2012

Systems with Session-based Workloads: Assessing Performance and Reliability

Nikola Janevski
West Virginia University

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Recommended Citation
https://researchrepository.wvu.edu/etd/3517

This Thesis is brought to you for free and open access by The Research Repository @ WVU. It has been accepted for inclusion in Graduate Theses, Dissertations, and Problem Reports by an authorized administrator of The Research Repository @ WVU. For more information, please contact ian.harmon@mail.wvu.edu.
Systems with Session-based Workloads: Assessing Performance and Reliability

by

Nikola Janevski

Thesis submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Jim Mooney, Ph.D.
Arun Ross, Ph.D.
Katerina Goseva Popstojanova, Ph.D., Chair

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2012

Keywords: Reliability, Performance, Web, Feedback Queue, Session-based Workloads

Copyright 2012 Nikola Janevski
Abstract

Systems with Session-based Workloads: Assessing Performance and Reliability

by

Nikola Janevski
Master of Science in Computer Science
West Virginia University
Katerina Goseva Popstojanova, Ph.D., Chair

Many systems, including the Web and Software as a Service (SaaS), are best characterized with session-based workloads. Empirical studies have shown that Web session arrivals exhibit long range dependence and that the number of requests in a session is well modeled with skewed or heavy-tailed distributions. However, models that account for session workloads characterized by empirically observed phenomena and studies of their impact on performance and reliability metrics are lacking.

For assessing performance, we use a feedback queue to account for session-based workloads in a physically meaningful way and use simulation to analyze the behavior of the Web system under Long Range Dependent (LRD) session arrival process and skewed distribution for the number of requests in a session. Our results show that the percentage of dropped sessions, mean queue length, mean waiting time, and the useful server utilization are all affected by the LRD session arrivals and the statistics of the number of requests within a session. The impact is higher in the case of more prominent long-range dependence. Interestingly, both the request arrival process and the request departure process are long-range dependent, even in the case when session arrivals are Poisson. This indicates that the LRD at the request level can be a result of the existence of sessions.

For assessing reliability, we propose a framework which integrates (1) the Web workloads defined in term of user sessions, (2) the user navigation patterns through the Web site, and (3) the reliability estimates of the Web requests based on the system architecture; then, we give a detailed reliability model of a Web system based on the proposed framework. We recognize the difficulty of solving the proposed model and use simulation to obtain the results. And last but not least, we use statistical design of experiment to quantify the results and to determine which factors have the highest impact on the system’s reliability. Our results show that some two-way and three-way interactions are very important for the session reliability of Web systems.
Acknowledgements

To Hedwig for showing me the way.

I would first like to thank my committee chair and advisor, Dr. Katerina Goseva Popstojanova, for her support and patient she offered me. Also, I would like to thank Dr. James Mooney and Dr. Arun Ross for being on my committee.

This work is funded in part by the National Science Foundation under the grants CNS-0447715 and CCF-0916284.
Contents

Acknowledgements iii

List of Figures vi

List of Tables viii

1 Introduction 1
   1.1 Performance .............................................................. 2
   1.2 Reliability ............................................................. 3
   1.3 Contributions ......................................................... 4

2 Related work 8
   2.1 Related work on performance modeling ............................. 8
   2.2 Related work on availability and reliability .................... 10

3 Background work 13

4 Performance of session-based systems 16
   4.1 Feedback queues ......................................................... 16
   4.2 Definition and solution of the model ............................... 17
   4.3 Analysis of the main findings ...................................... 20
      4.3.1 Effects of feedback queue and model distributions on request arrival/ departure process ................................................. 21
      4.3.2 Which factors affect the performance metrics? ............ 23

5 Reliability 38
   5.1 Session-based reliability modeling framework .................. 38
   5.2 Model of Web session reliability ................................... 40
      5.2.1 Integral user operational profile ............................. 40
      5.2.2 Requests’ reliability estimate .................................. 43
      5.2.3 Model integration and solution ................................ 46
   5.3 Experimental design and analysis ................................. 46
      5.3.1 Choice of factors and factor levels ......................... 48
      5.3.2 Statistical analysis of the effects ......................... 53
   5.4 Results ................................................................. 55
## List of Figures

4.1 General model of a queue with delayed feedback .......................... 17
4.2 System with session-based workload modeled as a queue with delayed feedback, where the delay system is represented as an infinite $G/G1/\infty$ queue .................. 18
4.3 Hurst estimates for geometric session length, exponential service time, and no think time. ................................................................. 26
4.4 Hurst estimates for geometric session, Pareto service time, and no think time. .... 26
4.5 Hurst estimates for geometric session, exponential service time, and deterministic think time. ................................................................. 27
4.6 Hurst estimates for geometric session, Pareto service time, and deterministic think time. ................................................................. 27
4.7 Hurst estimates for geometric session, exponential service time, and Pareto think time. ................................................................. 28
4.8 Hurst estimates for geometric session, Pareto service time, and Pareto think time. .... 28
4.9 Percentage of dropped sessions with deterministic service time and zero think time . 29
4.10 Percentage of dropped sessions with exponential service time and zero think time . 29
4.11 Percentage of dropped sessions with Pareto service time and zero think time. .... 29
4.12 Percentage of dropped sessions with deterministic service time and deterministic think time ................................................................. 30
4.13 Percentage of dropped sessions with exponential service time and deterministic think time ................................................................. 30
4.14 Percentage of dropped sessions with Pareto service time and deterministic think time ................................................................. 30
4.15 Percentage of dropped sessions with deterministic service time and Pareto think time ................................................................. 31
4.16 Percentage of dropped sessions with exponential service time and Pareto think time ................................................................. 31
4.17 Percentage of dropped sessions with Pareto service time and Pareto think time. .... 31
4.18 Mean queue length for deterministic service time and zero think time ............ 32
4.19 Mean queue length for deterministic service time and deterministic think time .... 32
4.20 Mean queue length for deterministic service time and Pareto think time ............ 32
4.21 Mean queue length with exponential service time and zero think time ............ 33
4.22 Mean queue length with exponential service time and deterministic think time .... 33
4.23 Mean queue length with exponential service time and Pareto think time ............ 33
4.24 Mean queue length with Pareto service time and zero think time ............ 33
4.25 Mean queue length for Pareto service time and deterministic think time ............ 34
4.26 Mean queue length for Pareto service and Pareto think time ..................... 34
4.27 Useful request utilization for deterministic service time and zero think time. .... 35
4.28 Useful request utilization for deterministic service time and deterministic think time 35
4.29 Useful request utilization for deterministic service time and Pareto think time . . . 35
4.30 Useful request utilization with exponential service time and zero think time . . . 36
4.31 Useful request utilization with exponential service time and deterministic think time 36
4.32 Useful request utilization with exponential service time and Pareto think time . . . 36
4.33 Useful request utilization with pareto service time and zero think time . . . . . . . 37
4.34 Useful request utilization for pareto service time and deterministic think time . . . 37
4.35 Useful request utilization for Pareto service and Pareto think time . . . . . . . . . 37

5.1 A framework for modeling session reliability . . . . . . . . . . . . . . . . . . . 39
5.2 Example of user navigation pattern based on the TPC-W benchmark . . . . . . . 41
5.3 Deployment diagram of an e-commerce site modeled by TPC-W . . . . . . . . . . 45
5.4 Sequence diagram for the Buy Confirm request . . . . . . . . . . . . . . . . . . . 45
5.5 Pseudo-code for the simulation . . . . . . . . . . . . . . . . . . . . . . . . . . . . 47
5.6 qq-plot of the residuals’ quantiles vs. the normal distribution quantiles. The bending at the edges means that the tails of the distribution of the residuals are heavier then the tails of the normal distribution. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 54
5.7 Residuals vs. fitted values. The funnel shape indicates that the constant variance assumption is not satisfied. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55
5.8 Main effects plot for session reliability . . . . . . . . . . . . . . . . . . . . . . . . 57
5.9 Interaction effect plot for session reliability . . . . . . . . . . . . . . . . . . . . . 59
5.10 Main effect plot for request-based reliability . . . . . . . . . . . . . . . . . . . . . 60
5.11 Interaction effect plot for request-based reliability . . . . . . . . . . . . . . . . . 62
List of Tables

4.1 Models/ distributions and the corresponding parameters . . . . . . . . . . . . . . . 18
5.1 Test Statistics for Session Reliability . . . . . . . . . . . . . . . . . . . . . . . . 56
5.2 Test Statistics for Request Reliability . . . . . . . . . . . . . . . . . . . . . . . . 61
Chapter 1

Introduction

The World Wide Web is arguably the biggest and most widely used distributed system. Since
its introduction, the Web has been popular due to the convenience of using a Web browser as a
GUI and the intuitive and uniform way to access abstract or physical resources. The ability to
update and maintain Web applications without distributing and installing software on potentially
thousands of client computers is a key reason for their popularity, as is the inherent support for
cross-platform compatibility [61]. Furthermore, the ubiquity of Web browsers, wide access to
high speed Internet connections, and fast pace development of new Web-based technologies have
led to an exponential growth of the number of users and services offered through the Web. Many
business and everyday activities nowadays depend on Web based systems and rely on their high
reliability, availability, and responsiveness. Therefore, it is of crucial importance to be able to
assess Web system performance and reliability realistically and assure the quality of service.

Traditionally, evaluation of Web server performance and reliability accounted for request-based
workloads and it was focused on assessment and prediction of request-based metrics (e.g., through-
put in number of completed requests, percentage of dropped requests, and so on). Web workloads,
however, are in a form of sessions, each consisting of multiple individual requests originated from
the same user. For example, placing an order on an e-commerce Web site involves requests relating
to selecting a product, providing payment and shipping information, and receiving a confirmation.
So, for a customer trying to place an order or a retailer trying to make a sale, the real measure
of a Web server quality is its ability to complete the entire sequence of requests within a session
without a failure and in a timely fashion [7].

In addition to the traditional large scale Web and e-commerce applications, new paradigms,
such as Software as a Service (SaaS) [2], [37] where software is delivered to customers as a ser-
vice on demand over the Web, are critically dependent on uninterrupted usage, with virtually no
downtime. Obviously, losing users due to reliability and/or performance issues results in signifi-
cant loss of revenue.

In the following discussion, we give overviews of performance and reliability.

1.1 Performance

The most common mathematical framework for modeling performance is queueing theory [22].
The two most popular queueing models are open and closed queues. In open queues, the users
arrive from outside, wait in the queue, get service, and leave the system. In closed queues, there
is a fixed number of users that after finishing service, return to the back of the queue to wait to
be serviced again. However, using either a closed or open queuing system with a request-based
workload does not account for session characteristics and therefore does not result in a realistic
model of a Web system. Instead, in this thesis we use feedback queue which accounts for session-
based workloads because this allows us to calculate metrics that represent the QoS at session level.
In this queue sessions arrive as in an open queue, but for each request within a session the user
sends a new request only after receiving a response on the previous request. In other words, a
feedback queue allows us to model a varying number of users at the site over time as in an open
system (rather than a fixed number of users \( N \) as in a closed system). On the other side, it behaves
as a closed system for the requests within each session. Upon completing all requests in a session,
the user leaves the system, again as in an open system.

In addition, we consider a queue with a finite size, since our interest is the throughput in suc-
cessfully completed sessions, that is, the percentage of sessions being dropped due to the queue
being full. When a server works under high utilization the queue length tends to grow to the
point when the queue becomes full resulting in dropping the incoming request. For a server that
runs session-based workloads a dropped request could be anywhere in the session, and will lead
to aborted, incomplete sessions. Obviously, the quality of service of a Web system, from both
user’s and provider’s perspective, is best assessed by the number of successfully completed ses-
sions. Considering only request-based workload does not allow for assessment of the percentage of
dropped sessions (i.e., unsatisfied users) or the amount of server utilization wasted on completing
requests from aborted sessions.

The realism of the performance assessment is not based only on the type of the queuing model
(i.e., open, closed, or feedback queues), but also on the models and distributions used for the asso-
associated random variables and the values of the corresponding parameters. In choosing these models, distributions, and parameters our work is motivated by recent empirical results. For example, recent studies on session Web workloads have shown that the arrival of Web sessions, for systems under moderate to high load, is a Long Range Dependent (LRD) process (i.e., asymptotically second order self-similar process) [60], which means that sessions arrive in bursts over many time scales. In addition, the number of requests in a session follows a skewed or heavy-tail distribution [18], [43]. Motivated by these empirical results, in our feedback queueing system we use LRD processes for session arrivals and then distributions with different coefficients of variation for the number of requests in each session.

1.2 Reliability

For Web users, another important quality attribute is reliability. For example, a report by the Boston Consulting Group showed that 41% of the users who experienced a failure stopped using that Web system [1].

The concept of session-based Web reliability, which is defined as the probability that a user session completes successfully, without experiencing a failure in any of the requests that constitute that session, was first introduced by [18]. In [18] and the follow-up work [21], session reliability was estimated empirically based on the data collected by several Web servers. A similar approach of log-based analysis of Software as a Service (SaaS) for the session reliability was recently used by [2].

The assessment of session reliability in this thesis is taking into account much broader perspective. In particular, we first propose a framework for modeling session reliability and then within this framework we build a specific model using assumptions and parameters values based on real Web system data. We solve the model using simulation and systematically explore the impact of different factors and their interactions on the session and request reliability using a design of experiments approach.

Our work is motivated by the fact that session reliability depends on many factors. Obviously, it depends on the user navigation pattern, that is, the types of requests the user could make and the probability of making them. The reliability of each request within a Web session is determined by the reliability of the components and the way they interact in serving that specific request. Thus, a faulty component will affect the session reliability only if the user makes a request that encounters that component. In other words, even if a component is faulty, if the probability of evoking it
is very low not many users will experience failures due to that component. The duration of user sessions in number of requests is also important for the session reliability because, provided that the Web system has faulty components, longer sessions typically have a greater chance to contain a request that may fail. It follows that determining the session reliability of a Web systems involves many factors that interact with each other, and these interactions cannot be neglected.

1.3 Contributions

The main contributions of this thesis related to the performance assessment metrics, which are percentage of dropped sessions, mean queue length, and useful request utilization, are as follows:

- We use a finite size feedback queue to account for session-based workloads. So far, the only paper in the literature on performance modeling that used feedback queues is [55]. However, [55] considered infinite queue size, Poisson session arrivals, and geometric distribution for the number of requests in a session. Furthermore, the mean response time was the only output metric explored in [55].

- Based on the empirical findings in [60], we model session arrivals with a long-range dependent (LRD) process. In addition, we use a Poisson session arrival process, which is obtained by reshuffling the LRD process to have independent arrivals, allowing for a fair comparison with models that assume Poisson session arrivals, such as [55].

- The number of requests within a session is modeled with discrete lognormal distribution with different means and coefficients of variation to explore the impact of these statistics on several performance metrics. We also use geometric distribution. In the related work which considered session-based workloads, the number of requests in a session was modeled with an exponential [7] or geometric distribution [55]. Both of these distributions have a tail that decays exponentially which is not the case with real Web workloads [18], [43].

- We use several performance metrics, such as the percentage of dropped sessions, mean queue length, mean waiting time, and useful utilization. In addition, we explore the nature of the request arrival process and request departure process, both of which are dependent not only on the session arrival process and the distribution of the number of requests per session, but also on the service and think time. These processes have not been studied in the related work on session-based workloads [7], [55].
Our results show that the percentage of dropped sessions, mean queue length, mean waiting time, and the useful server utilization are all affected by the LRD session arrivals and the statistics of the number of requests within a session. The impact is higher in the case of more prominent long-range dependence. Interestingly, both request arrival process and request departure process are long-range dependent, even in the case when session arrivals are Poisson, and think time and service time are exponential. This indicates that the LRD of the request arrival is a result of the existence of sessions and that feedback queues are a suitable model for Web traffic. Our findings have strong practical implications on the performance assessment of Web systems, as well as on developing scheduling policies and admission control policies.

The research on performance modeling is published in [26].

The main contributions of this thesis on the reliability metrics include:

- **We propose a hierarchical framework** for modeling session reliability of Web systems. This framework represents the user view at the session layer and the system view at the service layer. In particular, it integrates the **workload model** defined in terms of user navigation patterns through the Web site and session length in number of requests with the **reliability of each Web request** determined as a function of the reliabilities of components involved into serving that request and the way these components interact.

Since a hierarchical approach allows combination of different processes (e.g., execution and failure behavior) and/or different levels of granularity in building the model, it has been used for performability modeling [23], e-commerce performance assessment [40], as well as in some of the related work on Web availability [33], [31] and services reliability [54]. The framework proposed here adapts these concepts of hierarchical modeling and refines them to fit one of the objective of our study - reliability modeling of session-based systems.

Session reliability in the past has only been evaluated empirically, based on the data collected by front-end Web servers, by simply counting the number of erroneous requests within a session [18], [21], [2]. It appears that this is the first modeling framework which explicitly integrates the workload model with the component based reliability model of a multi-tier Web systems. The proposed framework is generic and can be used for modeling different systems that involve session based workloads (e.g., e-commerce, SaaS, etc) by building a model that accounts for the specific assumption of that particular domain.
Within this framework, we build a probabilistic model which accounts for the characteristics and assumptions specific for Web systems. The session layer model is based on the assumptions and parameters’ values of real Web workloads and widely used benchmarks. In particular, we model users navigation patterns using a Discrete Time Markov Chain (DTMC) inspired by the TCP-W specification [58]. However, unlike TCP-W which only allows for exponentially distributed session duration, we model the session duration in number of requests by a mixture of two distributions (i.e., lognormal for the body and heavy-tailed Pareto for the tail) based on the results of the previous empirical analysis of Web workloads [39], [20]. The service layer model, which is used to estimate the reliability of each request in a session, is based on the concepts of the architecture-based software reliability [19]. This layer takes into account the reliability of software components and the way they interact within a three-tier Web system consisting of front-end Web server, application server, and back-end database.

Instead of the rather limited sensitivity analysis based on varying one parameter at a time, typical for the modeling studies in the related work [31], [32], [17], and [62], we use design and analysis of experiments for quantitative evaluation of Web systems reliability. Unlike in more mature areas, such as medicine and physics, the use of formal design and analysis of experiments is not a common practice in computer science. Using formal experimental design allows us to conduct the analysis and evaluation in a sound context, including testing the statistical significance of the results. Furthermore, it allows us to quantify the impact that each factor and their interactions have on the Web system reliability. For example, our results show that in addition to some individual factors, some two-way and even three-way factor interactions may have significant impact on the Web systems reliability. This observation could not be made by taking the common approach of conducting sensitivity analysis based on varying one factor at a time.

Several interesting findings have potential practical value result from the investigation presented in this thesis. Longer sessions in the body of the distributions (represented by higher values of $\mu$) result in significantly lower session reliability. Surprisingly, the value of the tail index (i.e., $\alpha$) and the percent of points in the tail of the distribution ($1 - b$) do not affect the session reliability significantly. It appears that when the body of the number of requests in a session is characterized with a skewed distribution (e.g., lognormal), the contribution of the heavy tail does not decrease session reliability significantly. As expected, user navigation
profiles and requests’ reliabilities do affect the session reliability significantly. Another observation is that several two- and three-way factor interactions affect the session reliability noticeably. This is particularly important since such two and higher way factor interactions are not apparent when using the traditional one-factor-at-a-time approach. Last but not least, compared to session reliability, the request-based reliability provides an optimistic estimate and it is much less sensitive to the changes of the factors and their interactions. These observations clearly support the argument that session reliability provides better representation of the users view on the Web systems quality than request-based reliability.

The research on the reliability of session-based systems is currently under review [27].
Chapter 2

Related work

In the following two sections we summarize related work. Section 2.1 contains the related work on performance modeling and Section 2.2 contains the related work on availability and reliability modeling.

2.1 Related work on performance modeling

Building realistic models for Web systems would be impossible without empirical studies on their workloads. Empirical analysis of session-based Web workloads was performed in [18], [21], and [20]. An empirical study of inter-session and intra-session characteristic of Web workloads was done in [18], [21]. The workload metrics considered were session duration (in time units), number of requests per session (i.e., session duration in number of requests), and the number of bytes transferred per session. Based on the empirical results of eleven Web servers in the follow-up work [20] it was shown that the number of requests per session exhibits heavy tailed behavior with large number of short sessions and a small number of sessions with a very large number of requests.

Queueing models are the most common mathematical formalism used for modeling performance of Web systems. Mainly, there are two types of models used: closed, and open queueing networks. A feedback queue is a third model which has rarely been used and for which not much theoretical results are available.

Closed queueing networks have been used for modeling performance of Web systems in [46], [5], [59], [6]. In [46] the authors used simulation to examine how the autocorrelation propagates through a closed queueing network. Their results showed that if the service time at one queue
is correlated the correlation will propagate through the whole queueing network. However, the proposed model did not include the users think time. If the think time is added the autocorrelation might not propagate because adding a think time performs random translation of the stochastic process and it can ruin the autocorrelation.

An Approximate Mean Value Analysis (AMVA) algorithm for analysis of closed queueing networks with service time modeled as a MAP process was proposed in [5]. The solution gave approximate results for the mean utilization and mean response time.

Modeling mean response time and throughput in a closed queueing network model of multi-tier Web system was done in [59]. Each tier was modeled with a separate queue with processor sharing discipline. Each queue had a transition to its predecessor; after a request has finished processing at that tier it could go back to the previous tier or continue to the next tier. The queueing network was fed by an infinite server queue where each server represented a think time for a user. Since this is a closed queueing model, the number of users in the system is fixed and does not allow for session characteristics to be incorporated in the model. The service times and think times were assumed to be exponential and as a result the Mean Value Algorithm (MVA) was used for solving the model.

All these models assumed fixed number of users (typical for closed queueing systems), request-based workloads, and infinite queues.

Open queueing models have also been used for modeling Web systems performance, e.g. in [31], [32], and [29]. One of the earliest papers that considered the request loss probability [31] used single open queue for modeling a Web server. The request arrivals were modeled with Poisson arrivals and exponentially distributed service time. A follow-up paper [32], which also used single queue with Poisson request arrivals and exponential service time, considered the request unavailability due to long response time.

In [29] authors presented approximate analytical results for the queue length, request loss, and waiting time, for an open queue where the request arrival process was modeled with fractional Brownian motion (fBm) and the service time with long-tailed distribution.

Even though some of the open queue models considered finite queues and/or LRD arrival process, all of them accounted for request-based workloads only.

It appears that the only studies that took into account session-based workload are [6], [7], and [55]. In [6], authors proposed an overload control mechanism for closed queues with geometrically distributed numbers of requests in a session. Session-based admission control mechanism was proposed in [7], where the number of requests in a session was modeled with an exponential distribution, while the session arrival process was not explicitly specified. The goal of [55]
was to study the difference between open and closed queueing models and to explore the use of a feedback queue as a model for systems with session workloads. The session arrivals in [55] were modeled with a Poisson process, the service time with a hyperexponential distribution, the number of requests per session was assumed to follow a geometric distribution, and the queue size was infinite. The output metric considered was the mean response time. The request arrival and departure processes were not studied.

2.2 Related work on availability and reliability

Before embarking on a discussion of the related work on reliability and availability modeling, we provide definitions of these two similar but distinct quality attributes. (The definitions are taken from [22].)

**Definition 1.** Reliability is defined as the probability that the software will not cause failure of the system for a specified time under specified conditions.

**Definition 2.** Availability is the ability of a system to perform its required function at a stated instant or over a stated period of time. It is usually expressed as the availability ratio, i.e., the proportion of time that the service is actually available for use by the customers within the agreed service hours.

We discuss the related work on availability because the modeling approaches for assessing reliability and availability are very similar. An analytical modeling approach of Web availability was taken in [33], [31], [32], [17] and [62], while a measurement-based approach was adopted in [45]. Below, we summarize the main contributions of these studies.

A framework for modeling availability of e-business systems, based on a hierarchical approach, was proposed in [33]. This framework included the user operational profile, the functions provided by the system, the services needed to implement each function, and the resources required by each service.

The unavailability due to long response time was modeled in [32]. The performance models used in this work were $M/M/1$ and $M/M/1/b$ for the single server Web system, and $M/M/c$ and $M/M/c/b$ for the multi-server Web systems. All these models assume that request arrivals follow a Poisson process and that the service time is exponentially distributed. A model of an e-commerce travel agency based on the hierarchical framework [33] was presented in [31]. This
model included the impact that performance issues (i.e., probability of dropping a request due to full buffer) have on availability as in [32]. In addition, hardware failures with perfect and imperfect coverage were modeled with a Continuous Time Markov Chain. The numerical results included sensitivity analysis of model parameters.

An availability model of an e-commerce site, which used a Discrete Time Markov Chain (DTMC) based on the Customer Behavior Model Graph (CBMG) adopted from [41] to represent the user navigation patterns, was presented in [17]. Considering availability at a request level and following the approach previously used to compute software reliability [16], the expected availability and expected session length were derived using the theory of DTMCs. It should be noted that in [17] the session length is restricted by the absorbing Markov chain and its expected value is an output variable of the model.

In [62], the activity of a Web user was modeled as an ON-OFF process, with the ON (active) period having a Weibull distribution and the OFF (thinking) period following a Pareto distribution. Markov Regenerative Process (MRGP) was used to model availability, where each state represented the user status (active, thinking, and seeing a failure) and the platform status (up or down). This work did not consider the user navigation pattern (i.e., different types of active states). Furthermore, the Web platform was modeled as a single unit that can either be up or down (with three types of possible failures near-user, in-middle, near-host), without considering components of which the system was built.

An empirical analysis of Web system availability from an end-user perspective was conducted in [45]. The authors used a tool developed in Java to monitor several Web sites (i.e., an online retailer, search engine, and directory service) by periodically sending requests to these Web sites. The monitoring lasted for six months. In addition to the raw (‘success’ or ‘failure’) availability, the authors estimated several other cases, such as availability ignoring local problems, availability ignoring local and network problems, and availability ignoring local, network, and transient problems.

We next summarize the work on analytical modeling of composite services reliability presented in [54], and several papers which were focused on empirical studies of Web servers reliability [57], [18], [21], and reliability of software as a service [2].

[54] proposed an analytical model in which the control flow among services was modeled with a discrete time Markov chain (DTMC). In this work only failures of hardware resources were considered; they were modeled using a continuous time Markov chain (CTMCs), where different states reflect different levels of resource reliability (e.g., the ‘down’ state of a resource indicates
it is unreliable). The two models were then integrated using Markov reward models (MRM), to calculate the overall reliability of the composite Web service.

An empirical characterization of Web reliability was done by [57]. In this study, the authors used server logs from two Web sites, one educational and one for open source development. This work considered only source content failures that prevent the acquisition of the requested information by Web users because of problems such as missing or unaccessible files, or trouble with starting JavaScript. The metrics considered were number of hits, bytes transferred, users, and user sessions. The estimation of the reliability was based on the Nelson model and Goel-Okumoto model from software reliability theory.

Session reliability was first introduced in [18] as a metric that represents users perception of the Web site reliability and it was estimated as the percentage of sessions completed without a failure of any requests within that session. The empirical results were based on data collected by the front-end Web servers of eleven Web sites.

In a recent work by [2], the concept of session reliability was used in the context of Software as a Service (SaaS). Similarly as in [18], in [21] the assessment was done based on empirical data collected by the front-end Web server. The authors noted that longer sessions have higher probability of encountering errors and concluded that “session reliability offers a reliability measure highly relevant to and easily understood by SaaS customers.”
Chapter 3

Background work

Heavy-tailed distributions are commonly found in network and Web traffic. They have been used for modeling interarrival times, file sizes, service time, etc. Here we give a definition of heavy-tail distribution (The discussion on heavy-tail distribution is based on [51]).

**Definition 3.** The random variable $X$ has a heavy-tail distribution if there exists $\alpha > 0$ such that:

$$P[X > x] \sim x^{-\alpha}, \quad x \to \infty$$  \hspace{1cm} (3.1)

The parameter $\alpha$ is called the index of the tail.

Heavy-tail distributions are also called power law distributions because the tail is a power function of $x$. It is interesting to observe the moments of a heavy-tail distribution.

$$\int_0^\infty x^{\alpha-1} P[X > x] dx \approx \int_1^\infty x^{\beta-1} x^{-\alpha} dx \begin{cases} < \infty & \text{if } \beta < \alpha \\ = \infty & \text{if } \beta \geq \alpha \end{cases}$$  \hspace{1cm} (3.2)

It follows that the moments above the $\alpha$th do not exist. So if $\alpha < 1$ then all the moments of the distribution are infinite. If $1 \leq \alpha < 2$ the distribution has finite mean but infinite variance. This type of distributions are at the same time interesting and challenging since the probability the random variable will have a large value is non-negligible.

Another interesting phenomenon occurring in network and Web traffic is self-similarity. Self-similarity is defined as [48]:

**Definition 4.** A stochastic process $Z = \{Z(t)\}_{t \in \mathbb{R}}$, that takes real values, is self-similar with Hurst exponent $H > 0$ (H-s) if, for any $a > 0$,

$$\{Z(at)\}_{t \in \mathbb{R}} \overset{d}{=} \{a^H Z(t)\}_{t \in \mathbb{R}}$$  \hspace{1cm} (3.3)
where \( d \) mean equality of the finite-dimensional distributions. In plain English, self-similarity means that no matter at what scale (ms, sec, min, hours, etc.) we observe the process it will still follow the same distribution. Practically, it means that the variance of the process does not go away even if we observe the process at larger time scales.

Long-range dependence is another property that goes hand in hand with heavy-tail distributions and self-similarity. It is defined as [51]:

**Definition 5.** A stationary sequence \( \{X_n, n \geq 1\} \) is long-range dependent (LRD) if

\[
\text{Cov}(X_n, X_{n+h}) \sim h^{-\beta} L(h), \quad h \to \infty,
\]

(3.4)

where \( 0 < \beta < 1 \) and \( L(\cdot) \) is slowly varying function.

Basically, long-range dependence means that there is significant dependence between the values of the process even at very large time lags. This makes the analysis more difficult because the casual assumption of independence cannot be used. If a process is long-range dependence then it is also asymptotically second order self-similar. However, if a process is self-similar it does not mean that it is LRD, e.g. fractional Brownian motion with \( H = 0.5 \) is self-similar but not LRD.

A common explanation for LRD is that heavy-tails cause LRD [51].

When talking about LRD there are two types of LRD, long-range interval dependence (LRiD), and long-range count dependence (LRcD). LRiD means that the values of the interarrival/interdeparture times are LRD while, LRcD means that the associated count process is LRD. In this thesis we are looking at LRcD.

One of the most important stochastic processes is Brownian motion. Brownian motion is defined as [34]:

**Definition 6.** Brownian motion, \( B(t) \), is characterized by the following properties:

1. \( B(0) = 0 \)
2. \( B(t) \) is almost surely continuous
3. \( B(t) \) has independent increments with distribution \( B(t) - B(s) \sim \mathcal{N}(0, t-s) \) for \( 0 \leq s \leq t \)

A generalization of Brownian motion is fractional Brownian motion (fBm) which is self-similar. The fBm is defined as:

**Definition 7.** Fractional Brownian motion, \( B(t) \), is characterized by the following properties [48]:
1. \( B(0) = 0 \)

2. \( B(t) \) is almost surely continuous

3. \( B(t) \) has independent increments with distribution \( B(t) - B(s) \sim \mathcal{N}(0, t-s) \) for \( 0 \leq s \leq t \)

4. \( B(t) \) has a covariance function
\[
E[B(t)B(s)] = \frac{1}{2}(|t|^{2H} + |s|^{2H} - |t-s|^{2H}),
\]
where \( 0 < H \leq 1 \) and \( H \) is called the Hurst index or exponent.
Chapter 4

Performance of session-based systems

4.1 Feedback queues

For systems with session based workloads, a queue with delayed feedback represent a very intuitive mathematical model because it captures the natural feedback inherent in a user session. For example, the user will request the home page, after the server processes the request, the user will spend sometime reading the content, and then make another request for a page linked on the home page. This process can continue until the user completes the session.

A general feedback queue is shown in Figure 4.1 [15]. The arrival process represents the arrival of new users to the system. After going through the queue and receiving service the user can return back to the system through the feedback loop; however, the user goes through the delay system where he/she is delayed for a random period of time. After the delay, the user returns to the queue. For Web systems the delay is called think time. The process of users returning to the system is called the return process and together with the arrival process forms the input process to the queue. Users that finish service, form the output process. Some of these users will return through the feedback loop and this forms the feedback process while, others will leave the system, which forms the departure process. How many times the user will return to the system for service determines the length of the sessions and can be represented by a probability distribution.

Previous studies [7], [58], have considered the session length in time as a measure of the time the user spends using the Web system; however, this metric does not take into account that the time the user spends at a given Web system depends on the responsiveness on the Web system since an overloaded Web system will have slow response time and the session lengths will tend to be longer. On the other hand, the number of requests in a session is a more adequate metric since it
does not depend on the responsiveness of the system. Of course, if the system is very slow a user might leave the system and this will affect the number of requests in a session but, this is true of the session length, also.

Besides being a more accurate measure, the distribution of the number of requests per session is necessary for modeling the Web system as a delayed queue with feedback and it allows for the QoS, as experienced by the user, to be measured. For example, the percentage of sessions that will be dropped if the system is overloaded can be obtained by using a feedback queue.

The arrival process of the feedback queue represents the session arrival. The feedback process represents the users that are making additional requests. The departure process models users that have completed their visit to the system. Before making an additional request, each user spends some time, called think time, processing the content of the previous request. This is the delayed system and for session-based systems it can be modeled as an infinite server queue, $G/GI/\infty$, with general arrival process and i.i.d. service time that represents the users think time. This is shown in Figure 4.2.

### 4.2 Definition and solution of the model

The models/distributions and corresponding parameters used for the random variables associated with the feedback queue are given in Table 4.1 and are briefly described next.

Motivated by the empirical findings for Web servers working under moderate and heavy workloads [60], we use LRD process to model *session arrivals*. Web traffic has dual nature [36], that is, both the number of sessions per second (i.e., the count) and the inter-arrival time can be LRD.
In order to achieve the dual nature of the session arrivals, we use the method of inverse transformation proposed in [28]. A LRD process, \( \{Y_k\} \), with a marginal cumulative distribution function (CDF), \( F_Y(y) \), can be generated from another LRD process, \( \{X_k\} \), with CDF \( F_X(x) \) using the transformation:

\[
\{Y_k\} = F_Y^{-1}(F_X(X_k)), \quad k = 1, 2, ...
\]

(4.1)

where, \( F_Y^{-1} \) is the inverse CDF of \( \{Y_k\} \). This transformation actually first transforms the sequence \( \{X_k\} \) into a uniformly distributed random variable \( F_X(X_k) \sim U(0, 1) \) and then generates the sequence \( \{Y_k\} \) using the inverse CDF, \( F_Y^{-1} \), of the desired marginal CDF, \( F_Y \) [52]. In addition to generating a sequence with desired marginal distribution, the LRD is also preserved [28]. For our simulation \( \{X_k\} \) is the Fractional Gaussian Noise (FGN) with \( F_X(x) \) normally distributed and \( F_Y(y) \) is exponentially distributed. The FGN was simulated using the FFT method proposed
by Paxson [49] because it provides fast way to simulate FGN and still preserves all the relevant statistical properties [14]. The resulting process, \{Y_k\} defined with equation (4.1) is a LRD process with exponentially distributed inter-arrival times. This enables us to obtain a Poisson process for session arrivals by reshuffling of \(Y_k\) and thus carry on fair comparison with the LRD arrival process. For the Hurst exponent of the LRD session arrivals, \(H_{\text{session}}\), we used two values \{0.6, 0.8\}.

For the number of requests in a session we use both the geometric and the discrete lognormal distribution [10].

The probability mass function of the discrete lognormal distribution \(P(X = r)\) is given by

\[
P_r(\mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi} r!} \int_0^\infty e^{-\lambda/r} \exp \left( -\frac{(\ln \lambda - \mu)^2}{2\sigma^2} \right) d\lambda
\]

where \(\mu\) is the location parameter and \(\sigma\) is the shape parameter. (For the relationship between these parameters and the mean and standard deviation see [10].)

The discrete lognormal distribution is flexible; it can decay slower then the geometric distribution and have higher variance, or it can decay faster and have lower variance. This allows us to examine the impact that the coefficient of variation has on the performance metrics of interest. For the mean of the number of requests in a session, based on the findings in [18] and [55], we use three values \{2, 6, 11\} requests. For the coefficient of variation for the discrete lognormal distribution, \(C\), defined as the ratio of the standard deviation to the mean we use the values \{0.5, 1.5\}. Thus, discrete lognormal distribution with \(C = 0.5\) has lower variance then the geometric (or exponential) distribution, while for \(C = 1.5\) it has higher variance.

The probability mass function of the geometric distribution \(P(X = r)\) is given by

\[
P_r(\pi) = p(1 - p)^r, \ r = 1, 2, ...
\]

where \(p\) is the probability that the user will end the session. The mean of the geometric distribution is \(\frac{1}{p}\), the variance is \(\frac{1-p}{p^2}\), and the coefficient of variance is \(\sqrt{1 - p}\).

For the service time, motivated by [9], we use a Pareto distribution. Also, we use a deterministic distribution and an exponential distribution with the same mean since we want to see the effect that the distribution of the service time has on the LRD and the performance metrics. For the mean value of the service time, \(\mu_s\), we use 57 ms which is in the range of values reported in [55], while, for the tail of the Pareto distribution we use, \(\alpha_s = 1.6\), because for this value the distribution is heavy-tailed and also the simulations are stable [9].

For the think time we use a Pareto distribution which is in agreement with empirical and theoretical research [3]; and a deterministic distribution with the same mean. The mean value
\( \mu_t = 5000 ms \) is chosen as in [7] and \( \alpha_t = 1.6 \). In addition to the Pareto and the deterministic distribution, we also perform simulation in which we set the think time to be equal to zero, i.e. we use a feedback queue with no think time. This allows us to examine the impact that the think time has on the LRD and the performance metrics.

Finally, for the queue size (i.e., the maximum number of requests in the queue) we use 511 requests, which is the default value for Apache [35]. The scheduling policy used is FCFS. We keep these two parameters fixed in our simulations since they are not of interest.

Not much work exists on analytical solution of feedback queues. Using LRD session arrival processes, with skewed distributions for the number of request per session, and finite queue size imposes using simulations to solve the feedback queuing system. In the simulation, sessions arrive as in an open system. If the session has more than one request, the next request is generated after the first request was served and an amount of think time has passed. In other words, requests belonging to a same session are processed as in closed system. Of course, at any point of time there may be multiple active sessions in the feedback queue. If a request arrives but the queue is full then that request and the session it belongs to are dropped.

We wrote a program in R language to run the simulations. The removal of the transient warm-up and cool-down periods from the simulations was done by visual inspections of the request arrival process. The validation of the simulation was done by checking the limiting cases, i.e., for the number of requests in a session equal to one when the feedback queue becomes an open queue, as well as for a very high number of requests in a session when it becomes closed queue [55].

### 4.3 Analysis of the main findings

Analysis of the main findings is organized around the following research questions.

- **RQ1:** Why is a feedback queue an appropriate model for session-based workloads and to what extent does the service time, think time, or the LRD of the session arrival affect the LRD of the request arrival/departure process?

- **RQ2:** Which factors, such as service time, think time, number of requests in a session, and session arrival process, affect the performance metrics?
4.3.1 Effects of feedback queue and model distributions on request arrival/departure process

In this section, we explore what is the impact of the feedback loop and to what extent it affects the request arrival process, i.e. whether the LRD of the request arrival process can be a result of the existence of sessions.

Although a feedback queue seem like an intuitive model for session-based workloads, for it to be a satisfactory model it has to be in agreement with the vast amount of empirical research which has shown that Web traffic is LRD at the request level [8], [39], and [60]. To examine whether this property is satisfied by our model we perform a series of simulations where the number of requests in a session is geometrically distributed with three different means 2, 6, 11 and for the session arrival process we use three cases: Poisson arrivals, and LRD arrival processes with $H = 0.6$, and $H = 0.8$.

First, we set the service time to simple exponential distribution, and the think time to zero. This results in a feedback queue with no delay, i.e. once the user’s request is served and the number of requests in the session is not depleted, it is moved to the back of the queue for additional service. Setting the parameters this way allows us to see whether the feedback of the queue introduces any LRD dependence to the request arrival process. The results for the Hurst exponents are shown in Figure 4.3.

There are three cases that can be distinguished from Figure 4.3, when the utilization is [0.5, 0.9), [0.9, 1.0) and 1.0.

In the first interval [0.5, 0.9), both request arrival and departure process are LRD. The reason for the LRD can be explained by observing that the interdeparture time, $D_n$, of the output process is the sum of the idle time, $V_n$, and the service time, $S_n$, i.e. $D_n = S_n + V_n$. When the utilization is lower then 0.9 the idle time contributes significantly to the interdeparture time and increases its variability. Also, it has been shown that a process is LReD if and only if the interdeparture time has infinite variance [11]. Another observation that can be made about this interval is that the value of the Hurst exponent slightly increases as the mean number of requests increases. This is especially noticeable for the case when the session arrival process is Poisson. When the mean number of requests in a session is 2 the LRD is very low because mean of 2 requests per session implies that not many requests return for service and as a result the effect of the feedback is less noticeable. In other words, the queue acts more like open queue and the session arrival process dominates the requests arrival.
In the second interval \([0.9, 1.0)\), it is easily observable that the estimates for the Hurst exponent approach 0.5, or simply stated the LRD slowly decreases. The only exception is when the session arrival process is LRD with \(H = 0.8\) and the mean number of request in the session is two. The slowly decrease of the LRD of the request arrival process can be explained by noticing that the heavy-tailness of the request interdeparture time decreases as the utilization increases because the idle time approaches zero. For the case when \(H = 0.8\), and the mean number of request in session is two, the request arrival process still exhibits LRD. The explanation is that not many request return to the queue and as a result the session arrival process dominates the input process, and as a result the request arrival process is LRD.

The last case is when the utilization approaches one. Under high utilization the idle time is zero and the interdeparture process is pure Poisson process. Again, when the session arrival process is LRD with \(H = 0.8\) and the mean number of requests in a session is two the request arrival process is still LRD. The explanation is the same as previously.

The previous observations strongly indicate that the LRD of the request arrival process for utilization in the interval \([0.5, 1.0)\) is due to the existence of a feedback in the queue. In other words, having sessions causes the request arrival process to be LRD even when session arrivals are Poisson. Our previous observations are for the case when the service time is exponential. Next we relax some of these assumption by setting the service time to follow a heavy-tailed, namely, the Pareto distribution with tail index \(\alpha = 1.6\).

Since some studies [24], [56] have emphasized that the LRD of the request arrival model is due to the heavy-tailness of the service time or think time, we performed simulations where we set the service time to follow the Pareto distribution, with the same value for the mean as in the case for the exponential service time. The results are shown in Figure 4.4. The explanations are the same, except for the case when the utilization is 1. In this case, the idle time approaches zero but, the service time is heavy-tailed and as a result the interdeparture time of the output process is also heavy-tailed. This causes the count process associated with the departure process to be LRD. Another interesting observation is that the heavy-tailed service time does not affect significantly the LRD of the request arrival process in the interval \([0.5, 0.9)\), i.e. no matter whether the distribution of the service time is heavy-tailed or simple exponential the request arrival and departure processes are LRD. This means that the LRD of the request arrival and departure process is caused mainly by the feedback, i.e. the session workload. Previous studies in network traffic have proposed different models to explain the LRD of the request arrival process, however to the best of our knowledge, none of these studies have explored the possibility that the LRD is due to the existence of sessions.
This is the first study to establish this fact.

Based on our results we conclude that the answer to our question is that the feedback queue causes the LRD of the traffic and is a good model of session-based workload. Also, the service time affects the LRD only when the utilization is approaching one.

Another question we want to answer is whether the think time affects the LRD of the processes. In order to answer this question we performed simulations by using the same parameters as before and varied only the think time. For the think time we used a deterministic distribution and a Pareto distribution. The results are shown in Figure 4.5, Figure 4.6, Figure 4.7, and Figure 4.8.

From the figures, it is easy to notice that the think time does not significantly affect either the request arrival process or the request departure process. This surprising result can be explained by observing that the think time only performs a random translation of the returning requests. In other words, it only translates the arrival times for the requests but it does not significantly modify the request interarrival time.

Other factors (service time, think time, session arrival process, and the distribution of number of requests) may affect the performance metrics and that is the next question we are will consider.

### 4.3.2 Which factors affect the performance metrics?

The first performance metric we examine is the **percentage of dropped sessions**. In Figure 4.9, Figure 4.10, and Figure 4.11 are shown the plots for the percentage of dropped sessions when the think time is zero and and the service time is deterministic, exponential, and Pareto, respectively. From the figures it is obvious that the service time does not make a significant difference in the percentage of dropped sessions when the think time is zero.

We also performed experiments with deterministic and Pareto think time. The results from the simulations with deterministic think time are shown in Figure 4.12, Figure 4.13, and Figure 4.14, and the results from the simulations with Pareto think time are shown in Figure 4.15, Figure 4.16, and Figure 4.17. All of these figures strongly indicate that the think time does not significantly affect the percentage of dropped sessions.

Looking at the plots, the same conclusion can be drawn for the service time. These unexpected observations can be explained by looking at the request arrival process. The LRD of the request arrival process is a result of the presence of a feedback, i.e. it feeds backwards the thinned request departure process. The think time only performs a random translation of the request departure process, but it does not significantly affect its LRD properties. Unlike the think time, the service
time plays a role in the request interdeparture time, however, the interdeparture time is a sum of the idle time and the service time and under loads in the interval \([0.5, 0.9)\), the idle time dominates the service time so the effect of the service time is not noticable. When the load approaches one, the request interdeparture time is determined by the service time, but under this high load the queue is already saturated and it will start dropping requests even if the request arrival process is not LRD.

On the other hand, the LRD of the session arrival process and the mean number of requests in a session makes significant difference on the percentage of dropped sessions. The percentage of dropped sessions is highest for LRD session arrivals with high value of the Hurst exponent \(H_{\text{session}} = 0.8\). The percentage of dropped sessions for less self-similar arrivals (i.e., \(H_{\text{session}} = 0.6\)) is very close to the case with Poisson session arrivals. In particular, for a high utilization values, the 10-15\% more sessions are dropped for highly self-similar process (i.e., \(H_{\text{session}} = 0.8\)) then when session arrivals follow the Poisson distribution obtained by reshuffling the LRD process.

Moreover, the percentage of dropped sessions is larger for higher mean number of requests per session, with a more noticeable impact of the LRD, observable for lower utilization values than in case of mean number of request per session equal to two requests.

Next, we explore the impact that the factors have on the mean queue length. The results for deterministic service time are shown in Figure 4.18, Figure 4.19, and Figure 4.20. The results for exponential service time are shown in Figure 4.21, Figure 4.22, and Figure 4.23. The results for Pareto service time are shown in Figure 4.24, Figure 4.25, and Figure 4.26.

The distribution of the think time and service time do not make a significant difference on the mean queue length. On the other hand, the mean queue length is affected by the LRD of the session arrival process, with the highest mean queue length for the high self-similar session arrivals \((H_{\text{session}} = 0.8)\), and less significant difference between \((H_{\text{session}} = 0.6)\) and Poisson arrivals. The reason for this behavior is the fact that under the LRD model sessions arrive in bursts and tend to fill in the queue fast, resulting in larger mean queue length. Also, it can be observed that smaller mean number of requests per session (i.e., 2 requests) results in larger mean queue length, especially noticeable for high values of Hurst exponent \((H_{\text{session}} = 0.8)\). This is due to the fact that there are smaller number of dropped sessions when the mean number of requests per session is smaller, which actually increases the number of requests in the queue.

The last performance metric we explore is the useful request utilization. Unlike the previous performance metrics, where the think time did not make a difference, for the useful request utilization, the think time does make a difference. When the think time is zero the useful request utilization is almost 100\%, while for deterministic and Pareto think time with equal means the
useful request utilization decreases by almost 10%. To understand why such a difference exists we have to notice that in the case when the think time is zero the requests follow each other closely compared to the case when the think time is deterministic or Pareto with mean 5000 ms. When the mean of the think time is 5000ms there is significant delay between two consecutive request from the same session. When the think time is zero the session lasts a short time and either the session runs into a busy period and at the very beginning it is dropped or all the requests belonging to that session are successfully completed. On the other hand, sessions with think time between requests are equally likely to be interrupted in the middle or the end, and as a result it is more likely for some useful work to be lost.

Highly LRD session arrivals can have $5 - 8\%$ lower useful request utilization then in case of Poisson session arrivals. Also, the higher the mean number of requests in a session the lower the useful request utilization because the server wastes resources on completing requests in longer sessions that are dropped.

**Does the service time affect the performance metrics?**

Next, we want to see if the service time affects any of the performance metrics.

The results for the **percentage of dropped sessions** for zero think time and deterministic, exponential, and Pareto service time are shown in Figure 4.9, Figure 4.10, and Figure 4.11, respectively. Results for deterministic think time and deterministic, exponential, and Pareto service time are shown in Figure 4.12, Figure 4.13, and Figure 4.14, respectively. Results for Pareto think time and deterministic, exponential, and Pareto service time are shown in Figure 4.15, Figure 4.16, and Figure 4.17, respectively.

Surprisingly, it appears that the service time does not affect the performance metrics significantly. This can easily be explained by remembering that the LRD of the request arrival processes does not depend significantly on the service time but is due to the feedback, and the request arrival process is LRD no matter the distribution used for the service time.
Figure 4.3: Hurst estimates for geometric session length, exponential service time, and no think time.

Figure 4.4: Hurst estimates for geometric session, Pareto service time, and no think time.
a) Arrival process
b) Departure process

Figure 4.5: Hurst estimates for geometric session, exponential service time, and deterministic think time.

a) Arrival process
b) Departure process

Figure 4.6: Hurst estimates for geometric session, Pareto service time, and deterministic think time.
a) Arrival process

b) Departure process

Figure 4.7: Hurst estimates for geometric session, exponential service time, and Pareto think time.

a) Arrival process

b) Departure process

Figure 4.8: Hurst estimates for geometric session, Pareto service time, and Pareto think time.
Figure 4.9: Percentage of dropped sessions with deterministic service time and zero think time

Figure 4.10: Percentage of dropped sessions with exponential service time and zero think time

Figure 4.11: Percentage of dropped sessions with Pareto service time and zero think time
Figure 4.12: Percentage of dropped sessions with deterministic service time and deterministic think time

Figure 4.13: Percentage of dropped sessions with exponential service time and deterministic think time

Figure 4.14: Percentage of dropped sessions with Pareto service time and deterministic think time
Figure 4.15: Percentage of dropped sessions with deterministic service time and Pareto think time

Figure 4.16: Percentage of dropped sessions with exponential service time and Pareto think time

Figure 4.17: Percentage of dropped sessions with Pareto service time and Pareto think time
Figure 4.18: Mean queue length for deterministic service time and zero think time

Figure 4.19: Mean queue length for deterministic service time and deterministic think time

Figure 4.20: Mean queue length for deterministic service time and Pareto think time
Figure 4.21: Mean queue length with exponential service time and zero think time

Figure 4.22: Mean queue length with exponential service time and deterministic think time

Figure 4.23: Mean queue length with exponential service time and Pareto think time
Figure 4.24: Mean queue length with Pareto service time and zero think time

Figure 4.25: Mean queue length for Pareto service time and deterministic think time

Figure 4.26: Mean queue length for Pareto service and Pareto think time
Figure 4.27: Useful request utilization for deterministic service time and zero think time

Figure 4.28: Useful request utilization for deterministic service time and deterministic think time

Figure 4.29: Useful request utilization for deterministic service time and Pareto think time
Figure 4.30: Useful request utilization with exponential service time and zero think time

Figure 4.31: Useful request utilization with exponential service time and deterministic think time

Figure 4.32: Useful request utilization with exponential service time and Pareto think time
Figure 4.33: Useful request utilization with Pareto service time and zero think time

Figure 4.34: Useful request utilization for Pareto service time and deterministic think time

Figure 4.35: Useful request utilization for Pareto service and Pareto think time
Chapter 5

Reliability

5.1 Session-based reliability modeling framework

Session-based reliability metrics provide a way to assess users’ view of Web systems reliability. It can be interpreted as the probability that a user of a Web system will complete her/his session successfully, without experiencing a failure in any of the requests that constitute that session [18], [21]. In this thesis, we propose a framework for modeling session reliability of multi-tier Web systems, which consist of front-end Web server (e.g., Apache or Microsoft IIS), application server that implements the logic, and back-end database server. What makes the proposed approach unique and different from the existing research work in the literature is the aspect of integrating within a single framework (1) the Web workloads defined in term of user sessions, (2) the user navigation patterns through the Web site, and (3) the reliability estimates of Web requests based on the system architecture.

The proposed framework is shown in Figure 5.1. The session layer describes the users view on the Web system and integrates information such as the number and types of requests within a user session. Each user request in the session layer model is served by invoking a set of components, possibly from different tiers of the Web system. The service layer in our approach represents the system view and considers the software components of the front-end Web server, application server, and database server, and the way they interact in serving each request. This layer provides the reliability estimates of each request to the Session layer. The thick arrow in Figure 5.1 represent the model creation in a top-down fashion. Basically, this process consists of mapping the structural information from the higher layer to the lower layer. The thin arrows in Figure 5.1 represent data flow in our framework.
The Session layer in our framework describes the coarse grain user view. A session consists of requests, but what types of requests and in what order is fully defined by the user navigation pattern, which provides the possible requests that can be made and the probabilities for making them. For example, after a user confirms a purchase, the next possible requests are to go back to the home page or to search for another product. The user navigation pattern is naturally represented by a DTMC in which each state represents a request made by the user; a traversal through the DTMC represents a Web session consisting of one or more requests. In our framework, the session duration in number of requests is captured by the user session model. Together, the user navigation pattern and the user session model, form the integral user operational profile, which fully describes the users’ behaviors.

While the user behavior is entirely described by the Integral user operational profile, for estimating the session reliability, it is necessary to know the reliability of individual requests within the session. For this purpose, we need a finer grain system view represented in the service layer, which uses the input information from the Requests’ reliability estimates block shown Figure 5.1
and passes the corresponding requests’ reliability estimates to the session model.

Each request is for a specific service, for example, it can be a request for a static html page (e.g., home page) in which case only the components of the front-end Web server are involved in servicing the request. On the other hand, a request for searching through the Web system, such as searching for a flight, in addition to a front-end static component involves components from the application and database servers of the three-tier Web system. Therefore, in order to estimate the reliability of each request, we need to know the components’ reliabilities and the way these components interact in serving each Web request, which is defined by the software architecture. This part of our framework builds on the theory and practice of the architecture-based software reliability (see, for example, [19] and references given therein).

It should be noted that the presented reliability modeling framework can be used to represent other paradigms that have workload defined in terms of sessions by taking into account the specifics of that domain when building the model within the framework.

5.2 Model of Web session reliability

In this section, we present the model for the session reliability of a multi-tier Web system based on the framework proposed in Figure 5.1.

5.2.1 Integral user operational profile

The user’s behavior is undoubtedly crucial for the observed reliability of any Web system, and, in our model, it is described through the integral user operational profile, which integrates the user navigation pattern through the system (described with a discrete time Markov chain model) and the user session model (represented by a probabilistic model of the number of requests in a session).

User navigation patterns model

An important aspect of Web users’ behavior is their navigation pattern through the Web system. This, intuitively, is modeled as a discrete time Markov chain (DTMC), where each state represents a different Web request. The user navigation pattern is defined by the structure of the DTMC chain and the transition probability matrix. The structure of the chain is a directed graph and it defines what are the possible requests that can be made by the users after the service of a given request has been completed. But, not all requests are made with the same probability and this is captured by
the transition probability matrix

\[ P = [p_{ij}] \]  

(5.1)

where the transition probability \( p_{ij} \) defines the probability that the user’s next request will be \( j \) given that the current request is \( i \).

Variants of the Customer Behavior Graphs (CBG) based on the TPC-W benchmark [58] (given in Figure 5.2) were widely used for generating Web workloads for performance modeling [44], [42] and measurement studies [12], as well as for availability modeling of Web systems [17]. However, the TPC-W benchmark was designed for testing Web systems performance in steady-state and it does not capture the heavy-tailed nature of the session length. In particular, the minimum duration, in minutes, of the session length in TPC-W is modeled with the exponential distribution with mean time of 15 minutes. Each session starts from Home and it ends after the minimum duration, following the exponential distribution, has expired and the next request is Home. Therefore, neither the TPC-W benchmark nor the papers which used its simplified variants for performance [44], [40], [42] and availability modeling [17] accounted for the heavy-tailed behavior of the session in Web systems shown to exist by the empirical work [39], [20].

![Figure 5.2: Example of user navigation pattern based on the TPC-W benchmark](image)

Our user navigation pattern model, as TPC-W, is defined by an irreducible DTMC, which
allows for general distribution for the number of requests that consists a Web session. In practice, the structure of the user navigation pattern and the transition probabilities can be extracted from the logs maintained at the Web server. For systems that are under development, the structure of the user navigation pattern can be extracted from the specification and/or design documents, while the transition probabilities can be assigned by an expert or adopted from benchmarks, such as the TPC-W.

We treat the number of request in a Web session as a random variable that can follow a general distribution. The model of the user sessions that specifies the probability distribution function of the number of request per session is presented next.

**User session model**

The empirical studies of Web workloads [39], [20] have shown that the number of requests per session follows a heavy-tailed distribution. This motivated us to include into our user session model the heavy-tailed behavior of the number of requests per session and to explore its impact on the session reliability. In practical terms, a random variable that follows a heavy-tailed distribution can give rise to extremely large values with nonnegligible probability. That is, in our case, Web Systems tend to observe sessions with a very large number of requests, although these sessions may be rather rare.

For modeling the number of requests per session, we use a mixture of two distributions, which gives us more flexibility when experimenting with the session reliability. The tail of the distribution often can be modeled well with one distribution (with CDF, $H(x)$ and pdf, $h(x)$), but that same distribution can diverge significantly at the body, which is better modeled with a different distribution (with CDF, $G(x)$, and pdf, $g(x)$) [30]. The mixing probability, $b$, gives the percent of points that fall into the body of the distribution. The point, $x_b$, where the transition from the body to the tail happens is related to $b$ by $G(x_b) = b$, or $x_b = G^{-1}(b)$.

The probability density function of a mixed distribution is given by the equation [53]:

$$f(x) = \begin{cases} \frac{b}{G(x_b)}g(x), & x \leq x_b \\ (1 - b)h(x - x_b), & x > x_b \end{cases}$$

and the corresponding cumulative distribution function (CDF) is given by:

$$F(x) = \begin{cases} \frac{b}{G(x_b)}G(x), & x \leq x_b \\ b + (1 - b)H(x - x_b), & x > x_b \end{cases}$$
Equation (5.2) and (5.3) can be used for mixing *any* two distributions. In our session model, for the body we use the lognormal distribution whose pdf is given by

\[
g_{\mu, \sigma}(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}}, \quad x > 0.
\]  

(5.4)

where \( \mu \) is the location parameter and \( \sigma \) is the scale parameter of the lognormal distribution.

We model the tail of the distribution with the generalized Pareto distribution based on the empirical results which showed that the number of requests per Web session has a heavy tail [39], [20]. The generalized Pareto distribution has the following pdf

\[
h_{\xi, \mu, \sigma}(x) = \frac{1}{\sigma} \left(1 + \frac{\xi(x - \mu)}{\sigma} \right)^{-\frac{1}{\xi} - 1}
\]

(5.5)

where \( \mu \) is the location parameter, \( \sigma \) is the scale parameter, and \( \xi \) is the shape parameter of the generalized Pareto distribution. The shape parameter for the generalized Pareto distribution is related to the index of the tail, \( \alpha \), by \( \alpha = 1/\xi \). Throughout this study, we use the index of the tail, \( \alpha \), which indicates whether the distribution is heavy-tailed \( \alpha < 2 \) or not \( \alpha \geq 2 \).

When using heavy tailed distributions for modeling the tail, \( b \) (i.e., \( x_b \)) can be estimated using Hill’s plot [25] and/or LLCD plots (see [21] and references therein).

The parameters of the distributions (5.4) and (5.5), for systems that have been deployed, can be estimated using the Web logs; while, for systems that are in development they can be specified by an expert, based on previous experience with similar systems or previous empirical studies.

### 5.2.2 Requests’ reliability estimate

To estimate the session reliability, the reliability of each request has to be estimated at the Service layer which provides the system view, and then integrated with the model build at the Session layer. A request can be quite complex, because it can use several components from multiple tiers (i.e., Web server, application server and/or database server) and components can be accessed multiple times depending on the underlying software architecture. In the proposed framework this is handled by the Requests’ reliability estimates block (see Figure 5.1) which takes into account the reliability of components involved in serving a particular request and the way these components interact defined by the software architecture.
Components’ reliabilities

Component can be defined in different ways and there is no generally accepted definition. In this paper, component is defined as a logically independent unit of the system which performs a well-defined function [19]. Depending on the focus of the study and the available information, the decomposition of the system on components can be done at different levels of granularity. In this work, we use the description provided in the TPC-W specification to come up with several components that execute the logic of an e-commerce system at the application server. Another example of system decomposition to components and the way they interact can be found in the measurement-based study of Web systems performance [12].

The components’ reliabilities can be estimated from the problem-report tracking systems or based on domain expert knowledge and historical data from similar systems.

Software architecture

Software architecture defines how different components interact and provides basis for estimating the requests’ reliabilities. In this paper, we consider a three-tier architecture with possible components based on the TPC-W specification, shown in the UML deployment diagram presented in Figure 5.3. The front-end Web server fully serves the static Web requests and forwards dynamic requests to the application server. The application server consists of components that implement the logic of serving the dynamic requests. In general, some components may be used by multiple requests, such as for example the Login component which is used by the Customer Registration Page, and Order Inquiry requests, or Promotions component which is used by Search Request, Search Results, Best Seller, and New Products requests. Components of the Application server interact with Database server(s), such as for example Search Logic component which queries the database for the specific search criteria.

Serving each request involves one or more components shown in Figure 5.3. The interaction of components in serving each request can be represented in different ways. For illustration, we use UML sequence diagram to show the interactions of components in serving the Buy Confirm request, as shown in Figure 5.4. It follows that the reliability of the Buy Confirm request, \( R_{Buy\,Confirm} \), can be calculated using the components’ reliabilities and the knowledge of their interactions as:

\[
R_{Buy\,Confirm} = R_{Buy\,Confirm\,Page} \times R_{Orders} \times R_{Payment\,Authorization} \times \left( R_{Database\,Server} \right)^2
\]

(5.6)
Figure 5.3: Deployment diagram of an e-commerce site modeled by TPC-W

For example, if we assume that $R_{Buy\ Confirm\ Page} = 1$, $R_{Orders} = 0.92$, $R_{Payment\ Authorization} = 0.925$, and $R_{Database\ Server} = 0.9995$, then the reliability of the request Buy Confirm is $R_{Buy\ Confirm} = 0.8501$. If the reliability of component Orders increases to 0.99 and the reliability of the component Payment Authorization increases to 0.96, then $R_{Buy\ Confirm} = 0.95$.

The software architecture can be obtained by instrumenting the Web system to trace client requests during the execution and track which components are used to satisfy each individual client request, as in our previous work on measurement of Web systems performance [12]. Alternatively, specification and architecture level design documents can be used to deduce the same information.

Figure 5.4: Sequence diagram for the Buy Confirm request
5.2.3 Model integration and solution

The integration of the model is done in a hierarchical fashion, in two steps. At the higher, coarse grained Session layer, a session is described by the navigation pattern which gives the probabilities of different requests appearing in the session, and by the session length in number of requests which follows a given probability distribution function. At the lower, fine grained Service layer, the reliability of each request (in our example \( R_i \), where \( i \in \{ \text{Home, Order Inquiry, Order Display, Best Seller, New Products, Search Request, Search Results, Product Detail, Shopping Cart, Customer Registration, Buy Request, Buy Confirm, Admin Confirm, Admin Request} \} \)) is estimated using the components’ reliabilities and the software architecture. The values of requests’ reliabilities are passed to the Session layer to be combined with the integral operational profile to produce an estimate of the session reliability.

We use simulation to solve the integration of models described in sections 5.2.1 and 5.2.2. A total of 100,000 sessions were simulated as follows. First, the number of requests in a session (i.e., the session length) is simulated based on the mixture of distributions defined by equation (5.2). Each session starts with Home request. Each following request \( i \) in the session is simulated based on the transition probability matrix, \( P = [p_{ij}] \), of the DTMC. Once the request \( i \) is simulated, a failure is generated with probability equal to \( 1 - R_i \). These two steps are repeated until the specific, previously generated random number of requests has been reached.

We implemented the simulation, whose pseudo-code is presented in Figure 5.5, in R language [50]. (For more details on simulation of random variables readers are referred to [52].)

A simulation approach always introduces variances in the results, and it is important to verify that the results are due to the varying of the variables and not due to the variance introduced by the simulation approach. As stated, in our simulation 100,000 sessions are simulated, which ensures good convergence of the simulations. In addition, the statistical methods that we use (described in the next section) verify that the observed results are due to change of the values and not due to variance from the simulation.

5.3 Experimental design and analysis

Once the model is built, it can help us answer many questions such as ‘What happens if some of the variables are changed?’, ‘Do variables interact?’, ‘What are the contributions of each variable and their interactions on the response variable?’ and so on.
failed_sessions ← 0 //number of failed sessions is zero at the beginning
for i = 1 to 100,000 do
    current_request ← HOME //all sessions begin with making a HOME request
    session_length ← generate number of requests in session
    failures ← 0 // the number of failed requests in the session at the beginning is zero
    // each request in the session is generated based on the transition probability matrix
    for j = 1 to session_length do
        probability_next_request ← generate_from_uniform_distribution(0, 1)
        cumulative_transition_probability ← 0
        j ← 0
        while probability_next_request ≤ cumulative_transition_probability do
            cumulative_transition_probability ← cumulative_transition_probability + P_{current_request,j} 
            j ← j + 1
        end while
        next_request ← j
        probability_failure ← generate_from_uniform_distribution(0, 1)
        // if the number generated is greater than the reliability of the request then count it as a failure
        if probability_failure > R_{next_request} then
            failures ← failures + 1
        end if
        current_request ← next_request
    end for
    if failures > 0 then
        failed_sessions ← failed_sessions + 1
    end if
end for

Figure 5.5: Pseudo-code for the simulation

In practice, we often deal with models that do not have a simple, close form analytical solution. A common approach to deal with complex models is to perform sensitivity analysis by the traditional ‘one-factor-at-a-time’ approach, where only one factor is varied, while holding all other
factors constant. However, this approach does not consider that the influence of one factor may depend on the value of another factor, that is, it does not allow the interaction among factors to be explored.

Instead of one-factor-at-a-time, in this thesis we use the design of experiments approach [47] which allows us to explore the effect of various factors and their interactions on the session reliability of Web systems in a systematic way. The fact that the effects of a factor can be estimated at several levels of the other factors allows for conclusions that are valid over a range of experimental conditions [47]. Even more, design and analysis of experiments allows us to quantify the contributions that each factor and their interactions have on the response variable (i.e., session reliability in our case). Last, but not least, it provides rigorous statistical testing of the significance of the results.

Factor in design of experiments is any independent variable that may influence the outcome of the experiment. Factor levels are the different values that a factor can have in the experiment. The individual effect that a factor has on the outcome of the experiment is called main effect, while the grouped effect of two or more factors is called interaction effect. If all possible combinations of factor levels are considered then the design of experiments is called full factorial; and if each treatment (combination of factor levels) is repeated equal number of times, then the design of experiment is called balanced. More details about design of experiment can be found in [47] and [38]. In this study, we use full factorial design because all interactions between factors are possible. Further, we use balanced design since it supports more accurate results for some analysis methods [38].

5.3.1 Choice of factors and factor levels

Choosing adequate factors and the corresponding factor levels are important points of designing an experiment. For factors we chose: the user navigation pattern defined by the transition probability matrix $P = [p_{ij}]$ of the DTMC (Section 5.2.1); the parameters $\mu$, $\alpha$, and $b$ of the session length distribution (Section 5.2.1); and the requests’ reliabilities, $R = [R_t]$ (Section 5.2.2). In the following discussion, we present our choices for factor levels and give a proper justification.

The user navigation pattern plays an important role in determining the reliability of the Web system. This is the reason for including it as a factor in the design of experiment. The levels for this factor were taken from the TPC-W specification [58], because it is based on real-world systems. Specifically, the two levels we use for the transition probability matrix $P = [p_{ij}]$ of the
DTMC shown in Figure 5.2 are the shopping profile and ordering profile specified by the matrices $P_{\text{shopping}}$ and $P_{\text{ordering}}$, given with (5.7) and (5.8), respectively. The ordering profile describes the behavior of users that are heavy buyers. Thus, in the ordering profile the probabilities for making requests such as Buy Confirm and Buy Request are higher compared to the corresponding probabilities in the shopping profile, which describes the navigation patterns of users who do some browsing and some ordering.

The ordering of the requests in the matrices (5.7) and (5.8) is \{Admin Confirm, Admin Request, Best Seller, Buy Confirm, Buy Request, Customer Registration, Home, New Products, Order Display, Order Inquiry, Product Detail, Search Request, Search Results, Shopping Cart\}.

The user session model is the second component of the Integral users operational profile. As described in Section 5.2.1, we model the number of requests in a session as a mixture of two distributions, the lognormal for the body and the Pareto for the tail. It is natural to ask the question how these distributions affect the reliability of the system? Or, to be more specific, how do the parameters ($\mu$, $\alpha$, and $b$) of the mixture distribution affect the session-based reliability?

$\mu$ defines the position of the body of the distribution (higher value means more sessions with

$$
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & .9953 & 0 & 0 & 0 & 0 & 0 & .0047 & 0 & 0 \\
.9000 & 0 & 0 & 0 & 0 & 0 & .1000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .0168 & 0 & 0 & 0 & .0305 & .9455 & 0 & 0.0072 \\
0 & 0 & 0 & 0 & 0 & 0 & .0085 & 0 & 0 & 0 & 0 & 0 & .9915 & 0 & 0 \\
0 & 0 & 0 & .4615 & 0 & 0 & .1932 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .3453 \\
0 & 0 & 0 & 0 & .8667 & 0 & .0094 & 0 & 0 & 0 & 0 & 0 & .1239 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .3125 & 0 & 0 & 0 & .3125 & 0 & .0469 & 0 & .0308 & .2973 \\
0 & 0 & 0 & 0 & 0 & 0 & .0157 & 0 & 0 & 0 & .9579 & .0049 & 0 & 0.0215 \\
0 & 0 & 0 & 0 & 0 & 0 & .0070 & 0 & 0 & 0 & 0 & 0 & .993 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .0073 & 0 & .8800 & 0 & 0 & 0 & .1127 & 0 & 0 \\
0 & .0059 & 0 & 0 & 0 & 0 & .0774 & 0 & 0 & 0 & .0456 & .7315 & 0 & .1396 \\
0 & 0 & 0 & 0 & 0 & 0 & .0636 & 0 & 0 & 0 & 0 & 0 & .8500 & .0864 \\
0 & 0 & 0 & 0 & 0 & 0 & .2658 & 0 & 0 & 0 & .6637 & .0010 & 0 & 0.0695 \\
0 & 0 & 0 & 0 & 0 & 0 & .2586 & .6967 & 0 & 0 & 0 & 0 & 0 & 0 & .0447 \\
\end{bmatrix}
$$

(5.7)
\[ P_{\text{Ordering}} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & .8349 & 0 & 0 & 0 & 0 & .1651 & 0 & 0 \\
.9000 & 0 & 0 & 0 & 0 & 0 & .1000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .0002 & 0 & 0 & 0 & 0 & .0332 & .9665 & 0 & .0001 \\
0 & 0 & 0 & 0 & 0 & 0 & .0003 & 0 & 0 & 0 & 0 & 0 & 0 & .9997 & 0 & 0 \\
0 & 0 & 0 & .8000 & 0 & 0 & .1454 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .0546 \\
0 & 0 & 0 & 0 & .9900 & 0 & .0002 & 0 & 0 & 0 & 0 & 0 & 0 & .0098 & 0 & 0 \\
0 & 0 & .0500 & 0 & 0 & 0 & .0500 & 0 & 0 & 0 & .0270 & 0 & .0026 & 0 & .8704 \\
0 & 0 & 0 & 0 & 0 & 0 & .0505 & 0 & 0 & 0 & .9438 & .0034 & 0 & .0023 \\
0 & 0 & 0 & 0 & 0 & 0 & .9940 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .1169 & 0 & .8800 & 0 & 0 & 0 & 0 & .0031 & 0 & 0 \\
0 & .0100 & 0 & 0 & 0 & 0 & .3651 & 0 & 0 & 0 & .1871 & .0720 & 0 & .3658 \\
0 & 0 & 0 & 0 & 0 & 0 & .0816 & 0 & 0 & 0 & 0 & 0 & 0 & .9000 & .0184 \\
0 & 0 & 0 & 0 & 0 & 0 & .0487 & 0 & 0 & 0 & .7331 & .2181 & 0 & .0001 \\
0 & 0 & 0 & 0 & 0 & 0 & .9500 & .0419 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .0081
\end{bmatrix} \]
higher number of requests). We used two levels for this factors. One level was estimated, using the maximum likelihood estimator, from the Lane Department of Computer Science and Electrical Engineering (LCSEE) Web server logs to be $1.0583$ and for the other level we chose a slightly higher value, $\mu = 2$, because we wanted to examine how longer sessions influence the session reliability.

$\alpha$ is the tail index of the Pareto distribution and it determines whether the tail is heavy or not. When $\alpha < 2$ the distribution has a heavy tail. First, we estimated the tail index from the LCSEE Web server to be $\alpha = 1.9929$, which is on the boundary between heavy-tailed and light-tailed distribution. This motivated us to experiment with two additional levels. One of them is $\alpha = 1.2$, which we chose because it represents a heavy tailed traffic. While, for the other level we chose $\alpha = 3$, which represents light-tailed traffic. These three values of $\alpha$ enabled us to examine how the session reliability changes over a wide range of values and, also, to avoid possible misconceptions about the effect that this parameter may have.

The mixing probability, $b$, determines the percent of points that fall into the body, i.e., $1 - b$ determines the percentage of points that fall into the tail. The estimated value for $b$ for the LCSEE data was $0.99611$, which means that a very small percent of sessions (less than 1%) belong to the tail. We used this value as one of the levels. But, we wanted to explore what would happen if the tail constitutes a larger percent, e.g., 10% or as big as 50%. That is why we also included the levels $b = 0.9$ and $b = 0.5$ for this factor.

And last but not least, the requests’ reliability, $R_i$, is another very important factor that determines the reliability of the whole Web system. We represent this factor as a single vector, $\mathbf{R} = [R_i]$, whose elements are the reliabilities of the individual requests, $R_i$. We included two levels for this factor, $R_{low}$ and $R_{high}$. The two levels differ only in the reliability of the component Buy Confirm, which is used heavily in the ordering profile, but rarely in the browsing profile. For the two levels $R_{low}$ and $R_{high}$, we assigned reliability of $0.999$ to each request, except for Buy Confirm which was assigned reliability of $0.85$ and $0.99$, respectively.

It follows that the resulting design of experiments is $\mu \times \alpha \times b \times P \times R = 2 \times 3 \times 3 \times 2 \times 2$ full factorial experiment, with total of 72 cells. We denote factors with the symbols used in the models: $\mu$, $\alpha$, $b$, $P$, and $R$. The subscripts, one for each factor (i.e., $i$, $j$, $k$, $l$, $m$) designate the specific factor-level combination from which the response is obtained. The last subscript $n = 1 \ldots, r$ designates repeat observations (i.e., replications) for fixed factor levels.

The number of replications for each combination of fixed factor levels (i.e., each cell) is important aspect of the design of experiments since it increases the statistical power of the tests and also
it enables us to determine how much variance is due to the experimental error. We replicated our experiment \( r = 100 \) times for each of the 72 cells of the full factorial design. (Note that each of the 100 replications in each of the 72 cells consists of 100,000 simulated sessions (see Figure 5.5).)

With this convention in notation, the responses (i.e., observed values of the session reliability) from the five-factor experiment with \( r \) (in our case 100) repeat simulations per factor-level combination would be denoted \( y_{ijklmn} \) with \( i = 1, 2, j = 1, 2, 3, k = 1, 2, 3, l = 1, 2, \) and \( m = 1, 2 \). A statistical model for the \( n \)-th observation from cell \((i, j, k, l, m)\) is:

\[
y_{ijklmn} = \theta_{ijklm} + e_{ijklmn} \tag{5.9}
\]

where \( \theta_{ijklm} \) represents the effect of the assignable causes and \( e_{ijklmn} \) represents the random error effects (i.e., residuals).

The assignable cause portion \( \theta_{ijklm} \) can be further decomposed into terms representing the main effects and the interactions among the five model factors\(^1\):

\[
\theta_{ijklm} = \theta + \beta_{\mu_i} + \beta_{\alpha_j} + \beta_{b_k} + \beta_{P_l} + \beta_{R_m} + \\
\beta_{\mu_i,\alpha_j} + \beta_{\mu_i,b_k} + \beta_{\mu_i,P_l} + \beta_{\mu_i,R_m} + \beta_{\alpha_j,b_k} + \\
\beta_{\alpha_j,P_l} + \beta_{\alpha_j,R_m} + \beta_{b_k,P_l} + \beta_{b_k,R_m} + \beta_{P_l,R_m} + \\
\beta_{\mu_i,\alpha_j,b_k} + \beta_{\mu_i,\alpha_j,P_l} + \beta_{\mu_i,\alpha_j,R_m} + \beta_{\mu_i,b_k,P_l} + \\
\beta_{\mu_i,b_k,R_m} + \beta_{\mu_i,P_l,R_m} + \beta_{\alpha_j,b_k,P_l} + \\
\beta_{\alpha_j,b_k,R_m} + \beta_{\alpha_j,P_l,R_m} + \beta_{b_k,P_l,R_m} + \\
\beta_{\mu_i,\alpha_j,b_k,P_l} + \beta_{\mu_i,\alpha_j,b_k,R_m} + \beta_{\mu_i,\alpha_j,P_l,R_m} + \\
\beta_{\mu_i,b_k,P_l,R_m} + \beta_{\alpha_j,b_k,P_l,R_m} + \\
+ \beta_{\mu_i,\alpha_j,b_k,P_l,R_m} \tag{5.10}
\]

Each of the subscripts \( \mu_i, \alpha_j, b_k, P_l, \) and \( R_m \) represents one level of the corresponding factor. The parameters in (5.10) having a single subscript represent main effects for the factor identified by the subscript. For example, \( \beta_{P_l} \) represents the main effect of the user navigation pattern on the session reliability. Two factor interactions are modeled by the terms having two subscripts, and the three factor interactions by the terms having three subscripts, and so on. The interaction components of (5.10) model joint effects that cannot be suitably accounted for by the main effects.

---

1. For this representation to be unique, constraints must be imposed on the values of the parameters. The commonly imposed constraints are: \( \sum \beta_{\mu_i} = 0, \sum \beta_{\alpha_j} = 0, \sum \beta_{\mu_i,\alpha_j} = 0, \sum \beta_{\mu_i,\alpha_j} = 0, \) etc.
Thus, a two factor interaction (e.g., $\beta_{P_i, R_m}$) is present in a model only if the effects of two factors on the response cannot be adequately modeled by the main effects only (i.e., $\beta_{P_i}$ and $\beta_{R_m}$).

### 5.3.2 Statistical analysis of the effects

The most popular method for formal statistical analysis of the design of experiments is the classical ANOVA $F$ statistics, which assumes that the errors are normally distributed with zero mean and equal variances [47]. If these assumptions are not met then the ANOVA $F$ statistics cannot be used.

One way of checking for normality is to plot the qq-plot with the normal distribution on the x-axis and the sample residuals\(^2\) on the y-axis. This plot is shown in Figure 5.6. From the figure, it can be noticed that the residuals have symmetric distribution (around zero) with longer tails than the normal distribution indicated by the bending at the edges. Thus, the assumption of normally distributed residuals for our data is not satisfied. However, ANOVA $F$ statistic is robust to slight deviations from normality. Therefore, we continue with examination of the residuals’ variances. For this, we plot the residuals versus the fitted values\(^3\). From the result in Figure 5.7, it is obvious that as the fitted data increases the variance decreases significantly, meaning that the sample data does not satisfy the assumption of equal variance. It is well known that ANOVA is very sensitive to violation of this assumption, which is the main reason for not using the classical ANOVA $F$ statistic for analysis of our results.

Instead, we have to resort to using non-parametric test for the effects. By doing a thorough search in the literature of nonparametric tests, we found that the best test is the recently developed, Brunner-Dette-Munk test [4]. In analysis of variance we test the hypotheses of no main effects, and no interaction effects. The nonparametric versions of the model (5.10) and hypotheses are in terms of distribution functions (i.e., $F_{ijklm}$) of the observations ($Y_{ijklmn}$) in each cell. Since the formulation of the nonparametric hypotheses requires an introduction of an elaborate notation, they are given in the Appendix.

The results of the hypotheses tests are given in Table 5.1. The first column is the test statistic, $Q_N$; the second column gives the degrees of freedom; and the third column is the p-value. The p-value is practically zero for all factors and their interactions, which means that the statistical hypotheses that the distributions are equal or, in other words, the hypotheses that there is no change

---

\(^2\)Sample residuals are defined as $e_{ijklmn} = y_{ijklmn} - \hat{y}_{ijklmn}$, where $\hat{y}_{ijklmn}$ is an estimate of the corresponding observation $y_{ijklmr}$.

\(^3\)The fitted value is an estimated value for an observation.
in the output when the factor/s is/are changed is rejected. Which leaves us with the alternative hypotheses that at least one of the factors or factor interaction causes statistically significant change in the session reliability.

However, the fact that the results are statistically significant does not mean that they are practically significant, because some factors and some interactions are more important than others. Which factors and which interactions are important can be determined by calculating the effect of each factor and each interaction. Due to space limitation we do not describe the method used for calculating the effects, but, we refer the interested reader to [13] for full description of the method. We implemented this method in R [50]. For the present discussion, it is important to note that the effects give the contributions that each factor or combination of factors have on the output variable. The main factors and their interactions as well as the effects are discussed in the next section.
5.4 Results

5.4.1 Discussion of the factors’ effects on session reliability

In this section we discuss the results of the design of experiments presented in Table 5.1, supported by graphical methods. The main effects are shown in Figure 5.8, which displays the main factors (i.e., $\mu$, $\alpha$, $b$, $P$, and $R$) and for each factor its levels (e.g., for $\mu$ there are two levels, 1.0583 and 2) on x-axes and the values of the session reliability on y-axes.

The factors that we take into consideration from the users session model are the parameters $\mu$, $\alpha$, and $b$ of the mixture of lognormal and Pareto distributions. Next, we discuss our findings about the impact that these factors have on session reliability.

From Figure 5.8, it can be noticed that $\mu$ (the location parameter of the lognormal distribution for the body of the distribution of the number of request in a session) has significant impact on the session reliability, i.e., higher $\mu$ results in lower reliability and vice versa. This is further corroborated by the value for $\mu$’s main effect in Table 5.1. This happens because higher $\mu$ means higher chance for longer sessions (i.e., sessions with larger number of requests) and, of course, the
probability that at least one request in the session will fail increases with the length of the session.

Table 5.1: Test Statistics for Session Reliability

<table>
<thead>
<tr>
<th></th>
<th>$Q_N$</th>
<th>df</th>
<th>$p_Q$</th>
<th>Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>one-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>487699.87</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0252</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>2471.36</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0004</td>
</tr>
<tr>
<td>$b$</td>
<td>16613.63</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0004</td>
</tr>
<tr>
<td>$P$</td>
<td>383784.05</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0471</td>
</tr>
<tr>
<td>$R$</td>
<td>705086.15</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0559</td>
</tr>
<tr>
<td><strong>two-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha$</td>
<td>1370.41</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b$</td>
<td>764.86</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu P$</td>
<td>17826.34</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0087</td>
</tr>
<tr>
<td>$\mu R$</td>
<td>14125.27</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0110</td>
</tr>
<tr>
<td>$\alpha b$</td>
<td>16528.69</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0005</td>
</tr>
<tr>
<td>$\alpha P$</td>
<td>8.65</td>
<td>2.00</td>
<td>0.01</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha R$</td>
<td>1679.26</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0001</td>
</tr>
<tr>
<td>$b P$</td>
<td>17.10</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b R$</td>
<td>866.22</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0001</td>
</tr>
<tr>
<td>$P R$</td>
<td>77284.10</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0349</td>
</tr>
<tr>
<td><strong>three-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b$</td>
<td>1246.33</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha P$</td>
<td>26.53</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha R$</td>
<td>54.48</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b P$</td>
<td>128.94</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b R$</td>
<td>13.00</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu P R$</td>
<td>24478.84</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0063</td>
</tr>
<tr>
<td>$\alpha b P$</td>
<td>67.79</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b R$</td>
<td>1289.61</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0001</td>
</tr>
<tr>
<td>$\alpha P R$</td>
<td>1831.76</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b P R$</td>
<td>1001.28</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td><strong>four-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b P$</td>
<td>1113.46</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha b R$</td>
<td>78.37</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha P R$</td>
<td>602.73</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b P R$</td>
<td>324.05</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b P R$</td>
<td>2005.31</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td><strong>five-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b P R$</td>
<td>237.09</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

The index of the tail, $\alpha$, has a little influence on the session reliability. For very heavy tail, $\alpha = 1.2$, the session reliability is lower compared to $\alpha = 1.9929$. This small change can be
attributed to the fact that longer sessions are more likely to encounter at least one request failure. When $\alpha$ changes from 1.9929 to 3 there is not very significant increase in the reliability, because $\alpha = 1.9929$ is on the border of heavy tailness, so when $\alpha$ changes to 3 the number of very long sessions is not decreased drastically. The value for the effect of $\alpha$ in Table 5.1 is 0.0004 which means that the index of the tail does not seem to be very relevant.

The mixing probability, $b$, similarly to the index of the tail $\alpha$, does not affect much values of the session reliability. This, on first sight, somewhat surprising fact can be attributed to the choice of the distribution for the body and the distribution for the tail. That is, the lognormal distribution is a skewed distribution with a long tail, so making the tail heavier by decreasing $b$ and thus including more points from the Pareto distribution, does not make a significant difference. Maybe, if the distribution for the body is not skewed $b$ and $\alpha$ would make greater differences. (It should be noted though that we chose the mix of the lognormal and Pareto distributions for the number of requests in a session based on the data collected from a real Web server which clearly indicated that the distribution of the body of the number of requests is skewed.)

As expected, the user navigation pattern $P = [p_{ij}]$ has a significant effect on the session reliability. The value for its main effect in Table 5.1 is 0.0471. In our study, the ordering profile $P_{ordering}$ leads to lower reliability than the shopping profile $P_{shopping}$, because the reliability vector, $R = [R_i]$, has lower reliability for the Buy Confirm request, which is a request much more frequently used in the ordering profile.
The reliability of the requests, \( R = [R_i] \), has a strong influence on session reliability, as it can be seen in Figure 5.8 and from Table 5.1. This certainly is an expected result - sessions consist of requests and the less reliable they are the less reliable the session is.

Having explored the main effects of each of the factors on the session reliability, we next address the interaction effects, which are those combinatorial effects that two or more factors have on the response variable. This is particularly important since such two and higher way factor interactions are not apparent when using the traditional one-factor-at-a-time approach.

The two-way factor interaction effects plots shown in Figure 5.9 visualize the session reliability changes that result from the combined varying of two factors. In this plot levels of each factor are displayed on the x-axis, and for each level of the other factor (shown on the right hand side of the y-axes) a separate line that represents the means is drawn. If these lines are parallel then there is no interaction between the two factors, whereas non-parallel lines suggest the presence of two-way factor interactions. The values of the response variable (i.e., session reliability) are shown on the left hand side of y-axes.

As it can be observed from Figure 5.9 and Table 5.1 only the following three combinations of two factors interact: (1) \( \mu \) and profile \( P = [p_{ij}] \); (2) \( \mu \) and requests’ reliability \( R = [R_i] \); and (3) profile \( P = [p_{ij}] \) and requests’ reliability \( R = [R_i] \).

When the level of \( \mu \) is varied from 1.0583 to 2, it can be noticed that the session reliability is drastically lower for the reliability level \( R_{\text{low}} \) (which has 0.85 reliability of the Buy Confirm request) than for \( R_{\text{high}} \) (which has 0.99 reliability of the Buy Confirm request). This can be explained as follows. When \( \mu = 2 \) the number of request in the sessions from the body of the distribution increases, and for the \( R_{\text{low}} \) reliability level more users will get to the less reliable Buy Confirm request.

The discussion on the interaction of \( \mu \) and user navigation pattern \( P = [P_{ij}] \) is similar. When \( \mu = 2 \) the number of request in the sessions from the body of the distribution increases, and, for the ordering profile \( P_{\text{ordering}} \), that means that more users will get to the Buy Confirm request which has low reliability. The session reliability under the shopping profile \( P_{\text{shopping}} \) is not affected that significantly by the higher value of \( \mu \) because Buy Confirm has a low probability of being accessed in this profile.

The interaction of profile \( P = [p_{ij}] \) and requests’ reliability \( R = [R_i] \) is expected, because the profile defines the transition probabilities between different Web requests. The ordering profile, \( P_{\text{ordering}} \), has a greater probability, compared to the shopping profile, \( P_{\text{shopping}} \), of making the faulty request Buy Confirm. And, thus, the reliability depends both on the reliability of the Web
requests and on the users’ navigation pattern. This is also substantiated by the value for this interaction effect 0.0349 (see Table 5.1), which is higher than some of the main effects (e.g. $\mu$).

Besides the main effects and two-way interactions, our results show that one of the three-way interactions, also, has significant effect on the session reliability. This is the interaction of $\mu$, $P = [p_{ij}]$, and $R = [R_i]$. The effect for this interaction is 0.0063, which is quite low compared to the main effects, but it still makes a noticeable change in the session reliability. For example, longer sessions (i.e., higher $\mu$), ordering profile $P_{ordering}$, and low reliability $R_{low}$ contribute to lower session reliability.

Figure 5.9: Interaction effect plot for session reliability
5.4.2 Comparison with request-based reliability

Request-based reliability is another metric of Web systems reliability, which is defined as the number of non-failed requests over the total number of requests [57], [18]. Obviously, request-based reliability does not take into account how the failed requests are distributed within user sessions. In this section we analyze the request-based reliability and compare and contrast it to the session reliability.

The results of the hypothesis tests and the main and interaction effects on the request-based reliability are presented in Table 5.2. As in case of session reliability, the p-value is practically zero for all factors and their interactions, which means that the statistical hypothesis that the distributions are equal or, in other words, the hypothesis that there is no change in the request-based reliability when the factors and their interactions are changed are rejected. However, there is a difference in the contributions of factors on the request-based reliability compared to session reliability. Thus, it is obvious from Table 5.2 that only the main factors requests’ reliabilities $R = [R_i]$ and navigation patterns $P = [p_{ij}]$ and the two-way interaction between them have significant effect on the request-based reliability. The same can be observed from Figures 5.10 and 5.11.

![Figure 5.10: Main effect plot for request-based reliability](image)

The reasons behind the significant effect of the requests’ reliability $R = [R_i]$ and user navigation patterns $P = [p_{ij}]$, and their interaction on the request-based reliability are very similar as in case of session reliability.

Compared to the session reliability $\mu$ has markedly lower influence on the request-based reliability. This happens because, whether the sessions are longer or shorter does not change the
Table 5.2: Test Statistics for Request Reliability

<table>
<thead>
<tr>
<th></th>
<th>$Q_N$</th>
<th>df</th>
<th>$p_Q$</th>
<th>Effects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>one-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>3484.65</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>427.42</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b$</td>
<td>217.62</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$P$</td>
<td>50421.17</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0008</td>
</tr>
<tr>
<td>$R$</td>
<td>201614.49</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0009</td>
</tr>
<tr>
<td><strong>two-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha$</td>
<td>30.81</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b$</td>
<td>10.77</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu P$</td>
<td>1265.39</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu R$</td>
<td>2044.64</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b$</td>
<td>168.03</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha P$</td>
<td>8.69</td>
<td>2.00</td>
<td>0.01</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha R$</td>
<td>541.00</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b P$</td>
<td>2.83</td>
<td>2.00</td>
<td>0.24</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b R$</td>
<td>245.25</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$P R$</td>
<td>0.00</td>
<td>1.00</td>
<td>1.00</td>
<td>0.0006</td>
</tr>
<tr>
<td><strong>three-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b$</td>
<td>21.54</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha P$</td>
<td>5.20</td>
<td>2.00</td>
<td>0.07</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha R$</td>
<td>5.11</td>
<td>2.00</td>
<td>0.08</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b P$</td>
<td>1.06</td>
<td>2.00</td>
<td>0.59</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b R$</td>
<td>6.30</td>
<td>2.00</td>
<td>0.04</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu P R$</td>
<td>774.53</td>
<td>1.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b P$</td>
<td>2.83</td>
<td>4.00</td>
<td>0.59</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b R$</td>
<td>187.64</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha P R$</td>
<td>91.79</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$b P R$</td>
<td>30.44</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td><strong>four-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b P$</td>
<td>6.86</td>
<td>4.00</td>
<td>0.14</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha b R$</td>
<td>3.61</td>
<td>4.00</td>
<td>0.46</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu \alpha P R$</td>
<td>22.16</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\mu b P R$</td>
<td>18.92</td>
<td>2.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td>$\alpha b P R$</td>
<td>74.82</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
<tr>
<td><strong>five-way interactions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu \alpha b P R$</td>
<td>30.37</td>
<td>4.00</td>
<td>0.00</td>
<td>0.0000</td>
</tr>
</tbody>
</table>
Figure 5.11: Interaction effect plot for request-based reliability

number of failed requests significantly. Slightly lower request-based reliability for $\mu = 2$ is due to the fact that longer session in the body of the distribution typically have higher chance to reach the low reliability Buy confirm request, and thus result in some more failed requests.

It is important to emphasize that the magnitudes of the changes in the request-based reliability due to the main and interaction effects are much smaller compared to the session-based reliability. Thus, request-based reliability varies in the range from 0.9839 to 0.9990, while session reliability ranges from 0.8032 to 0.9989. (See the main effects plots in Figures 5.10 and 5.8, and the two-way interaction plots in Figures 5.11 and 5.9.) This observation clearly illustrates that the request-based reliability provides an optimistic estimate and does not truly represent the users’ view on Web systems reliability. Furthermore, it supports the argument previously made based on the initial empirical analysis [18], [21] - the way failed requests are distributed within Web sessions
provides better indication of the user’s view on a Web server quality than the mere number of failed requests.
Chapter 6

Conclusion

6.1 Performance

In this study we use a feedback queue to account for session-based workload and study the impact of its characteristics (i.e., the session arrival process and the number of request per session) on performance metrics of Web systems. In particular, we first assume a point process for the session arrivals, and then another model for intra-session characteristics (in this case a distribution for the number of requests within a session). By considering LRD session arrivals (instead of Poisson), skewed distribution for the number of request within a session (instead of geometrical distribution), and a finite queue size (instead of infinite) our work generalizes the existing work on feedback queues and prior simulation study which considered session workloads.

We explore several performance metrics: percentage of dropped sessions, average queue length, waiting time, and useful server utilization. In addition, we pay particular attention on the nature of the request arrival and request departure point processes, which has not been done in the related work.

To summarize, our results show that the LRD of the session arrival process, especially for higher values of the Hurst exponent, has bigger impact on all performance metrics (i.e., percentage of dropped sessions, mean queue length, mean waiting time, and useful utilization) when compared to the statistics of the number of requests per session. Performance metrics for lower values of the Hurst exponent (e.g., $H_{\text{session}} = 0.6$) are close to the values for Poisson session arrivals.

More interestingly, for a higher variation of the number of request per session the percentage of dropped sessions is smaller, as well as the mean queue length and mean waiting time. This is due to the fact that longer sessions have greater chance to be dropped, which will affect less
users. However, this results in lower useful utilization of the server, which has been busy serving requests of the dropped sessions. Furthermore, it is likely to lead to loss of revenue in the case of e-commerce sites which typically observe more purchasing requests in the longer sessions.

Another interesting observation is the fact that both request arrival and departure processes are LRD, even when the session arrivals are Poisson and the think time and service time are exponential. This indicates that the LRD of the request arrival and departure process is due to the presence of session. This is a very significant contribution to workload modeling since it shows that LRD can occur even when the service and think time are not heavy tailed. Note, however, that when session arrivals are Poisson, performance metrics are affected only at very high utilization and the impact is less significant.

Our results have several strong implications for performance modeling. It is obvious that in order to build a realistic model for systems with session-based workloads, both inter-session characteristics (e.g., LRD of session count and inter-arrivals) and intra-session characteristics (e.g., distribution of the number of request per session) have to be modeled accurately. The combination of feedback queue with finite queue size under LRD session arrivals and skewed distribution of the number of requests in a session, provides a model with physically meaningful parameters, which preserves both the session and the request characteristics. Furthermore, we show that although the session arrival model is not affecting significantly the LRD of the request arrival and request departure processes, it has to be taken into account because the LRD at session level determines the queueing delays and session losses, and thereby the quality of delivered services.

### 6.2 Reliability

Session reliability is one of the most important quality attributes of Web systems because it shows both how the users perceive the reliability of the Web system and how the revenue of the owner may be affected by failed requests. However, there is a lack of research in this area. The contributions of this thesis include providing a general framework for assessing session reliability; proposing a detailed model of a Web system session reliability; and using a design and analysis of experiments as a better alternative to sensitivity analysis based on changing one factor at a time.

The framework integrates the users’ behavior represented by the Session layer with the system behavior in serving Web requests represented by the Service layer. This framework is general enough to be used for different type of Web systems or even other systems (such as Software as a Service) that have sessions as an intrinsic characteristic. To the best of our knowledge, this is the
first framework which allows for modeling session reliability and incorporates workload related data (e.g., session length in number of requests). The specific model within this framework is based on realistic assumptions which, whenever possible, were based on the measurements from real Web systems. Taking the design of experiments approach, which is not a common practice in reliability studies of software and computer systems, we were able to conduct a systematic study of the effects different factors and their interactions may have on the session reliability of Web systems.

Our results showed that three main factors (out of five), three two-way interactions (out of ten) and only one three-way interaction (out of 10) have significant effect on the session reliability. It is important to note that some of the two-way interactions had higher effect than some main factors and are necessary to consider for a full understanding of the session reliability. The results of the design and analysis of experiments for the request-based reliability showed that it provides an optimistic estimates of the Web quality. This observation supports the argument that the session reliability, which accounts for the distribution of failed requests within Web users sessions (rather than treating all requests’ failures independently and equally), is a better representation of the user perceived quality of the Web systems.
Appendix A

Appendix

In this section we formulate the hypotheses about the main and interaction effects on the session reliability. Before giving more details about the hypothesis, we introduce the concept of contrast matrix. A matrix $C_{r \times d}$ is called a contrast matrix if $C_{r \times d}1_d = 0_{r \times 1}$ where $1_d = (1, \ldots, 1)^t$ is a $d$-dimensional vector of 1’s. The contrast matrix used in the Brunner-Dette-Munk test has the form:

$$Q_d = I_d - \frac{1}{d} J_d$$

(A.1)

where $I_d$ is a $d$ - dimensional unit matrix, and $J_d = 1_d 1_d^t$ is a $d \times d$ matrix of 1’s.

Next, the factors in our experiment are $\mu$, $\alpha$, $b$, $P$, $R$, and the levels for each factor are $\mu \in \{1.0583, 2\}$, $\alpha \in \{1.2, 1.9929, 3\}$, $b \in \{0.5, 0.9, 0.99611\}$, $P \in \{P_{\text{Shopping}}, P_{\text{Ordering}}\}$, and $R \in \{R_{\text{low}}, R_{\text{high}}\}$, respectively. In addition the number of replications per cell $(i, j, k, l, m)$ is denoted by $r = (1, \ldots, n_{ijklm})$; in our case, we use balanced design, so the number of replications per cell is the same for each cell and is equal to 100, $r = n = 100$.

The observations $X_{ijklm}$ in each cell are considered to be independent random variables with distribution functions $F_{ijklm} = \frac{1}{2}[F_{ijklm}^+ + F_{ijklm}^-]$. We denote the vector of the distribution functions by

$$F = (F_{\mu_1, \alpha_1, b_1, P_1, R_1}, \ldots, F_{\mu_1, \alpha_1, b_1, P_1, R_1}, \ldots, F_{\mu_2, \alpha_3, b_3, P_2, R_2})^t$$

(A.2)

Now, the simplest hypotheses that there is no main effect for the factors $\mu$, $\alpha$, $b$, $P$, and $R$, are expressed as
where \( \overline{F}_{\mu_i,...} \), is the mean over all \( 3 \times 3 \times 2 \times 2 = 36 \) distribution functions within level \( \mu_i \) of factor \( \mu \), etc. These hypotheses formally can be written as

\[
H_0^F (\mu) : \quad (Q_2 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_2 \otimes \frac{1}{2} 1^i_2) F = C_\mu F = 0 \tag{A.8}
\]

\[
H_0^F (\alpha) : \quad (-\frac{1}{2} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_2 \otimes \frac{1}{3} 1^i_3) F = C_\alpha F = 0 \tag{A.9}
\]

\[
H_0^F (b) : \quad (-\frac{1}{2} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_3 \otimes \frac{1}{2} 1^i_2) F = C_p F = 0 \tag{A.10}
\]

\[
H_0^F (P) : \quad (-\frac{1}{2} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_2) F = C_{P_{\text{profile}}} F = 0 \tag{A.11}
\]

\[
H_0^F (R) : \quad (-\frac{1}{2} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_2 \otimes Q_2) F = C_R F = 0 \tag{A.12}
\]

The hypothesis that there is no interaction between \( \mu \) and \( \alpha \) is expressed as

\[
H_0^F (\mu \alpha) : \quad \overline{F}_{\mu \alpha_1} + \overline{F}_{\mu \alpha_2} + \overline{F}_{\mu \alpha_3} = \overline{F}_{\mu_1} + \overline{F}_{\mu_2} + \overline{F}_{\mu_3}, \quad \mu_i = \mu_1, \mu_2, \mu_3, \quad \alpha_j = \alpha_1, \alpha_2, \alpha_3. \tag{A.13}
\]

The matrix notation of this hypothesis is

\[
H_0^F (\mu \alpha) : \quad (Q_2 \otimes Q_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{3} 1^i_3 \otimes \frac{1}{2} 1^i_2 \otimes \frac{1}{3} 1^i_3) F = C_{\mu \alpha} F = 0 \tag{A.14}
\]

Hypotheses for the other interactions can be defined in the same way.
References


