Konstantin Pussep, Christof Leng, Sebastian Kaune:
Modeling and Tools for Network Simulation: Modeling User Behavior in P2P Systems. In: Klaus Wehrle, Mesut Günes, James Groß, p. 447--461, Springer, July 2010. ISBN 978-3-642-12330-6.

20. Modeling User Behavior in P2P Systems

Konstantin Pussep (Technische Universität Darmstadt)
Christof Leng (Technische Universität Darmstadt)
Sebastian Kaune (Technische Universität Darmstadt)

20.1 Introduction

The evaluation of *peer-to-peer (P2P)* systems is crucial for understanding their performance and therefore their feasibility in the real world. Different techniques, such as testbeds, analytical analysis, and simulations, can be used to evaluate system performance. For peer-to-peer systems, simulations are often the most reasonable approach, simply because P2P systems are both inherently large-scale and complex.

While simulations are a popular technique for estimating the performance of large scale P2P systems with an acceptable level of detail, the modeling of the system is crucial to obtain realistic results. Aside from a proper model of the underlying network, a proper evaluation has to take user behavior into account, since it is a critical factor of the system dynamics. The modeling of user behavior is already important in traditional client/server environments, because it heavily influences the workload. In addition to their role as consumers, users become providers in P2P systems which aggravates their importance to system performance.

In general, a good model of *user behavior* for P2P systems has to capture both the consumer and provider nature of the participating peers. For example, churn, which describes peers connecting to and disconnecting from the system, has a severe impact on data availability and consistency, while resource distribution and popularity have a significant impact on load balancing. Therefore, a user behavior model must be abstract enough to allow efficient implementations but at the same time capture the most relevant components.

In this chapter we describe a general user behavior model for the simulation of P2P systems, and show how it can be applied to Distributed Hash Tables (DHTs), using Kademlia [310] as an example. We discuss a general model and the alternative approaches for individual components and present their impact on evaluation results.

20.2 System Model

The behavior of P2P users is rather complex. We aim to break it down into understandable and independent components. The main categories shown in Figure 20.1 are *churn*, workload, and user properties.

Churn describes the participation dynamics of P2P nodes. Users join the network, leave, and come back several times. Sometimes they even leave forever because they do not use the system anymore. For example, in systems such as BitTorrent [103] the user is interested in exactly one download per torrent and will normally not join a distribution overlay in which he already completed downloading.

Churn consists of the *lifetime* of a node which starts with its initial creation and ends with its permanent departure. During its lifetime, a node goes through several online and offline cycles (see Figure 20.2). The time span a node is online is called a *session*. The time between sessions is accordingly called *intersession*.

Joining and *leaving* are system-specific operations. In most overlays joining includes finding neighbors, initializing routing tables, replicating data, and other setup operations. Leaving cleans up the session state. A regular leave normally includes the notification of neighbors to help them reorganizing their state. Not all nodes leave the network orderly, some simply *crash*, i.e., they disappear from the overlay without any notification. This can be caused by software crashes, hardware crashes, or loss of connectivity.

A node in a P2P system is both a server and a client. As such it provides *resources* and executes *requests* for available resources. The provision and consumption of resources specifies the *workload* of the system. A realistic workload model is crucial for sensible performance evaluation. For example, if resources are shared files in filesharing systems, a request would stand for downloading a file. As previous studies [186] have shown, the workload is not uniform over all resources. Some resources are abundant, many are scarce. The same is true for requests, some resources are far more *popular* than the others.

Both churn and workload operate directly on the simulated system which itself relies on an underlay model of the Internet in the simulator framework (see Figure 20.1). The *use properties* (especially his strategy) interact with churn and workload. The properties also include the goals and interests of a user. Similar to the global popularity of resources in the workload, a user has no uniform interest in all resources, but consumes only certain types or even very specific resources. This interest clustering is especially important for reputation systems, but has also been used to build specialized content-clustered overlays [107]. Also part of the strategy is the user's willingness to cooperate. Some users try to maximize their benefit, e.g., by not uploading at all or tricking a reputation system. On the other hand, many users do not leave the network immediately after a completed download [279]. Thus, the user strategy does influence both workload and churn.

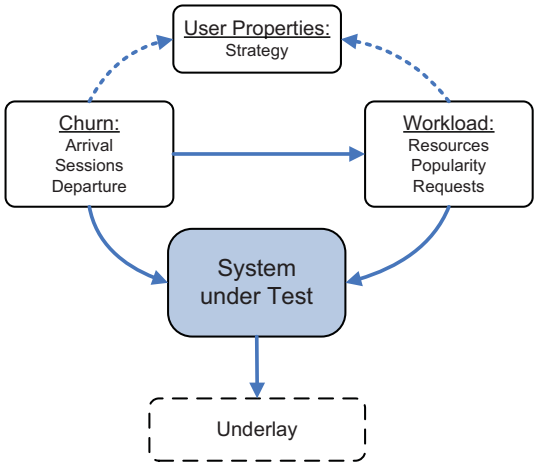


Fig. 20.1: P2P system under test. The *system* is controlled by the *workload* and *churn* components. Different users are modeled by *user properties*. Finally, the *underlay* decides about the transmission delays.

The *underlay*, the network infrastructure the overlay operates on, is important for a reasonable performance evaluation as well. Bandwidth, latency, message loss and other properties of the underlay links have to reflect the conditions of the target real-world environment. However, the modeling of underlays is out of scope for this chapter as it is mostly orthogonal to the user behavior (See Chapter for underlay modeling).

In the next three sections we discuss the components in more detail.

20.3 Modeling Churn

One of the most well-studied and analyzed components of user behavior in P2P systems is *churn*, i.e., the property of peers in the system to leave it and come again at will. Studies such as [448] by Stutzbach et al. measured and analyzed the churn behavior in different P2P systems, focusing on session and intersession times. They found churn characteristics being significantly different across systems.

A churn model can capture the change of online and offline events by considering the total lifetime of a peer and online sessions. Typically, a user participates in the system over a certain time, the *user lifetime* in the system. During this time a user may connect and disconnect from the system many times, therefore creating *online sessions*. Figure 20.2 shows an example lifetime of a peer with three sessions.

We divide the modeling of churn into *lifetime* and *session* models. The first model describes when a peer appears in the system for the first and the

last time. The latter covers peer's online sessions during its lifetime in the system.

In a simulation run, there is a number of N peers instantiated and available for simulations. At any point of time a subset of these peers is online resulting in the actual network size. This network size depends on four factors: arrival rate, join rate, leave rate, and departure rate.

Because scalability is an important property of P2P systems, simulations have to show how the system performs for a given network size. Here the number of *online* hosts is relevant. We will see how the network size can be predicted and how long it might take to reach a steady network state.

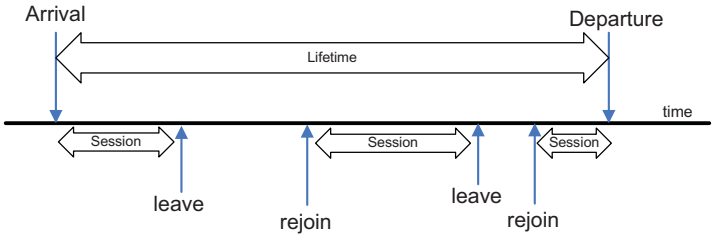


Fig. 20.2: Peer's lifetime can consist of several online sessions with different durations.

20.3.1 Lifetime Models

The lifetime of a peer is determined by two events: *arrival* and *departure*. The peers' arrivals in a system can be influenced by many factors, such as the application deployment process or the dynamics of social networks where people start using the system recommended by their friends. These factors are difficult to capture exactly and therefore a stochastic model appears reasonable to describe this behavior.

Similar arguments apply to peer departures, because users might stop using an obsolete system and switch to a new one. However, this process is typically of less interest for a simulation scenario.

Moreover, in real systems the lifetime of a peer in the system is much longer than the session time. The lifetime can be several days or even months long while the sessions are few hours long on average [59]. As the simulated time often ranges in the order of hours only, the rate of departures is so low that they can be ignored.

However, this is different for systems such as BitTorrent where peers are only interested in downloading one single file and depart from the system

permanently afterwards. This applies because for each file a distinct short-living overlay is created.

In the following we consider two possibilities of modeling the *arrival process*:

Deterministic Arrival Process

Here all peers arrive with a constant inter-arrival interval τ. The complete network of N peers is built up after the fixed time $N \cdot \tau$.

A possible downside of this approach is the lack of randomness, often observed in real system. Furthermore, the fixed inter-arrival intervals can cause a synchronization of periodic events. For example, consider a network with 1000 peers that arrive with the interval of 1 minute. Upon arrival each peer starts some periodic maintenance operation each 10 minutes after the bootstrap process. In this case each minute 100 concurrent maintenance operations can occur.

Poisson Arrival Process

If peers are assumed to arrive independently from each other a Poisson arrival process can be applied [197]. Given a peer arrival rate λ, peer arrivals are distributed exponentially with the same rate λ resulting in the mean inter-arrival interval $\tau = \frac{1}{\lambda}$. For each arrival event an absent peer is selected and connected to the system. Therefore, for N peers the network will be completely built up after (roughly) the time $\frac{N}{\lambda}$.

This is a more realistic model because arrival events happen with different intervals.

20.3.2 Session Models

An online session model describes peers going online and offline after they joined the system for the first time. This model contains two events: *join* and *leave*. The reason why it can be important to distinguish *join* and *arrival* events in a system is that peers coming back to the system can reuse contact information of other peers and the data items stored locally.

The modeling of online sessions can be done either *globally* or *per peer*. In the first case, there is a global rate of join and leave events. Here, when a global join event occurs an offline peer is selected to join the network. The same can apply to leave events. In a simulation environment this method requires only one periodic join and leave event to be scheduled.

The second method schedules one session join/leave event per peer. Once a peer has arrived to the system, its session length is computed and the respective *leave event* is scheduled. After this time is reached and the event

is fired, the intersession length is computed and the *join event* is scheduled. While this method has a larger memory footprint due to the number of scheduled events, it offers high flexibility. The session and intersession durations can follow any desired distribution: exponential, Weibull etc. Furthermore, the peers can be divided into groups, where some peers are long-lasting while others connect and disconnect frequently.

In the following we consider examples for each of both methods.

Exponential Session and Inter-Session Durations

In this model both the session and intersession time of peers follow an exponential distribution with the (possibly different) rates λ_s and λ_i respectively.

The mean session duration of a single peer is $mean_s = \frac{1}{\lambda_s}$ and the mean intersession duration is $mean_i = \frac{1}{\lambda_i}$. Then the expected probability of a peer being online at a given time is the *connectivity factor* $c = \frac{mean_s}{mean_s + mean_i}$. Therefore, for N available peers only approximately $c \cdot N$ peers will be online for any given time.

This rises an interesting trade-off: If we want to simulate with a fixed number of *online* peers and have realistic session and intersession times, then we might end up with a much larger number of peers that must be *available* for the algorithm. This can be a problem if memory is the limiting resource in the simulation environment.

Global Leave Rate with Peer Replacement

In order to reduce the global number of join and leave events a *global leave rate* can be applied, as used e.g., in [388]. Here we can use only one periodic event that dictates when a peer will leave the network. Each time the event is triggered one randomly selected online peer leaves the system.

For the join process no additional events are used. Instead an offline peer is randomly selected and joins the network to replace the peer going offline. Therefore, the network size is constant.

In more detail, when the churn process starts, there is a Poisson process deciding when the next *leave* event should occur. If the leave process is timed by an exponential distribution with the rate of λ_{leave}, then the expected inter-event interval is $1/\lambda_{leave}$ and the corresponding session time is expected to be $t_{mean} = \frac{N_0}{\lambda_{leave}}$ where N_0 is the desired network size.

A benefit of this model is that the memory consumption can be easily limited, e.g we can keep 10,000 peers online from the set of 11,000 available peers, while still keeping the desired mean session duration. For example, if the desired session duration is 50 minutes than the global leave rate is set to $10,000/50 = 200$ events per minute.

20.4 Workload Model

This model comprises two parts: the resource distribution that describes the resources offered and used in the system and the user requests specifying how these resources are requested and consumed.

Since in a P2P application the users are accessing resources residing on other peers, the question arises how the information is distributed among the users. The content items are spread in the system according to some distribution and again depending on the application type and the user structure.

The resource distribution defines how many resources are present in the system. The replication distribution specifies how many replicas of each file are present in the system. Finally, the popularity specifies how often single resources are requested by users.

Several works consider the specifics of P2P file sharing systems regarding the user request patterns, e.g., [186] and [400]. One interesting result is that systems where users consume multimedia content (so most file sharing systems, too) show a download-once behavior, e.g., the user almost never request the same content twice. This leads to a different popularity distribution compared to classical client-server systems, such as web servers.

The actual requests to the system are issued depending on offered resources, resource popularity and user's interests. In a simple model a user issues requests on a regular basis, e.g., each m time units. In a more realistic approach the users issue queries following a distribution, e.g., an exponential distribution.

For systems dealing with content (file sharing, video streaming) the skewed popularity of content can be expressed by heavy-tailed distributions of content replicas and queries. In real P2P systems Zipf or Zipf-Mandelbrot were found to fit the content and query distributions well [186].

20.5 User Properties Model

This section briefly captures user specific properties, e.g., interests, strategy. The interests define which resources the user is interested in, this can be captured by a resource category the user is interested in and results in different request rates and targets per peer. Therefore, for user u it defines the interesting content as a subset of all available resources.

The strategy typically describes the cooperativeness of the peer, as some users are willing to contribute more resources than others. For this reason, *altruistic users* will stay online longer and offer more resources, e.g., files, to the network, while other *strategic users* will try to minimize their contribution. A model can further include *malicious users* who try to break the systems specification. There are many works dealing with the strategies of the users, such as [251].

User properties are very specific to the used application, which makes it difficult to discuss them in a general way. Nonetheless, in most cases user properties simply influence other aspects of the user's behavior, e.g., online time, resources, and requests. Thus, it is generally a good idea to model user properties on top of the churn and workload models.

20.6 Use Case: Kademlia

In this subsection we analyze how to model realistic user behavior for distributed hash tables in order to evaluate the performance of the Kademlia overlay routing protocol. Kademlia is probably the most popular DHT routing overlay, used in large-scale file sharing applications such as the different BitTorrent clients or the eMule client. It differs from other DHTs in two aspects: its large routing table that makes it more resilient to churn and the usage of the symmetric XOR metric, that creates a local tree view of the overlay for each peer.

As a generic DHT Kademlia has many use cases. We consider a file-sharing application in which the DHT is used to locate files offered by peers using a file ID. That results in rare lookup and store requests because users only use the DHT while searching for file sources but not within the actual file transfer. For example, in eMule KAD [447], a variant of Kademlia protocol is used.

In the BitTorrent [103] derivative Azureus[1], Kademlia is used as a decentralized alternative to a centralized tracker. Here the peers can obtain the addresses of other peers downloading the same file and publish their own addresses.

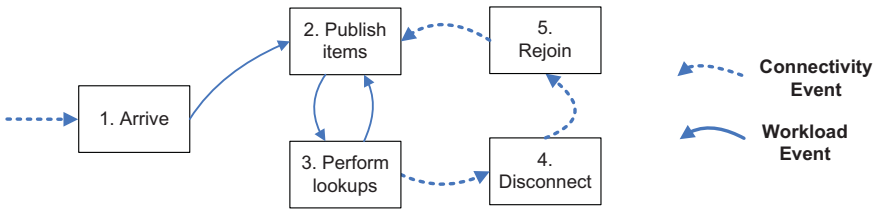


Fig. 20.3: User activities in a DHT. The events that cause the change the current activities are differentiated by the responsible component.

The general activity diagram of a Kademlia user is shown in Figure 20.3. At peer's arrival the Kademlia join procedure takes place. In the two following

[1] www.azureus.com

Component	Model
resources model	Zipf distributed popularity
request model	Zipf distributed popularity
lifetime model	constant arrival intervals
	exponential arrival intervals
session model	exponentially distributed sessions and intersessions
	Weibull distributed sessions/intersessions based
	Global (Poisson) leave rate with peer replacement
	no churn (for comparison)
departure rate	no permanent departure
user type model	simple user model
lookup rate (per peer)	1 lookup each 10 minutes

Table 20.1: Kademlia modeling parameters

phases, lookups and store operations are executed. Here especially the lookup rate and distribution are timed by the applied workload.

The arrival, leave and join events are timed by the churn component. If a peer goes offline then it cancels all running queries. Upon a peer's return to the system the workload component takes over the control of the peer and issues new publish and lookup operations. It further specifies the objects (representing any kind of resources to be published and looked up) available in the system together with their popularity and replication.

The following most useful characteristics for modeling user behavior in DHTs:

- Peer arrival rate: this one is important from two points of view: the initial creation of the network and the avoidance of event synchronization across peers. The latter might happen because of periodic maintenance operations running in the background (see Section 20.3.1) .
- Join and leave events: Performance of a DHT depends on the quality of the routing tables. Stale contacts in the table might cause expensive lookup timeouts and even the lookup success rate can degrade.
- Lookup/store activity: Bulk requests can overload peers responsible for popular content. Furthermore, Kademlia tries to minimize the maintenance overhead by refreshing routing tables during regular lookup operations. If there are no user requests over certain time intervals, active probing is done that increases maintenance overhead.
- Resources distribution, replication and popularity: Different popularity of content will distribute the load in the system unevenly. Also, resources with a low replication factor might become unavailable under churn and result in failed lookups.

20.7 Evaluation

In this section we analyze the impact of user behavior on the performance of the Kademlia protocol. Table 20.1 lists the different components of user behavior under consideration. We focus on the impact of churn being the main issue for the performance of search overlays. In detail, we compare the impact of varying churn setups: different churn models as introduced in Section 20.3 and the impact of single parameters: network size, session and intersession durations.

20.7.1 Methodology

Our simulation platform for the experiments is the discrete event-based simulator PeerfactSim.KOM[2]. The simulator offers a generic framework for different overlays and network models. We use the implementation of the Kademlia routing protocol with the basic setup as shown in Table 20.2.

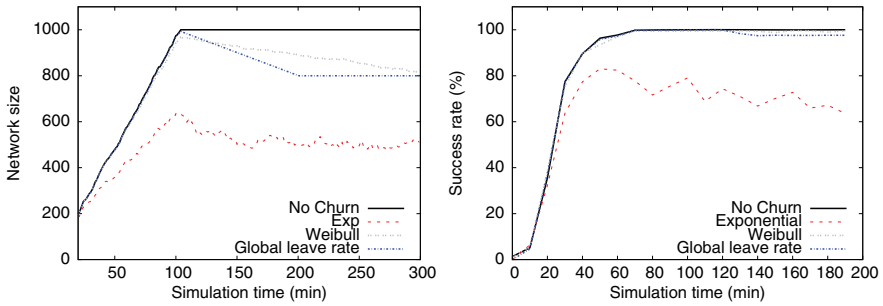
Parameter	Value	Description
ID-Length	80	length of Kademlia id space
b	2	order of the routing tree
k	10	number of contacts per bucket
refresh-interval	1 hour	routing table refresh interval
α	3	number of concurrent messages
republish-interval	1 hour	how often the items are republished

Table 20.2: Kademlia setup

In order to estimate the impact of user behavior on overlay routing the following metrics are used:

- **Routing table quality** measured as the ratio of fresh contacts, i.e., contacts that are online at the given time. This metric is measured globally for all peers in periodic intervals of 5 minutes.
- **Success rate** determines the ratio of lookups being able to find the desired objects. The success rate is aggregated over intervals of 5 minutes. A lookup can fail either if the object is not available at online peers or the peers holding the object are not found by the routing protocol due to the inconsistency of routing tables.
- **Dropped messages per peer** reflects the impact of stale (offline) contacts in the routing tables. Overlay messages in Kademlia are sent via UDP and, hence, get dropped if the destination peer is offline.

[2] www.peerfactsim.org



(a) Network size (b) Success rates

Fig. 20.4: Impact of churn model on performance.

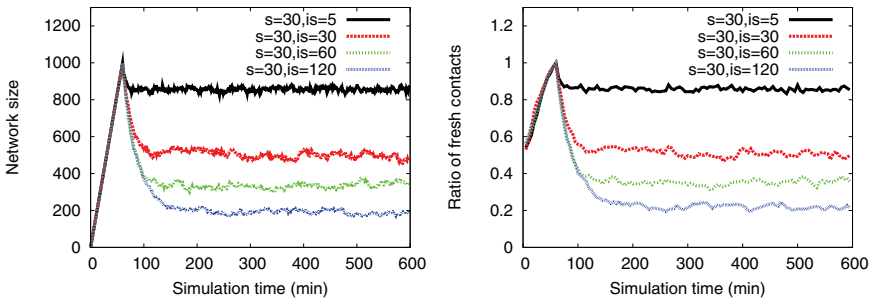
20.7.2 Setup 1: Fixed Network Size, Variable Session Models

At first we evaluate the impact of the session model used on the system performance. The following models are considered:

- Exponentially distributed session and intersession durations (see Section 20.3.2). Both the session and intersession rates are set to 0.05 events per minute (accordingly the means are 20 minutes each).
- Weibull distributed session and intersession durations (following the measurements of Steiner et al. in [441]). The Weibull distribution parameters scale and shape are set to 169.5385 and 0.61511 for session durations. The parameters for the intersession durations are set to 413.6765 and 0.47648 respectively.
- Global leave rate with peer replacement (see Section 20.3.2). Here the rate of leave events is set to 10 events per minute and the target network size to 800 peers.
- No churn for comparison. Here all peers are online once they arrived in the system.

The network sizes of each model during the simulation run are shown in Figure 20.4(a). As we can see here, depending on the distribution size the actual number of online peers is different. Most of them are similar, ranging between 800 and 1000 peers once the network is built up. Only the exponential churn model has different network sizes. For the exponential model with short session durations (both session and intersession means are set to 20 minutes) the network size oscillates around the expected mean of $\frac{N \cdot mean_s}{mean_s + mean_i} = \frac{1000 \cdot 20}{20+20} = 500$ peers.

For the model with a global leave rate the network size reaches the target size of 800 peers after 200 minutes. Here 100 minutes are required for all 1000 peers to arrive at the system and 100 more minutes to reach the desired



(a) Network size (b) Routing table quality

Fig. 20.5: Impact of varying intersession durations (**is** $= \{5, 30, 60, 120\}$ minutes) with the fixed session duration (**s** $= 30$ minutes).

network size of 800 peers. Later the network size stays unchanged, because each leaving peer is immediately replaced by an offline peer.

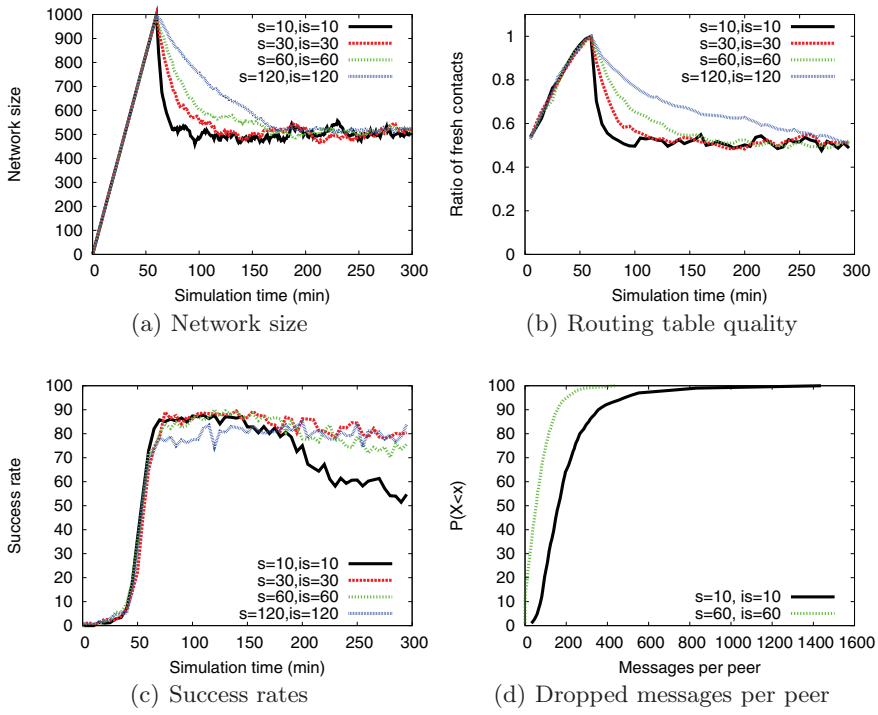
Figure 20.4(b) presents the impact of the churn model used on the success rate. For most of the models no significant difference is visible except the exponential session model, where the success rate degrades dramatically. The explanation is that because of the high churn rate, some objects become unavailable and therefore the lookups for these objects fail. We conclude that the impact of the network size is more relevant than the actual model being used.

20.7.3 Setup 2: Fixed Session Duration, Variable Intersession Duration

In order to evaluate the impact of intersession duration times on the overlay performance we fix the model to the exponential session model and vary the intersession durations. Four different values are used: 5, 30, 60, and 120 minutes. Figure 20.5(a) shows the network sizes for each of them. We can see that after the arrival process is finished the system reaches the expected size and the curve oscillates slightly. The quality of routing tables (shown in Figure 20.5(b)) reflects the network size. For example, the network size of 500 peers out of 1000 available (curve `s=30, is=30`) results in the fresh contact ratio of roughly 50%.

20.7.4 Setup 3: Fixed Network Size, Variable Event Rate

Furthermore, we analyzed the impact of varying online and offline event rates for the constant connectivity factor. Because the equation for connectivity $c = \frac{mean_s}{mean_s + mean_{is}}$ applies we obtain $mean_s = \frac{mean_{is}}{1-c}$. We fix the



(a) Network size (b) Routing table quality

(c) Success rates (d) Dropped messages per peer

Fig. 20.6: Impact of varying intersession (**is**) and session duration (**s**) with the fixed network size of 500 peers.

connectivity factor to 0.5 and therefore obtain equal durations for the session and intersession intervals.

The experiments conducted have expected interval durations of 10, 30, 60, and 120 minutes for both sessions and intersessions. Hence, the expected network size is $0.5 \cdot 1000 = 500$ peers.

As shown in Figure 20.6(a) the network sizes are very similar. Similarly, the quality of routing tables is around 50%. (see Figure 20.6(b)). However, the success rate is very differently as shown in Figure 20.6(c). It drops significantly if the intersession durations are cut to 10 minutes. Here the global leave rate is $500 \cdot \frac{1}{5} = 100$ events per minute and reduces the success rate from 80 to 60%. We further observe that session times of 10 minutes result in an unstable success rate.

Additionally, 20.6(d) presents the distribution of dropped messages in the network per peer for the two extreme setups. We observe that this metric also suffer under high churn rates.

The direct implication of these results is that the replication rate has to be adjusted to the churn rate in order to assure successful lookups. In a real

system where, the churn cannot be easily known in advance the replication mechanism has to be adaptive. For example, in Kademlia an adaptive replication factor k can result in a high recall rate without too much overhead.

20.7.5 Setup 4: Variable Arrival Processes

We further analyzed the impact of different peer arrival rates on Kademlia but found them having a negligible effect on the system performance. The alternatives compared were the deterministic and the stochastic processes as described in Section 20.3.1. Due to the lack of space the graphs are not shown.

Even if during the arrival process a part of objects is unavailable, once their holders arrive, the routing tables get repaired quickly. Hence, in the steady state we measured almost the same success rates despite the arrival process applied. Especially, we found no difference between a Poisson process and peer arrivals with the constant inter-arrival interval. However, this effect, even if not relevant for Kademlia, is expected to have higher impact on systems such as BitTorrent [103].

20.8 Further Reading

A substantial number of work exists on modeling of user behavior for single P2P overlays, such as Gnutella and DHTs. Herrera et al. proposed a group-based model [203] to model churn in a P2P system, where peers are divided into three groups, benefactors, peers and peepers, showing different characteristics regarding their session and intersession lengths. The modeling is based on the *connectivity factor*, that specifies the fraction of all available peers being online at any given point of time. This value is probabilistic, because the session and intersession times of peers are probabilistic and therefore fluctuate around the desired network size defined by the connectivity factor. This connectivity factor is computed by $C_f = \frac{OS}{OS+IS}$ where OS is the average session length and IS is the average intersession length.

The performance of different Distributed Hash Tables (DHTs) under churn was analyzed by Rhea et al. [388]. Here additionally to the churn the arrival process, i.e., with which rate and distribution the peers connect to the system is considered. This model is suitable especially for search overlays and we use it for comparison. The interesting property of the churn model applied, is that it allows a stable size of the network, i.e., the number of peers being online is constant during the simulation time. This is achieved by replacing each peer going offline by another peer connecting to the network immediately. This is different to pure probabilistic models and allows more control of the simulation parameters.

A work on DHTs with mobile participants [519] distinguishes churn behavior for heterogeneous networks, including mobile peers. Such peers might use mobile phones for Internet access and therefore show much higher churn rates due to expensive online access and failure rate. The modeling here is based on peer classes with different mean online times, failure probabilities, number of shared objects and average query rate.

Specific models can be applied for applications such as Gnutella, where the query circle model was introduced by Schlosser et al. [405]. They focus mostly on the query inter-arrival times and content popularity. Another work on Gnutella from Aggarwal et al. [26] concentrates on the impact of realistic and extreme user behavior types on the system. In their model, content replication, session length and query strings are considered.

Andreolini et al. characterized the resources distributed in Gnutella network [37], especially the file types (video, audio, archives), their sizes and popularity distributions.

20.9 Conclusions

In this chapter, we have studied the modeling of user behavior when simulating P2P systems. We presented modeling alternatives to represent the user behavior in a P2P system. As an example, we showed how it can be applied to Distributed Hash Tables.

We studied and demonstrated the impact of different modeling approaches using Kademlia as use case. In particular, arrival rates and churn have significant impact on the system performance resulting in diverging drop and success rates.