorem [25] to our randomized probing model in P2P network to obtain bounds on the maximum load:

Theorem 4 *For any fixed time interval I, the length of the longest queue in an initially empty P2P system with homogeneous and bounded nodal capacity for $d \geq 2$ over the interval $[0, I]$ is $\frac{C}{\log\frac{C}{2}\ \log d}\log\log N + O(1)$ with high probability, where C is the nodal capacity and the $O(1)$ term depends on I and λ.*

Hence in comparing the systems where queries have one choice and those have $d \geq 2$ choices, we see that the second yields an exponential improvement in both the expected time in the system and in the maximum observed load for sufficiently large N.

3.1.2 Heterogeneous and Bounded Case

In Section 3.1.1, we extended the supermarket model to analyze the effect of d-way random probing in balancing load in P2P systems with homogeneous and bounded nodal capacity. However, in practical P2P networks, participant nodes generally have different configurations. To tackle this nodal heterogeneity, we extend the preceding model to analyze behaviors of peer nodes with different capacities in face of randomized probing to balance load.

Here, we still consider P2P systems with static composition. Let's assume a system has N peer nodes to process queries. Their correspondent capacities are $\{c_1, c_2, \ldots, c_N\}$, which are positive and finite. We assume c_i's take nonuniform values, otherwise we can analyze the system by applying the model presented in Section 3.1.1. Next, we will try to model and investigate behavior of the P2P system with heterogeneous and bounded nodal capacity by utilizing results derived in the preceding section.

Let c^* denote the maximum values in the sequence $\{c_i\}_{i=1}^{N}$. Then, we calculate the residue capacity as $\{c^* - c_i\}_{i=1}^{N}$. We treat these residue capacity as the initial load of their corresponding nodes. Thus, value of the state variables s_i for $0 \leq i \leq c^*$ at time $t = 0$ equals to $s_i(0) = \mid \{c_i | c_i \leq c^* - i\} \mid /N$, i.e. the fraction of nodes bearing initial load (residue capacity) no less than i. When the system runs and queries come/leave, the area of residue capacity is reserved and load changes in the rest area within c^*. With this transformation, peer nodes have homogeneous capacity as c^* so that we can model the dynamic system by (3.1). The state variables satisfy the following equations:

$$\begin{cases} \dot{s}_i \ = \ \lambda(s_{i-1}^d - s_i^d) - (s_i - s_{i+1}), \quad i < C \\ \dot{s}_C \ = \ \lambda(s_{C-1}^d - s_C^d) - s_C \\ s_i(0) = \frac{|\{c_i | c_i \leq c^* - i\}|}{N}, \quad i \leq C \\ s_i(t) \geq s_i(0) \end{cases} \tag{3.4}$$

For example, in a small-scale P2P network having four nodes, they can accommodate at most 3, 3, 4 and 6 queries at a time, respectively. Thus, $c^* = 6$ and their residue capacities are $\overline{c} = \{3, 3, 2, 0\}$, which determines the initial load of the corresponding nodes $\{s_i(0)\}_{i=0}^{6} = \{1, 0.75, 0.75, 0.5, 0, 0, 0\}$. The system dynamics can be modeled by (3.4) and its solution describes the steady states of the P2P system.

Equations (3.4) are established by exerting constraints on the initial values and the range of state variables to (3.1). Their equilibrium states have certain relations.

Corollary 5 *Let $\tilde{T}_d(\lambda)$ denote the expected time a query spends in the system (3.4). Then, $\tilde{T}_d(\lambda)$ is bounded by:*

$$T_d(\lambda) \leq \tilde{T}_d(\lambda) \leq T_d(\lambda) + S,$$

where $T_d(\lambda)$ is the expected time a query spends in the homogeneous and bounded-capacity system (3.1) and S is a constant which equals to $\sum_{i=1}^{C} s_i(0)$.

Proof: In Section 3.1.1, we prove that (3.1) converges doubly exponentially to its equilibrium state $\{\pi_i\}$. We now derive the solution to (3.4) based on $\{\pi_i\}$. Let $\{m_i\}_{i=1}^{C} = \{max(\pi_i, s_i(0))\}_{i=1}^{C}$. We calculate the remains of $(\lambda m_{i-1}^d - m_i)$ for $1 \leq i \leq C$. Let $r = min\{(\lambda m_{i-1}^d - m_i)_{i=1}^{C}\}$. Then, we obtain the solution $(\tilde{\pi}_i)$ by iteratively computing $\tilde{\pi}_i = \lambda \tilde{\pi}_{i-1}^d - m_i$ for $1 \leq i \leq C$, starting with $\tilde{\pi}_0 = 1$.

If $\{\pi_i\}$ is no less than the value of the initial state $\{s_i(0)\}$ in (3.4), then $\{\pi_i\}$ is also the steady state of (3.4). $\tilde{T}_d(\lambda) = \sum_{i=1}^{C} i(\tilde{\pi}_{i-1}^d - \tilde{\pi}_i^d) = \sum_{i=1}^{C} \tilde{\pi}_i$. Therefore, $\tilde{T}_d(\lambda) \geq T_d(\lambda)$. If $\{\pi_i\}$ is less than $\{s_i(0)\}$, we have $\tilde{\pi}_i \leq \pi_i + s_i(0)$. The expected time a query spends in the system satisfies $\tilde{T}_d(\lambda) \leq T_d(\lambda) + S$, where S is a constant which equals to $\sum_{i=1}^{C} s_i(0)$. □

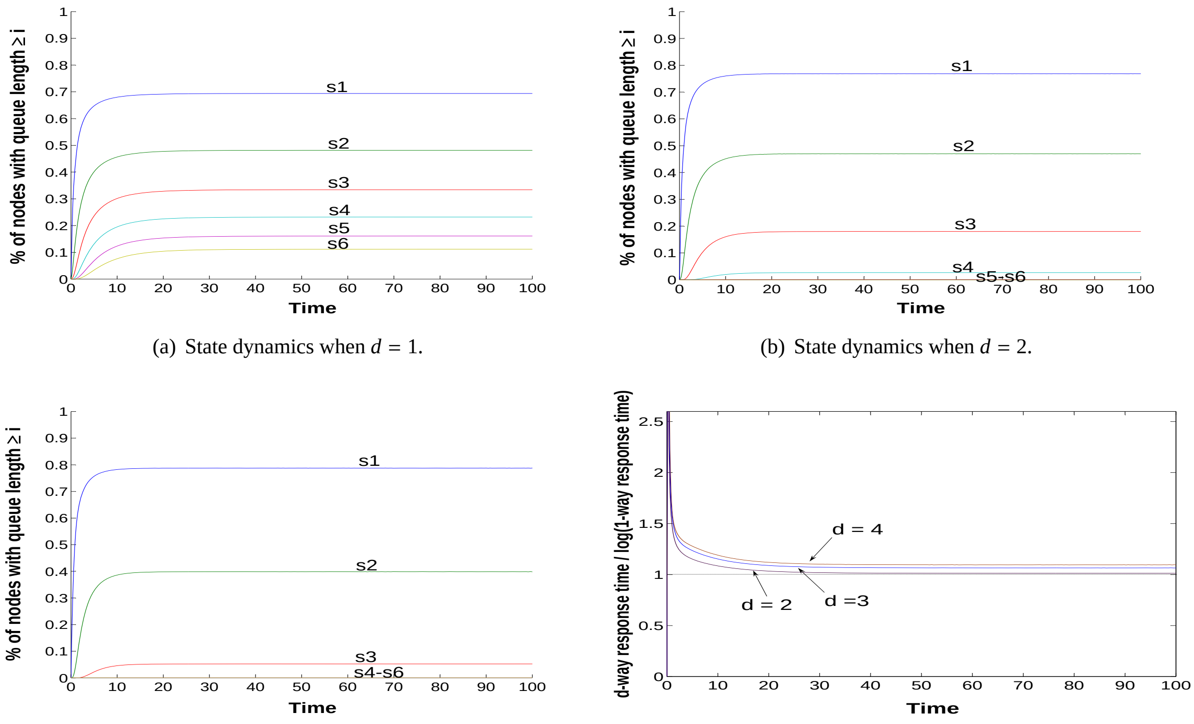
Based on Corollary 5, we apply Kurtz's theorem to derive the upper bound of the length of queues in the P2P system with heterogeneous and bounded nodal capacity.

Theorem 6 *For any fixed time interval I, the length of the longest queue in an initially empty P2P system with heterogeneous and bounded nodal capacity for $d \geq 2$ over the interval $[0,\ I]$ is $\frac{c^*}{\log \frac{c^*}{2}\ \log d} \log \log N + O(1)$ with high probability, where c^* is the upper bound of nodal capacity, and the $O(1)$ term depends on c^*, $s_i(0)$, I and λ.*

The result of Theorem 6 is reduced to the one in Theorem 4 when the system is homogeneous. This theorem indicates that the maximum load of peer nodes is affected by the distributions of nodal capacity. However, the power of 2-way randomized probing is still valid in P2P networks with heterogeneous and bounded nodal capacity.

3.2 Node Dynamics

The composition of a P2P network is dynamic. Compute nodes join and leave the P2P system in its lifetime. Guha et al. [10] observed that only 30% - 40% supernodes were online at any given time in a P2P network. Nodal churn must be considered when we analyze the behaviors of randomized probing to balance load among peer nodes. In this section, we model the dynamic composition of peer nodes in a P2P network by using a random variable. Then we investigate the impact of nodal churn to the expect time a queue spends in the system and the longest length of



(a) State dynamics when $d = 1$.

(b) State dynamics when $d = 2$.

(c) State dynamics when $d = 3$.

(d) $\alpha T_d(\vec{\lambda})/\log T_1(\vec{\lambda})$ changes with time.

Figure 2. Dynamics of query response time and nodal queue length in a simulated P2P network with churn and unbounded-capacity nodes.

queues among peer nodes. To make our analysis tractable, we first discuss randomized probing in a dynamic P2P network where nodes have infinite capacity, as in Section 3.2.1. Then, peer nodes with bounded capacities are investigated in Section 3.2.2.

3.2.1 Dynamic Nodes with Unbounded Capacity

In Section 3.1, we analyze the properties of randomized probing in static P2P systems by focusing on the factor of nodal capacity. In a dynamic P2P system with nodal churn, node composition is not fixed any more. We use a random variable n to characterize the number of peer nodes that changes with time. Then, variable $m_i(t)$, the number of nodes whose loads are at least i, is also random. Their ratio $m_i(t)/n$ denoted by $s_i(t)$ still describes the fraction of peer nodes bearing loads that are at least i.

Existing research work [30, 27, 5, 12, 20] on analyzing churn in P2P systems found that the arrival/departure processes of peer nodes can be modeled by a Poisson distribution when the system size becomes sufficiently large. Thus, we assume new nodes join the current P2P network in a Poisson distribution with rate λ_{in} and existing nodes leave the system in a Poisson distribution with rate λ_{out} relative to the node population, $\lambda_{in}, \lambda_{out} < 1$. When a peer node leaves, its original load will be removed from the P2P system. We assume the probability with which a node leaves the system is uniformly distributed among the existing nodes. As a result, the number of nodes join/leave the P2P system is $\triangle n = (\lambda_{in} - \lambda_{out})n\triangle t$ in a small time interval $\triangle t$. A new node can service coming queries when they are dispatched to it.

To characterize the system dynamics, we look into the change of random variable m_i, the number of nodes that have load at least i. Its value changes when a query is dispatched to a node queuing $i - 1$ queries, or a query is serviced and removed from a node having i queries or such a node leaves the system. Therefore,

$$\frac{\triangle\, m_i}{\triangle\, t} = \lambda n\Big[\Big(\frac{m_{i-1}}{n}\Big)^d - \Big(\frac{m_i}{n}\Big)^d\Big] - n\Big(\frac{m_i}{n} - \frac{m_{i+1}}{n}\Big) - \lambda_{out} n \frac{m_i - m_{i+1}}{n}.$$

Thus, the influence of nodal churn on system behavior is incorporated in the value of random variable m_i. Because $s_i = m_i/n$, we have $\dot{s}_i = (\dot{m}_i n - m_i \dot{n})/n^2$. We use $\{s_i\}_{i=0}^{\infty}$ and n as state variables and the state equations characterizing system dynamics are as follows.

$$\begin{cases} \dot{s}_i \ = \ \lambda(s_{i-1}^d - s_i^d) - (1+\lambda_{in})s_i + (1+\lambda_{out})s_{i+1}, \quad i \geq 1 \\ \dot{n} \ = \ (\lambda_{in} - \lambda_{out})n \\ s_0 \ = \ 1, \end{cases} \tag{3.5}$$

with the initial condition $s_i(0) = 0,\ 1 \leq i \leq n$ and $n(0) = N$, where N is the number of nodes in the initial system.

The population of the P2P system depends on the values of λ_{in} and λ_{out}. If $\lambda_{in} > \lambda_{out}$, n tends to increase. On the other hand, when n increases, more queries will arrive at the system. A similar situation happens when $\lambda_{in} < \lambda_{out}$. Thus, although nodes join/leave the system in runtime, the state variables $\{s_i\}_{i=0}^{\infty}$ can converge to steady state by adjusting the volume of queries.

Corollary 7 *Let $T_d(\lambda)$ denote the expected time a query spends in the system (3.5). Then, $T_d(\vec{\lambda})$ is bounded by:*

$$0 \leq T_d(\vec{\lambda}) \leq \frac{1+\lambda_{in}}{1+\lambda_{out}} \lambda^{d-1} \hat{T}_d(\lambda),$$

where $\vec{\lambda} = \{\lambda, \lambda_{in}, \lambda_{out}\}$ and $\hat{T}_d(\lambda)$ is the expected time a query spends in the homogeneous and infinite-capacity system ([16]).

Proof: Let $\dot{s}_i = 0$ in (3.5) for $i \geq 1$, and we get $s_{i+1} = \frac{1}{1+\lambda_{out}}[(1+\lambda_{in})s_i - \lambda s_{i-1}^d + \lambda s_i^d]$. $T_d(\vec{\lambda}) = \sum_{i=1}^{\infty} i(\pi_{i-1}^d - \pi_i^d) = \sum_{i=0}^{\infty} \pi_i^d$. By using induction, we can prove the expression in the corollary. □

Figures 2(a), 2(b) and 2(c) depict dynamics of the nodal queue length when one, two and three choices are made by random in node search. $\lambda_{in} = 0.2$, $\lambda_{out} = 0.1$ and $\lambda = 0.99$. It's clear that as d increases the number of nodes with the longest queues decreases, because incoming queries are more balanced distributed among nodes for larger d. Figure 2(d) presents the values of $\frac{1+\lambda_{out}}{(1+\lambda_{in})\log d} \cdot \frac{T_d(\lambda)}{\log T_1(\lambda)}$ for as $\lambda \to 1^-$. We can see they converge closely to 1 as time goes on. The small deviation from 1 is because the arrival rate of queries λ is set to 0.99 as to simulate $\lambda \to 1^-$.

We apply Kurtz's theorem [25] to our randomized probing model in P2P network to obtain bounds on the maximum load:

Theorem 8 *An initially empty P2P system in which nodes' arrival/departure follows a Poisson distribution with rate $\lambda_{in}n$ and $\lambda_{out}n$ and nodes have infinite capacity. For any fixed I, the length of the longest queue in the system for $d \geq 2$ over the interval $[0,\ I]$ is $O(\log \log n)$ with high probability.*

3.2.2 Dynamic Nodes with Bounded Capacity

A more general case is a P2P system that peer nodes arrive/depart at runtime and the participant nodes have heterogeneous and bounded capacities. In this section, we construct the state equations to describe the system dynamics and derive the bounds on length of the longest queue.

Let $c_1, c_2, \cdots, c_n$ denote the capacities of peer nodes in the system. The number of nodes n is a random variable. We assume $\{c_i\}_{i=1}^n$ follows a Pareto distribution, with a shape parameter k_c. The number of nodes bearing load at least i, $m_i(t)$, follows the equations:

$$\begin{cases} \dot{m}_i \ = \ \lambda n[(\frac{m_{i-1}}{n})^d - (\frac{m_i}{n})^d] - n[\frac{m_i}{n} - \frac{m_{i+1}}{n}] \\ \qquad\quad +\lambda_{in} n \cdot Pr\{c^* - c \geq i\} - \lambda_{out} n \frac{m_i - m_{i+1}}{n}, \quad i < c^* \\ \dot{m}_{c^*} \ = \ \lambda n[(\frac{m_{c^*-1}}{n})^d - (\frac{m_{c^*}}{n})^d] - n\frac{m_{c^*}}{n} - \lambda_{out} n \frac{m_{c^*}}{n}, \quad i = c^* \\ \dot{n} \ = \ (\lambda_{in} - \lambda_{out})n, \end{cases}$$

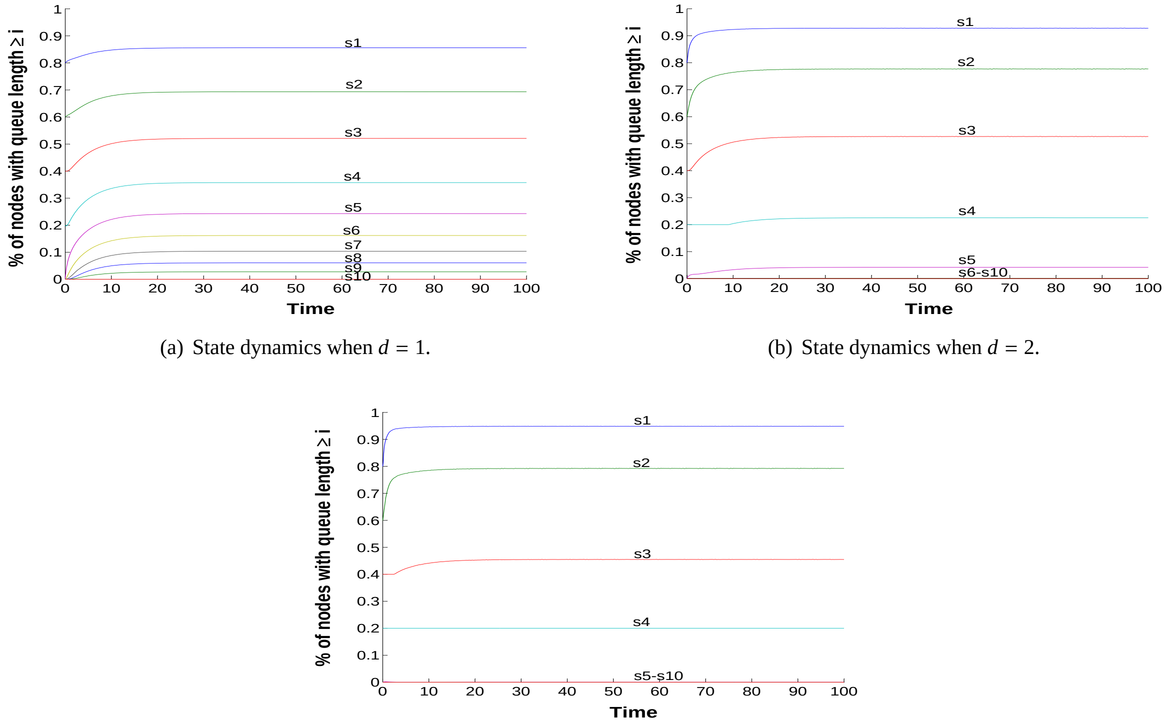
where c^* is a sufficiently large value that is greater than any possible value of node capacity. According to the CDF of Pareto distribution, $Pr\{c^* - c \geq i\} = Pr\{c \leq c^* - i\} = 1 - (\frac{c^*-i}{c_{min}})^{-k_c}$, where c_{min} is a minimum capacity.

By applying $\dot{s}_i = (\dot{m}_i n - m_i \dot{n})/n^2$, we construct the following state equations.

$$\begin{cases} \dot{s}_i \ = \ \lambda(s_{i-1}^d - s_i^d) - (1+\lambda_{in})s_i + (1+\lambda_{out})s_{i+1} \\ \qquad\quad +\lambda_{in}[1 - (\frac{c^*-i}{c_{min}})^{-k_c}], \quad i \leq c^* - c_{min} \\ \dot{s}_i \ = \ \lambda(s_{i-1}^d - s_i^d) - (1+\lambda_{in})s_i + (1+\lambda_{out})s_{i+1}, \\ \qquad\quad c^* - c_{min} < i < c^* \\ \dot{s}_{c^*} \ = \ \lambda(s_{c^*-1}^d - s_{c^*}^d) - (1+\lambda_{in})s_{c^*}, \quad i = c^* \\ \dot{n} \ = \ (\lambda_{in} - \lambda_{out})n \\ s_i(0) = \frac{|\{c_i | c_i \leq c^* - i\}|}{N}, \quad i \leq c^* \\ s_i(t) \geq s_i(0) \end{cases} \tag{3.6}$$

with initial condition $s_0(0) = 1$, $s_i(0)$, $1 \leq i \leq n(0)$ set according to the server configuration at $t = 0$.

It's difficult to calculate closed form solutions to (3.6). However, it is numerically solvable. Figure 3(a), 3(b) and 3(c) show the dynamics of state variables $\{s_i\}$ in an example system. The system consists of 10 nodes at $t = 0$. Their capacity are within the set of $\{6, 7, \ldots, 10\}$, and for each value in the range there are two nodes having that capacity. λ_{in}, λ_{out} and λ are 0.2, 0.1 and 0.99 respectively. The capacity of newly joined nodes follows a Pareto distribution with shape parameter $k_c = 2$ and the minimum capacity $c_{min} = 5$. The figures show the system evolves to steady states as time goes on. We can calculate $T_d(\vec{\lambda})$ based on values of these steady states.



(a) State dynamics when $d = 1$.

(b) State dynamics when $d = 2$.

(c) State dynamics when $d = 3$.

Figure 3. Dynamics of query response time in a simulated P2P network with churn and heterogeneous, bounded-capacity nodes.

4 Experimental Results

To quantify the performance of random choices schemes in real P2P systems, we conducted simulations on Cycloid [24] P2P network. Cycloid is a constant-degree DHT based on the network topology of cube-connected-cycle. We use two transit-stub topologies generated by GT-ITM [31]: “ts5k-large” which has 5 transit domains, 3 transit nodes per transit domain, 5 stub domains attached to each transit node, and 60 nodes in each stub domain on average. It is used to represent a situation in which Cycloid overlay consists of nodes from several big stub domains. To account for the fact that interdomain routes have higher latency, each interdomain hop counts as 3 hops of units of latency while each intradomain hop counts as 1 hop of unit of latency. We assume the object arrival locations are uniformly distributed over the Id space. The number of nodes is 4096 and the number of items is 20480. The capacity of nodes is modeled by a bounded Pareto distribution with its shape parameter as 2. The background load caused by existing items is modeled by a bounded Pareto distribution with its shape equal to 2. Pareto distribution reflects real world where there are machines with capacities that vary by different orders of magnitude.

To balance load in the P2P network, probes are sent to a number of peer nodes. Among the responders, the one having the least load is selected. We refer to this class of randomized load balancing algorithms as d-way probing, denoted by LAR_d, $d = 1, 2,$. We compare the performance of 1, 2, 4, and 6-way random probe schemes in terms of node probing time and total number of load rearrangements. From Figure 4 and 5, we can observe that the probing efficiency of the randomized load balancing algorithm LAR_d,($d > 2$), is almost the same as that for $d = 2$, though they need to probe more nodes than the latter. Our results are closely consistent with the performance results of randomized algorithms analyzed in Section 3.1 and 3.2.

5 Related Work

In most early DHT structures [26, 14, 13], each node chooses at random a point in the hash space, typically, the

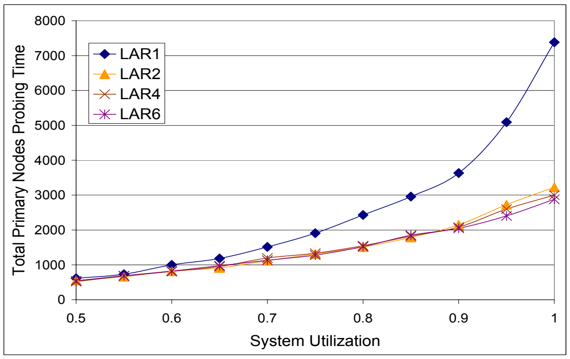


Figure 4. Total node probing time.

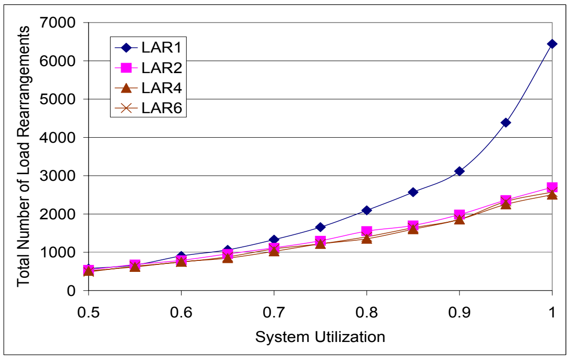


Figure 5. Total number of load rearrangements.

unit interval [0, 1), and becomes associated with the points of the hash space closest to the selected point. Assuming random node departures, this scheme guarantees that the ratio of largest to average node segment size is $O(\log n)$, with high probability [26]. Virtual server approach has been used to mitigate imbalance of key assignment between nodes. It was proposed that each real server works as $\log n$ virtual servers, thus greatly decreasing the probability that some server will get a large part of the ring. Some extensions of this method were proposed in [18] and [28], where more schemes based on virtual servers were introduced and experimentally evaluated. However, these schemes assume that nodes are homogeneous, objects have the same size, and object IDs are uniformly distributed.

CFS [7] accounts for node heterogeneity by allocating to each node some number of virtual servers proportional to the node capacity. In addition, CFS proposes a simple solution to shed the load from an overloaded node by having the overloaded node remove some of its virtual servers. However, this scheme may result in thrashing as removing some virtual servers from an overloaded node may result in another node becoming overloaded.

Byers et al. [3] proposed the use randomized search to achieve better load balance. Each object is hashed to $d \geq 2$ different IDs, and is placed in the least loaded node of the nodes responsible for those IDs. The other nodes are given a redirection pointer to the selected node so that searching is not slowed significantly. For homogeneous nodes and objects and a static system, picking $d = 2$ achieves a load balance within a $\log \log n$ factor of optimal. However, this scheme was not analyzed or simulated for the case of heterogeneous node capacities and node churn, which are defining characteristics of P2P networks. The paradigm of multiple random choices was also used in [22, 11, 8, 23]. Several peer nodes are probed before store a file to or dispatch a user query to the least loaded one.

Besides applying randomized probing algorithms to balance load in P2P networks, researchers also analyzed the characteristics of random choices theoretically. The main techniques used to analyze random choice problems are layered induction, witness trees, and fluid limits via differential equations. The *layered induction* technique pioneered by Azar et al. [1]. The random choice problem was modeled by balls-and-bins. It bounded the maximum load by bounding the number of bins with k or more balls via induction on k. The layered induction approach provides nearly tight results. An alternative technique for handling the problem called the *witness tree* method [6]. The key idea of this approach is to show that if a “bad event” occurs, i.e. if some node is heavily loaded, one can extract the history of the process a suitable tree of events called the witness tree. The probability of the bad event can then be bounded by the probability of occurrence of a witness tree. Generally, witness tree arguments involves the most complexity, and they have proved to be the most challenging in terms of obtaining tight results. The third technique studies algorithms that use random choices paradigm via *fluid limit* models [16, 4]. The system dynamics can be described by a family of differential equations. This approach is simple and flexible. When the system dynamics can be modeled by this method, the differential equations generally yield accurate numerical results. However, these theory work analyzed a system where compute nodes have homogeneous and infinite capacities. Moreover, node churn, a defining characteristic of P2P systems, is not modeled by these approaches. In this paper, we analyze the dynamic behavior of random choice paradigm in general P2P systems, where peer nodes join/leave at runtime and they have heterogeneous and bounded capacities.

6 Conclusions

In this paper, we model the randomized probing in general peer-to-peer systems. Theoretical analysis shows that two-way random probing is asymptotically superior to the one-way choice approach. However, by increasing the number of choices further, efficiency of the search algorithm does not improve significantly. Our random probing model is general in that the influence by nodal heterogeneity, capacity distribution and churn on search efficiency is investigated. The random approach is less sensitive to the node churn and heterogeneity in terms of the number of probes conducted before finding suitable nodes and the average response time of queries. Simulation and experiment results confirm our analysis. It's difficult to calculate the closed form solutions to state equations of the most general case. However, we can find the steady states numerically. For completeness in theory, we include the analysis of P2P systems consisting heterogeneous and bounded-capacity nodes with churn and simulation results. In this paper, we design and analyze the random probing algorithms within the context of P2P networks. However, our results have wide applicability and are of interest beyond the specific applications.

Although we analyze the performance of randomized probing in the general P2P environments, we still introduce some simplifying assumptions to make the problem tractable. One such assumption is that each peer node can service every query. However, in practice certain queries or requests may only be serviced by those nodes that have the required resources. To address this situation, we need to extend the composition model of a P2P network in a way that each query maps to a subset of nodes that can service it, and the changes of peer nodes' queue length will be quantified by distinguishing these different sets. However, this will make the analysis quite difficult. Another assumption is that the service time for a query is exponentially distributed with mean 1. In reality, different queries may require different amount of work and the processing power of peer nodes may vary. As a result, the service time follows a more complicated model. Although we do not discuss this case in our paper, our model can be extended to formulate the case by introducing other distributions of service time to the state equations of the system and numerically analyze the state dynamics.

Acknowledgments We would like to thank the anonymous reviewers for their constructive comments and suggestions. This research was supported in part by U.S. NSF grants ACI-0203592, CCF-0611750, DMS-0624849, CNS-0702488, CRI-0708232, and NASA grant 03-OBPR-01-0049.

References

[1] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 2000. A preliminary version of this paper appeared in Proc. of the 26th ACM Symp. on Theory of computing (STOC), 1994.

[2] M. Bienkowski, M. Korzeniowski, and F. M. auf der Heide. Dynamic load balancing in distributed hash tables. In *Proc. of Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, February 2005.

[3] J. Byers, J. Considine, and M. Mitzenmacher. Simple load balancing for distributed hash tables. In *Proc. of the 2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[4] J. W. Byers, J. Considine, and M. Mitzenmacher. Geometric generalizations of the power of two choices. In *Proc. of the 16th ACM Symp. on Parallelism in algorithms and architectures (SPAA)*, 2004.

[5] M. Castro, M. Costa, and A. Rowstron. In *Proc. of the 2nd Symp. on Networked System Design and Implementation (NSDI)*, 2005.

[6] R. Cole, B. M. Maggs, F. M. auf der Heide, M. Mitzenmacher, A. W. Richa, K. Schrder, R. K. Sitaraman, and B. Vocking. Randomized protocols for low-congestion circuit routing in multistage interconnection networks. In *Proc. of the 13th ACM Symp. on Theory of computing (STOC)*, 1998.

[7] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with cfs. In *Proc. of the 18th ACM symp. on Operating systems principles (SOSP)*, 2001.

[8] G. Giakkoupis and V. Hadzilacos. A scheme for load balancing in heterogenous distributed hash tables. In *Proc. of the 24th ACM Symp. on Principles of distributed computing (PODC)*, 2005.

[9] B. Godfrey and I. Stoica. Heterogeneity and load balance in distributed hash tables. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, 2005.

[10] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer voip system. In *Proc. of 5th Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2006.

[11] K. Kenthapadi and G. S. Manku. Decentralized algorithms using both local and random probes for P2P load balancing. In *Proc. of the 7th ACM Symp. on Parallelism in algorithms and architectures (SPAA)*, 2005.

[12] S. Krishnamurthy, S. El-Ansarh, E. Aurell, and S. Haridi. A statistical theory of chord under churn. In *Proc. of 4th Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2005.

[13] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: a scalable and dynamic emulation of the butterfly. In *Proc. of the 21st ACM Symp. on Principles of Distributed Computing (PODC)*, 2002.

[14] G. Manku, M. Bawa, and P. Raghavan. Symphony: Distributed hashing in a small world. In *Proc. of the 4th USENIX Symp. on Internet Technologies and Systems (USITS)*, 2003.

[15] M. Mitzenmacher. On the analysis of randomized load balancing schemes. In *Proc. of ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, 1997.

[16] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Trans. on Parallel Distributed Systems*, 12(10):1094–1104, 2001. Ph.D. thesis, U. C. Berkeley, 1996.

[17] M. Mitzenmacher, B. Prabhakar, and D. Shah. Load balancing with memory. In *Proc. of the 43rd IEEE Symp. on Foundations of Computer Science (FOCS)*, 2002.

[18] A. Rao, K. Lakshminarayanan, and S. Surana. Load balancing in structured P2P systems. In *Proc. of the 2nd Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proc. of Conf. on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, 2001.

[20] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *Proc. of USENIX Annual Technical Conference*, 2004.

[21] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, 2001.

[22] H. Shen and C.-Z. Xu. Elastic routing table with provable performance for congestion control in DHT networks. In *Proc. of the 26th IEEE Intl. Conf. on Distributed Computing Systems (ICDCS)*, 2006.

[23] H. Shen and C.-Z. Xu. Locality-aware and churn-resilient load balancing algorithms in structured Peer-to-Peer networks. *IEEE Trans. on Parallel and Distributed Systems*, 18(6):849–862, 2007.

[24] H. Shen, C.-Z. Xu, and G. Chen. Cycloid: a constant-degree and lookup-efficient P2P overlay network. *Performance Evaluation*, 63(3):195–216, 2006.

[25] A. Shwartz and A. Weiss. *Large Deviations for Performance Analysis: Queues, Communication and Computing*. Chapman and Hall, London, UK, 1995.

[26] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. *IEEE/ACM Trans. on Networking*, 11(1):17–32, 2003.

[27] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proc. of ACM Internet Measurement Conference (IMC)*, 2006.

[28] S. Surana, B. Godfrey, K. Lakshminarayanan, R. Karp, and I. Stoica. Load balancing in dynamic structured peer-to-peer systems. *Performance Evaluation*, 63(3):217–240, 2006.

[29] S. Tewari and L. Kleinrock. Proportional replication in peer-to-peer networks. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, 2006.

[30] Z. Yao, D. Leonard, X. Wang, and D. Loguinov. Modeling heterogeneous user churn and local resilience of unstructured P2P networks. In *Proc. of 14th Intl. Conf. on Network Protocols (ICNP)*, 2006.

[31] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *Proc. of IEEE Conf. on Computer Communications (INFOCOM)*, 1996.

[32] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. Kubiatowicz. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. *Journal on Selected Areas in Communications*, 12(1):41–53, 2004.

[33] C. Zheng, G. Shen, S. Li, and S. Shenker. Distributed segment tree: Support of range query and cover query over DHT. In *Proc. of the 5th Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, 2006.

[34] Y. Zhu and Y. Hu. Efficient, proximity-aware load balancing for DHT-based P2P systems. *IEEE Trans. on Parallel and Distributed Systems*, 16(4):349–361, 2005.