# Simulation of the response time distribution of fault-tolerant multi-tier cloud services

Anders N. Gullhav*[1], Bjørn Nygreen†[1], and Poul E. Heegaard‡[2]

[1]Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway
[2]Department of Telematics, NTNU, NO-7491 Trondheim, Norway

**Abstract**

We are considering the problem of obtaining the response time distribution of fault-tolerant multi-tier services. In the provision of software-as-a-service applications, the service provider is obliged to ensure a certain quality of service. Herein, we regard upper bounds on the response time. The services consist of multiple components with different functionality, which are prone to failures, and fail according to a certain failure time distribution. However, due to redundancy, a failure will not necessarily bring the service down, but rather increase the response time. A fundamental difficulty with estimating the response time distribution while considering failures is related to the disparity in the time scales of the time between failures and service times. To overcome this issue, we propose an approach based on a decomposition, which combines an analytic model of the failure process and a discrete event simulation model to sample the response time distribution. In an experimental study, we compare this simulation-based approach with an analytic approach, and illustrate how this approach can be utilised by service providers as decision support. We also show that in certain cases, the analytic approach might provide a safe bound on the response time.

***Keywords***— Simulation; Queueing; Markov processes; Cloud computing; Replication

## 1 Introduction

In the provision of cloud software services, the providers are often offering guarantees on the performance and the dependability of the services. Such guarantees are written in service level agreements (SLAs), which are contracts that include definitions of the services to be delivered and their quality of service (QoS) levels (Wu and Buyya, 2012). The QoS can be measured by different metrics, and the response time, throughput and availability are among the most common. If upper bounds on the response time are given, these bounds typically regard the mean response time. However, for some end-users, bounds on the higher percentiles of the response time distribution might be of more interest. Software-as-a-service (SaaS) applications; e.g., email systems, enterprise resource planning systems and other business applications; typically have a multi-tiered architecture. These multi-tier services are composed of several components that collaborate in the service delivery, and a typical web service provided through the SaaS model consists of a web server component, an application logic component, and a database component. Because of the interaction between the tiers, obtaining the response time distribution is typically more difficult for multi-tier services than for single-tier services.

*anders.gullhav@iot.ntnu.no
†bjorn.nygreen@iot.ntnu.no
‡poul.heegaard@item.ntnu.no

Marston et al. (2011) identify the lack of QoS and availability guarantees as one of the major weaknesses of running mission-critical business applications as SaaS applications in clouds. In this paper, we assume that the components of the services might fail, but also assume that fault tolerance through redundancy might be implemented so that a failure does not necessarily bring the service down. Nevertheless, a failure might result in increased response times. Therefore, we are interested in obtaining a response time distribution that accounts for failures. This response time measure can be seen as a unified measure of performance and dependability (or reliability), often referred to as performability (Meyer, 1980; Al-Kuwaiti et al., 2009).

The aim of the work presented in this paper is to provide a simulation approach that can be used to extract the response time distribution of a multi-tier service, where the virtual machines (VMs) that run the different service components fail according to a given process. By this response time distribution, one could for a specific service configuration in terms of resource and redundancy allocation, assess the risk of not delivering according to the guarantees given in the SLA. This information can be used in cooperation with optimisation models and algorithms, such as the ones by Gullhav and Nygreen (2015, 2016), that are managing the placement and resource assignment of all services provided by the service provider.

Under certain conditions, it would be possible to compute the response time distribution of a service analytically. Gullhav et al. (2013) propose an analytic approach that assumes exponential inter-arrival and service time distributions of the requests, and based on this, produces a response time distribution that accounts for failures in the VMs running the service components. Typically, obtaining a response time distribution using an analytic approach is faster than using simulation and, hence, as long as the distribution produced by the analytic approach is valid, an analytic approach is favourable. However, if one cannot make the simplifying assumptions often required by analytic models, the computed response time is likely to be inaccurate, and a simulation might give better results.

There already exist descriptions of analytic models and simulation models that concern the response time of multi-tiered services in the literature. Urgaonkar et al. (2005) present an analytic model of a multi-tier web service, where the service is represented by a closed queueing network, and each tier might be replicated into a number of parallel load-balanced servers. By using mean-value analysis, they derive the mean response time of the service. Xiong and Perros (2009) model a cloud service by an open queueing network of two M/M/1 queues in series with feedback. Instead of focusing on the mean response time, they concentrate on obtaining approximate values for the higher percentiles, e.g., the 90th and 95th. Grottke et al. (2011) give a method for approximating the response time distribution of different types of queueing networks with phase-type time distributions. A discrete event simulation modelling tool, which has the aim to support service providers in the performance management of their services, is proposed by Alam et al. (2012). A more extensive tool, CloudSim (Calheiros et al., 2011), supports modelling and simulation of cloud system components such as data centres, VMs and networks. However, none of these tools allow one to model and simulate failures. The fundamental difficulty of estimating response times by a simulation model that considers the failure processes in addition to the arrival and services processes is related to the disparity in time scales, where the time between failures is in the range of hours to days and service times are in the order of milliseconds. The disparity of the time-scales does not only cause problems in simulation models, but could also lead to numerical issues if using analytic modelling tools, such as continuous time Markov chains (CTMCs) (Trivedi et al., 1993).

The contribution of this paper is a simulation approach that is more flexible with respect to the underlying assumptions, compared to the analytic approach presented in Gullhav et al. (2013). A novelty of our work is that we include the occurrence of service failures in a simulation approach where the output is an estimated response time distribution. This is achieved by decomposing the overall problem into two subproblems: one that concerns failures, and one that concerns obtaining response time estimates. The latter is tackled by a simulation model that uses stratified sampling when sampling the response times of the requests.

In the next section, we give an overview of the problem under study. Then, we describe the decomposition-based simulation approach. Thereafter, we present an experimental study, where we compare the response time distributions obtained by the simulation approach and the analytic approach of Gullhav et al. (2013). Moreover, we also demonstrate how our approach can be of use for SaaS providers (SPs) in the management of their service provisioning. Lastly, some concluding remarks are given.

## 2 Problem description

The aim is to estimate the response time distribution of a multi-tier service while considering failures. A multi-tier service is in this work assumed to consist of a set $\mathcal{Q} = \{1, 2, \dots, Q\}$ of tiers, also denoted *components*. In the cloud context modelled herein, these components will run in separate VMs, which in turn are run on the physical machines (PMs) in a cloud data centre. We will use a three-tier web service, composed of a web server component (tier 1), an application logic component (tier 2), and a database component (tier 3), as a case.

By horizontal scaling, each component $q \in \mathcal{Q}$ might be realised by more than one VM to be able to serve demand with a tolerable response time. As an example, one might have several web servers that serve the client's requests in parallel. In this, we assume that a load-balancer will distribute each request to a single web server, and thus, the processing of a single request is not parallelised. There exist several schemes on how to distribute the requests among the VMs, such as random scheduling, round robin scheduling, or according to the current load of the VMs. In addition, if the VMs are placed at different geographical locations, it might be beneficial to take the location of the VMs into account. Herein, we assume a random scheduling of the requests among the VMs of a tier. In all, we assume that each request is processed once in each tier, and after being processed in the last tier, the response is sent to the end-user.

Moreover, we consider that the VMs of the service might fail, but we restrict ourselves to assume independent failures of the VMs. To improve the dependability and fault-tolerance of a service, it is possible to replicate its components by running backup VMs on PMs that are ready to take over service delivery in case a failure brings down one of the other VMs of the component. We denote the load-balanced VMs that serve demand in a failure-free situation as active replicas, and the backup VMs as passive replicas. When a tier operates correctly, a request from the end-user is transmitted to one of the active replicas of that tier and the selected active replica processes the request. At regular time intervals, the passive replicas are updated, so that they are ready to take over service delivery in case of a failure. For databases, the replicas typically use a group communication protocol to pass transactional updates. While for other statefull servers, the updates consist of state updates. We assume that the time from the failure to the passive replica is activated and ready to serve demand is stochastic, and this delay is denoted the failover delay. Thus, when a failure occurs in one of the active replicas, a passive replica is made active and starts to serve the end-users after a failover delay, and thereafter, the failed replica is being repaired. As a simplification, we assume perfect fault coverage, i.e., all failovers are successful. An example of a multi-tier service with both active and passive replicas is illustrated in Figure 1. The first tier is realised with two active replicas and one passive replica, the second tier is realised with three active replicas and two passive replicas, and the last tier is realised with one active replica and one passive replicas. In a failure-free situation, the requests follow the solid arrows and are serviced by the replicas labelled as active replicas. When a failure occurs, a passive replica is activated and some (or all) requests are instead routed along a dashed arc.

It could be very time-consuming to obtain a good estimate of the response time by simulation of a model that considers both the failure and repair processes and the arrival and service processes. In order to get a good estimate of the response time, one needs to obtain samples

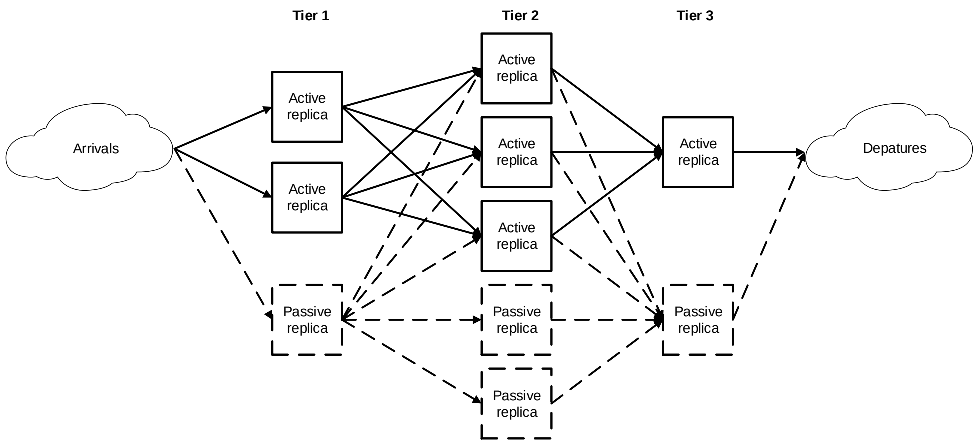

**Figure 1:** Illustration of a multi-tier service with active and passive replicas of its tiers

from the different failure scenarios where one or more replicas have failed. While the failures and repairs occur on a time-scale of hours, the inter-arrival and service times of the requests are in the range of milliseconds to a second. Therefore, one might have to perform very long simulations to obtain enough response time samples from the most infrequent failure scenarios.

## 3 Decomposition-based simulation approach

In this section, we describe the simulation approach that aims to estimate the response time distribution of fault-tolerant multi-tier services. Before we discuss the details of the approach, we present the general idea of our approach, which is based on decomposition.

### 3.1 Decomposition technique

Our approach to the problem of obtaining estimates of the response time distribution when considering failures is based on decomposing the problem into a dependability analysis subproblem and a performance analysis subproblem that are analysed independently. In contrast to the overall problem, these subproblems do not have the time scale issue as discussed above, which can simplify the analysis. Here, the dependability analysis subproblem consists of identifying and analysing failure scenarios, while the performance analysis subproblem consists of obtaining response time estimates of each failure scenario. In principle, both subproblems can be analysed by the use of measurements, simulation and analytic models. The most suitable approach for a given case is specific to the case and based on its characteristics (Jain, 1990). In this work, we investigate the performance analysis subproblem by a simulation model, referred to as the *performance submodel*, which is compared to the analytic approach by Gullhav et al. (2013). Moreover, we assume that the dependability analysis subproblem has some nice properties so that it could be tackled by an analytic model, referred to as the *dependability submodel*. While this submodel makes some simplifying assumption about the properties of cloud services, the use of an analytic model nicely illustrates the independence of the submodels of the decomposition technique. The decomposition technique and the approaches to analyse the subproblems are illustrated in Figure 2.

In simulation, a way to overcome the problem of two very distinct time scales is to decompose and use the variance reduction technique *stratified sampling* (Lewis and Orav, 1989). In this technique, the outcomes of the simulation are partitioned into different strata, with a known probability of belonging to a stratum. Then, a number of samples is obtained from each stratum, according to a *sampling scheme*, and the estimate of interest is obtained by weighting the estimates of the single strata using the probabilities of belonging to the stratum. We design our

| | | Dependability analysis subproblem | | |
|---|---|---|---|---|
| | | Measurements | Simulation | Analytic model |
| Performance analysis subproblem | Measurements | | | |
| | Simulation | | | This work |
| | Analytic model | | | Gullhav et al. (2013) |

**Figure 2:** Approaches to the dependability analysis and performance analysis subproblems

simulation approach with the different failure scenarios as strata, and obtain separate response time distributions of the strata. By this decomposition, we can easily obtain many response time samples from the failure scenarios with low probability, but possibly high response time. The sampling scheme is discussed in detail in the experimental study.

We only simulate failure scenarios with stables queues, i.e., scenarios where the total service rate of each tier is greater than the arrival rate. This means that the overall response time distribution, obtained by weighting the response time distributions of the strata with the failure scenario probabilities, is the response time distribution of the requests arriving to stable queues. Hence, we distinguish between requests that arrive to stable queues and request that arrive to unstable queues, and in this approach, the latter are not simulated, but assumed to be lost. Thus, with a positive probability of unstable (blocked) queues, the cumulative distribution function of the response time will not reach one, but rather one minus the probability of blocking. This type of distribution is called a defective distribution (Trivedi, 2002). Admission control, by for instance dropping requests, might be used as a mechanism to prevent web servers from becoming overloaded (Bhatti and Friedrich, 1999), and hence, assuming that arriving requests are lost when the queues are unstable is not unreasonable.

By the decomposition, we implicitly assume that the service attains a steady state in each of the stable failure scenarios. This means that the estimated response time distributions are based on steady-state assumptions, and hence, we do not manage to capture the transient behaviour of the service after a failure or repair has occurred. However, this approach can be justified by the fact that the time in a failure scenario is much longer than the inter-arrival and service times of the performance submodel (Trivedi et al., 1993).

Next, we present the details of our approach. Since the discussion of the simulation of the performance submodel depends on an understanding of the failure scenarios, we start by discussing the dependability submodel used to analyse the failure scenarios.

## 3.2 Dependability submodel

As discussed, the dependability submodel is used to analyse the failure scenarios, specifically by computing the probabilities of the scenarios. Herein, we assume that the failures occur according to a Poisson process, that the failover and repair times are exponentially distributed, and that the replicas are repaired independently. In this case, it is convenient to develop an analytic model based on a CTMC, from which the scenario probabilities can be obtained fast and, since we assume that the VMs fail independently, we can analyse the tiers separately.

Mathematically, we denote the failure rate of active and passive replicas as $\gamma$ and $\beta$, respectively, we let the failover rate be $\alpha$, and we let the repair rate be $\delta$, identical for both types of replicas. We let $k$ and $m$ be the number of active and passive replicas of a tier in fault-free state, with $\ell = k + m$, and represent the failure model of a single tier as a CTMC. The CTMC is depicted in Figure 3, where the state is represented by the tuple (# active not failed, # passive not failed). When a failure occurs in an active replica, a passive replica is made active after a failover delay if there are any non-failed passive replicas. Moreover, we assume that after a failure, the next failure does not occur before the passive replica is activated. This is a reasonable assumption, since the failover delay is much shorter than the time to failure. Additionally, we assume that a repair does not happen during the failover delay, and that all failovers are successful. After a failure in a replica, the replica starts being repaired regardless of the number of replicas currently being repaired. If the number of active replicas is less than $k$, a repaired replica will become active, while a repaired replica will otherwise be set passive.

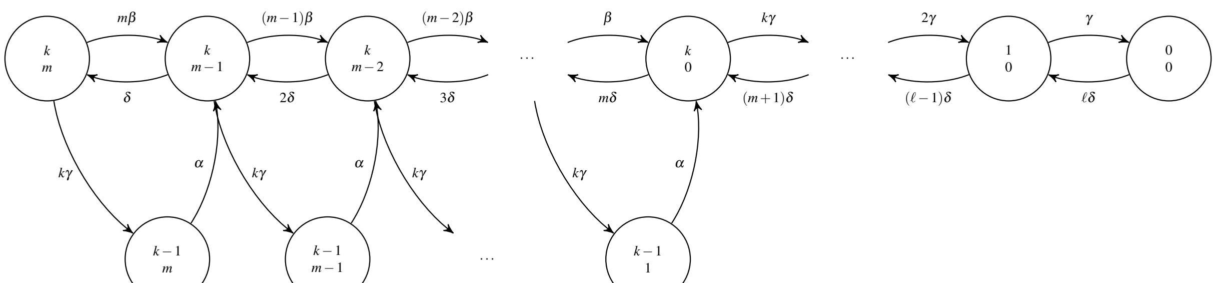

**Figure 3:** Failure model of a tier, with $k$ active and $m$ passive replicas, represented as a CTMC.

We denote the transition rate matrix of the CTMC of tier $q$ as $G_q$ and let $\boldsymbol{\pi}_q$ be the vector of stationary state probabilities of the CTMC. The probabilities $\boldsymbol{\pi}_q$ are found by solving the linear system of equations $G_q\boldsymbol{\pi}_q = \mathbf{0}$ with the additional normalising constraint $\sum_{i\in\mathcal{I}_q} \pi_{qi} = 1$, where $\mathcal{I}_q$ is the set of states of the CTMC of tier $q$ and $\mathbf{0}$ is a vector of only zeros. In the following, vectors are columns identified by bold font, and rows are indicated by a transpose mark.

The discussion of the dependability model above concentrates on a single tier. However, when estimating the response time of a service, we need to consider all service tiers in the simulation model. Therefore, we need to construct compound failure scenarios. A compound failure scenario $j$ is defined by a given number of active replicas of each tier, i.e., a vector $\boldsymbol{a}_j = [a_{j1}, a_{j2}, \ldots, a_{jQ}]^\intercal$, where $a_{jq}$ denotes the number of active replicas of tier $q$ in failure scenario $j$, and a probability $p_j$ of the occurrence of the failure scenario. Since we assume independent failures, the $p_j$ of failure scenario $j$ is computed as the product of the probabilities of having the corresponding number of active replicas of each tier, divided by one minus the blocking probability of the service, $p_B$:

$$p_j = \prod_{q\in\mathcal{Q}} \frac{P\big(\text{tier } q \text{ has } a_{jq} \text{ active replicas}\big)}{1 - p_B}.$$

With $\mathcal{I}_q^S$ denoting the states of the CTMC of tier $q$ where the queues at the tier will be stable, i.e., where the arrival rate is less than the total service rate, $p_B = 1 - \prod_{q\in\mathcal{Q}} \sum_{i\in\mathcal{I}_q^S} \pi_{qi}$. The reason to scale the $p_j$ probabilities by $1/(1 - p_B)$ is to ensure that the $p_j$ probabilities sum to one, which is necessary when weighting the response time distributions of the different strata.

## 3.3 Performance submodel

The service that is simulated in a given failure scenario, can be illustrated as a queueing network with a number of parallel queues in series. The discrete event simulation model is developed according to the process interaction world-view (Banks et al., 2010), with two entities: a *Request Generator* and a *Request*, with replicas modelled as resources. The Request Generator, which process is depicted in Figure 4, generates new requests until the maximum number of simulated requests (to obtain a given number of samples) is reached. Before every generation, the generator holds for a random time sampled from an inter-arrival time distribution. The Request entities, shown in Figure 5, are set up using active replicas as resources in a *mutual exclusion synchronisation* scheme. The Request entity is responsible for recording the observed response times, and is therefore time stamped when initialised. At each tier, it demands one replica resource from which it will receive service and, as discussed, the scheduling is random. If the demanded replica resource is busy, the request is queued until the resource becomes available. After the processing activity, which has a duration randomly sampled from a service time distribution, the request releases the replica resource. When the request has received service in the last tier, the response time is recorded.

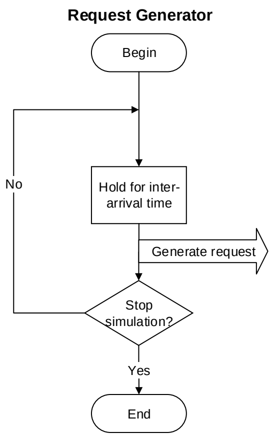

**Figure 4:** Process diagram of the Request Generator entity

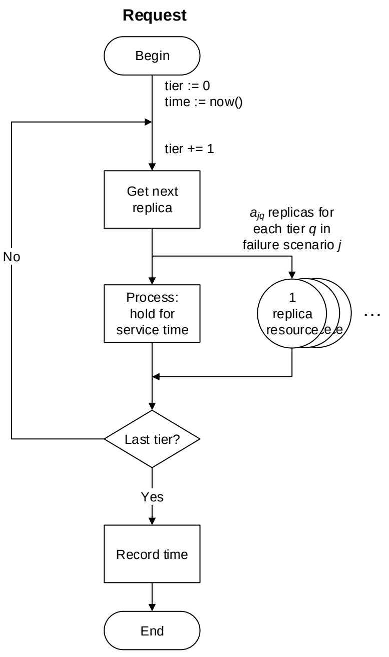

**Figure 5:** Process diagram of the Request entity

Formally, we let the inter-arrival times be distributed according a probability distribution with mean $1/\lambda$ seconds and standard deviation $\psi$. All VMs at a tier have the same features, and that the time to process a request is stochastically distributed according to a given distribution with mean $1/\mu_q$ seconds and standard deviation $\sigma_q$.

## 4 Experimental study

This section presents the results of experiments performed with the simulation approach presented in the previous section. As already stated, if an analytic approach produces a good approximation of the response time distribution, it is often beneficial to use this approach instead of simulating, due to the shorter computational times. Therefore, it is interesting to see how the analytic approximation compares to the simulation approach for both exponential and non-exponential service times. In addition to this comparison, we afterwards demonstrate how the simulation approach can be used as decision support for SPs.

### 4.1 Implementation and sampling scheme

The discrete event simulation model of the performance submodel is implemented in Simula/DEMOS (Birtwistle, 2003). The analytic dependability submodel and the response time approximation approach of Gullhav et al. (2013) are implemented in Mathematica 10.

In the sampling scheme, a stratum corresponds to a failure scenario with probabilities extracted from the dependability submodel. As discussed, a failure scenario is defined by a number of active replicas in each tier, given by the vector $\boldsymbol{a}_j$, and a probability $p_j$. To estimate the response time distribution, we run $R$ replicated simulations of each of the $J$ strata. We emphasise that letting $R$ depend on the strata, by sampling more in strata with high variance could reduce the variance of the estimated distribution. However, this type of sampling scheme is not applied herein. In each of the simulation runs, we first simulate $s$ requests to let the system reach the steady state condition before we start the sampling process. Then, we obtain an estimated response time distribution based on sampling the $n$ next requests. The $n$ samples of run $r$ of stratum $j$ are ordered in a vector $\boldsymbol{t}_{jr} = [t_{jr1}, t_{jr2}, \ldots, t_{jrn}]^\intercal$ so that $t_{jru} \leq t_{jrv}$ for $u < v$. The samples of all runs of stratum $j$ are averaged, to obtain a response time distribution represented by vector $\bar{\boldsymbol{t}}_j = [\bar{t}_{j1}, \bar{t}_{j2}, \ldots, \bar{t}_{jn}]^\intercal$, where $\bar{t}_{ju}$ is defined in equation (1). The estimate of the response time distribution, described by the vector $\hat{\boldsymbol{t}} = [\hat{t}_1, \ldots, \hat{t}_n]^\intercal$, is obtained by weighting the samples of the strata according the probabilities $p_j$. The $\hat{t}_u$ times, representing the $n$ points of the estimated response time distribution, are calculated as shown in equation (2).

$$\bar{t}_{ju} = \frac{\sum_{r=1}^{R} t_{jru}}{R} \quad \forall j = 1, \ldots, J, \forall u = 1, \ldots, n \tag{1}$$

$$\hat{t}_u = \sum_{j=1}^{J} p_j \bar{t}_{ju} \quad \forall u = 1, \ldots, n \tag{2}$$

The response time distributions obtained by this simulation approach are estimates of real distributions, and to obtain precise estimates, one has to obtain enough samples from several independent, replicated runs. Herein, each distribution is constructed by $n$ points, where each point is based on weighting $N = JR$ samples from independent runs. In all tests presented in this section, $s = 100,000$ and $n = 10,000$. Moreover, the number of failure scenarios, $J$, takes a value in $\{2, 4, 8, 12, 16, 24\}$, and we set $N$ to the same number for all cases, and so that $J$ divides $N$, i.e., $R$ is an integer. With the potential values of $J$, we used $N = 768$.

### 4.2 Comparison of the simulation and the analytic approach

In the tests described in this section, we have constructed synthetic test cases based on a three-tier service. The number of active replicas of each tier is given by a vector $\boldsymbol{k} = [3, 4, 3]^\intercal$, and the number of passive replicas is given by a vector $\boldsymbol{m} = [2, 1, 1]^\intercal$. A test case is compounded of a *dependability case* and a *performance case*. The data of the dependability cases is given in Table 1, while the performance cases are shown in Table 2. We refer to the test cases by joining the case labels in the first column of the tables, separated by a hyphen, e.g., `d1-p2`. Regarding the dependability cases, the differences between them are the failure rates. While the active replicas in `d1` have a failure rate of one per day, the failure rates are two and four per day in `d2` and `d3`, respectively. The failure rate of passive replicas is half of the failure rate of active replicas in all cases. For the performance cases, the arrival rate is fixed, but the service rates of the active replicas are decreasing with increasing case label number. For now, we let the inter-arrival and service times be exponentially distributed.

**Table 1:** Dependability model rates (in hours$^{-1}$)

| **Label** | $\gamma$ | $\beta$ | $\alpha$ | $\delta$ |
|---|---|---|---|---|
| `d1` | 0.0417 | 0.0208 | 60.0 | 2.0 |
| `d2` | 0.0833 | 0.0417 | 60.0 | 2.0 |
| `d3` | 0.167 | 0.0833 | 60.0 | 2.0 |

**Table 2:** Performance model rates (in seconds$^{-1}$)

| **Label** | $\lambda$ | $\mu_1$ | $\mu_2$ | $\mu_3$ |
|---|---|---|---|---|
| `p1` | 100 | 90 | 130 | 110 |
| `p2` | 100 | 80 | 110 | 100 |
| `p3` | 100 | 70 | 90 | 90 |
| `p4` | 100 | 60 | 70 | 80 |
| `p5` | 100 | 50 | 50 | 70 |
| `p6` | 100 | 40 | 30 | 60 |

Since the analytic dependability model and the simulation-based performance model are independent, we can reuse the simulation results of a given performance case for the different combinations of dependability cases. That is, the estimated response time distributions for cases `d1-p1`, `d2-p1` and `d3-p1` are based on the same response time samples, but they are weighted with different probability $p_j$ (see equation (2)) when obtaining the estimate.

For the test cases created by combining the dependability and performance cases, we have compared the estimated response time distribution obtained by the simulation approach with the analytic approximation of Gullhav et al. (2013). Table 3 presents the statistic of the Kolmogorov–Smirnov (K-S) test. The K-S test is a non-parametric goodness-of-fit test that is used to test the null hypothesis stating that the two distributions are the same. The K-S statistic $D_n$, where $n$ represents the sample size, is defined in equation (3) below (Trivedi, 2002).

$$D_n = \sup_t |\hat{F}_n(t) - F_0(t)| \tag{3}$$

In equation (3), $\hat{F}_n(t)$ corresponds to the empirical distribution based on the weighted response time samples $\hat{\boldsymbol{t}}$ of size $n$, and $F_0(t)$ corresponds the distribution obtained by the analytic approach. The statistic measures the absolute value of the maximum difference between the empirical distribution and the distribution of the analytic approach over $t$. In Table 3, we can see that the maximum difference between the two distributions is small, and in all cases the corresponding p-value is larger than 0.999 except for two cases, where the p-value is larger than 0.9. To reject the null hypothesis at a significance level of 5%, the p-value should have been less than 0.05, corresponding to $D_n = 0.0136$ for $n = 10000$. Figure 6 shows the distributions for case `d3-p4`, which has the largest value on $D_n$, in a plot, and we see that the they are almost indistinguishable.

As shown, the response time distribution produced by the analytic approximation closely matches the distribution obtained by simulation when the performance model has exponential

**Table 3:** Kolmogorov–Smirnov statistic, $D_n$, when comparing the analytic approximation and the estimate obtained by the simulation approach.

| **Case** | d1 | d2 | d3 |
|---|---|---|---|
| p1 | 0.00299 | 0.00329 | 0.00498 |
| p2 | 0.00153 | 0.00172 | 0.00236 |
| p3 | 0.00150 | 0.00181 | 0.00265 |
| p4 | 0.00178 | 0.00280 | 0.00573 |
| p5 | 0.00071 | 0.00082 | 0.00274 |
| p6 | 0.00321 | 0.00292 | 0.00230 |

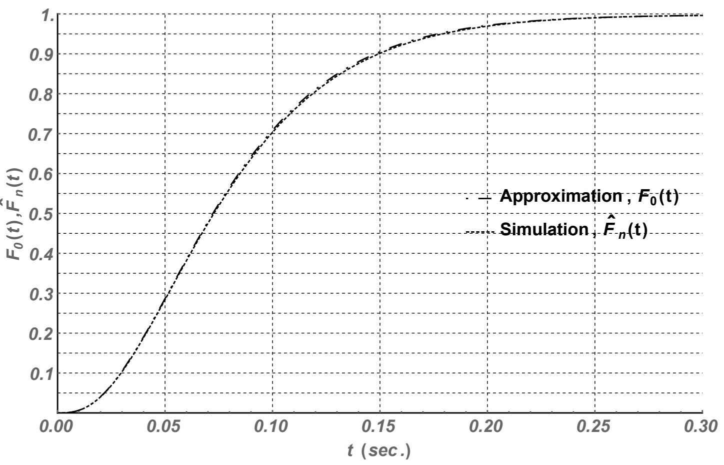

**Figure 6:** Response time distributions for case d3-p4: comparison of analytic approximation and simulation.

inter-arrival and service times. Next, we want to compare the simulation approach with log-normal service time distributions to the analytical approach, still based on exponential service times. While the exponential distribution only takes one parameter, i.e., the rate or mean, the log-normal distribution takes both a location and a scale parameter. When setting these parameters in the experiments, we set them based on the same rates as in Table 2, but with different values for the standard deviation. The standard deviation of an exponentially distributed random variable with mean $1/\mu$, is $1/\mu$, so that the coefficient of variation (CV) (the standard deviation divided by the mean) is constant and equal to one.

Table 4 presents the K-S statistic of the comparison of the distribution obtained by the analytic approach, based on exponential service times, and the distribution obtained by simulating a service with log-normal service times. The comparison is performed for log-normal distributions with different CV. We see that the differences between the distributions are much larger, and the differences are larger for small values on CV; the difference is gradually reduced when reading the table from left to the right. Furthermore, the p-values of these comparisons are 0 in all cases except d1-p6, d2-p6 and d3-p6, where the p-values are less than 0.02. This means that we can reject the null hypothesis, i.e., that the distributions are the same, at a significance level of 2% (or lower). These results are complemented with the plots in Figure 7 for case d1-p1 and Figure 8 for d3-p5. When the CV of the log-normal service time distribution is 0.9 or 1.0, the differences between the curves are smaller than for lower CV. In addition, we see that the blocking probability in case d3-p5 is non-negligible, since the distributions do not reach one.

If the QoS guarantees of a service specify an upper bound on a high percentile of the response time distribution, e.g., the 90th or 95th, the SP will be more concerned about the estimates for these percentiles. Table 5 presents the relative differences at the 90th and 95th percentiles of the response time distributions. The relative difference at the $u$th percentile is calculated

**Table 4:** Kolmogorov–Smirnov statistic, $D_n$, when comparing the analytic approximation and the estimate obtained by the simulation approach with log-normal service times with different coefficient of variation (CV).

| | CV = 0.5 | | | CV = 0.7 | | | CV = 0.9 | | | CV = 1.0 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Case** | d1 | d2 | d3 | d1 | d2 | d3 | d1 | d2 | d3 | d1 | d2 | d3 |
| p1 | 0.161 | 0.161 | 0.159 | 0.0970 | 0.0968 | 0.0953 | 0.0483 | 0.0479 | 0.0461 | 0.0341 | 0.0338 | 0.0319 |
| p2 | 0.168 | 0.168 | 0.168 | 0.102 | 0.102 | 0.101 | 0.0498 | 0.0496 | 0.0488 | 0.0342 | 0.0339 | 0.0330 |
| p3 | 0.178 | 0.177 | 0.177 | 0.108 | 0.108 | 0.107 | 0.0513 | 0.0509 | 0.0498 | 0.0325 | 0.0321 | 0.0308 |
| p4 | 0.190 | 0.189 | 0.187 | 0.116 | 0.115 | 0.113 | 0.0516 | 0.0507 | 0.0481 | 0.0294 | 0.0285 | 0.0260 |
| p5 | 0.214 | 0.214 | 0.213 | 0.133 | 0.133 | 0.131 | 0.0576 | 0.0569 | 0.0550 | 0.0299 | 0.0292 | 0.0273 |
| p6 | 0.250 | 0.250 | 0.250 | 0.157 | 0.156 | 0.155 | 0.0565 | 0.0561 | 0.0548 | 0.0171 | 0.0166 | 0.0154 |

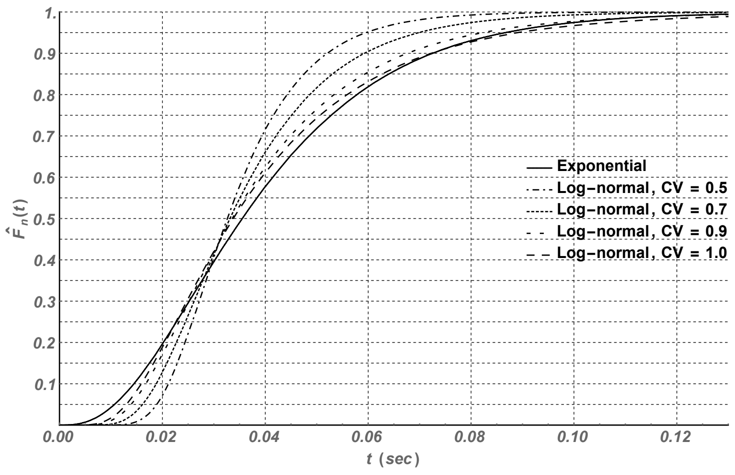

**Figure 7:** Response time distributions for case d1-p1: comparison of analytic approximation (exponential service times) and simulation with log-normal service times with different coefficient of variation (CV)

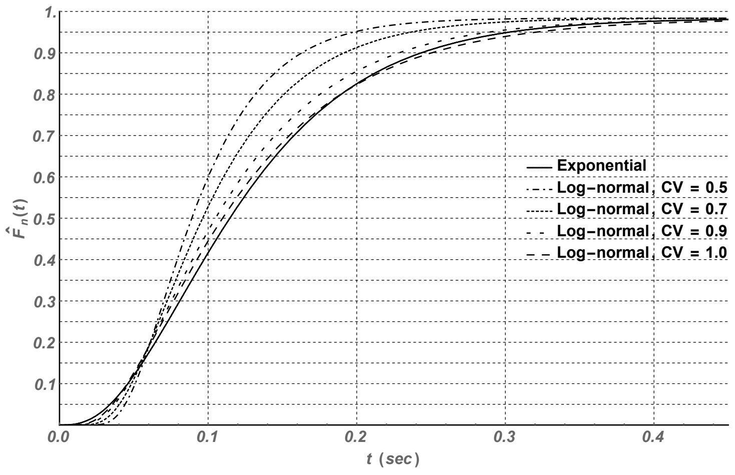

**Figure 8:** Response time distributions for case d3-p5: comparison of analytic approximation (exponential service times) and simulation with log-normal service times with different coefficient of variation (CV)

as $|\tau_u - \hat{\tau}_u|/\hat{\tau}_u$, where $\tau_u$ and $\hat{\tau}_u$ correspond to the times of the $u$th percentile of the analytic distribution and simulated distribution, respectively. We can see that in any case, the relative difference is less than 10%. Generally, in the comparison between the analytic approximation and the simulation with log-normal services times with CV = 0.9, the differences are smaller at the 95th percentile, while the opposite is true for CV = 1.0. The sign of the relative differences is not shown in the table. For the 90th and 95th percentiles, the distribution with log-normal service times with CV = 0.9 has lower response times than the response time distribution with exponential service times over all cases. This means that for CV = 0.9, the analytic approximation provides an upper bound on the response times at the considered percentiles. Concerning the response time distribution with log-normal service times with CV = 1.0, the opposite case is true. For this, the response time distribution of the analytic approximation gives lower response times at the 90th and 95th percentiles. In cases where the analytic distribution deviates from the simulated, but where it can be expected to give upper bounds on the real response times, it is still possible to use the analytic approach to validate if a service configuration has satisfactory QoS by using these safe upper bounds.

**Table 5:** Relative difference (in %) between the response times of the analytic approximation and the simulation approach with log-normal service times with coefficient of variation equal 0.9 and 1.0. The differences in response times is calculated at percentiles 90 and 95. In the cases where N/A is reported, the blocking probability is higher than 5%.

| | **90th percentile** | | | | | | **95th percentile** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CV = 0.9 | | | CV = 1.0 | | | CV = 0.9 | | | CV = 1.0 | | |
| **Case** | `d1` | `d2` | `d3` | `d1` | `d2` | `d3` | `d1` | `d2` | `d3` | `d1` | `d2` | `d3` |
| `p1` | 6.65 | 6.56 | 6.15 | 0.202 | 0.288 | 0.729 | 4.50 | 4.42 | 4.00 | 3.49 | 3.57 | 4.00 |
| `p2` | 6.52 | 6.44 | 6.21 | 0.614 | 0.698 | 1.00 | 4.50 | 4.42 | 4.18 | 4.03 | 4.13 | 4.41 |
| `p3` | 5.99 | 5.89 | 5.61 | 1.51 | 1.62 | 1.96 | 3.95 | 3.83 | 3.53 | 4.80 | 4.91 | 5.28 |
| `p4` | 5.90 | 5.68 | 5.03 | 2.17 | 2.39 | 3.05 | 3.98 | 3.68 | 3.03 | 5.22 | 5.44 | 6.17 |
| `p5` | 6.70 | 6.49 | 5.92 | 2.30 | 2.53 | 3.33 | 5.27 | 5.07 | 4.12 | 4.73 | 5.05 | 6.24 |
| `p6` | 8.56 | 8.39 | 8.24 | 1.98 | 2.19 | 2.75 | 8.33 | 8.43 | N/A | 2.57 | 2.70 | N/A |

## 4.3 Decision support for SaaS providers

In this section, we demonstrate how the simulation approach can be of use for an SP that provides a multi-tier SaaS service. Here, we start by considering the three-tier service defined with the same arrival, service and failure rates as the `d1-p4` case. Furthermore, we let the service times be log-normally distributed with CV = 0.9, and assume that the SP offers its end-users an SLA specifying that the 90th percentile of the response time distribution of the service should be less than 0.3 seconds. In this metric, the response time distribution is scaled by a factor equal to one minus the blocking probability to obtain the defective distribution, which means that blocked, and thereby lost, requests are counted as having response times higher than the threshold. To show how the simulation approach could be utilised for decision support, we assume that the SP wants to assess the response time distribution of a service with different numbers of active and passive replicas at each tier. We name a combination of the number of active and passive replicas at each tier of the service, i.e., the tuple $(\boldsymbol{k}, \boldsymbol{m})$, a replication pattern. Table 6 presents five replication patterns (labelled `RP1` to `RP5`), which we will analyse below.

**Table 6:** Replication patterns

| | $\boldsymbol{k}^\intercal$ | $\boldsymbol{m}^\intercal$ |
|---|---|---|
| `RP1` | $[2, 2, 2]$ | $[0, 0, 2]$ |
| `RP2` | $[2, 2, 2]$ | $[1, 0, 2]$ |
| `RP3` | $[2, 3, 2]$ | $[0, 0, 2]$ |
| `RP4` | $[2, 3, 2]$ | $[1, 0, 2]$ |
| `RP5` | $[3, 2, 2]$ | $[1, 0, 2]$ |

A plot of the five estimated response time distributions is depicted in Figure 9, and Table 7 summarises some important statistics about the distributions. The mean response time reported in the table is the mean response time of the non-blocked failure scenarios and, hence, must be interpreted in connection with the blocking probability, $p_B$. With the SLA specifying that the 90th percentile of the response time distribution should be less than 0.3 second, we examine the five replication patterns in order:

- We start by inspecting `RP1`, and see that the resulting blocking probability is too high for it to satisfy the SLA requirement. Both tier 1 and 2 have a high probability of being in an unstable state, where the arrival rate is higher than the total service rate of the tier. If a passive replica is added to one of these tiers, the blocking probability is expected to decrease. In `RP2`, a passive replica is added to tier 1, and the resulting blocking probability and response time at the 90th percentile is lower.
- If we instead of adding a passive replica to tier 1, add another active replica to tier 2, we obtain `RP3`. We can see that the blocking probability of `RP3` is similar to that of `RP2`, but regarding the 90th percentile, `RP3` is better than `RP2`.
- Combining `RP2` and `RP3` to obtain `RP4`, we now see that the SLA requirement is satisfied, and one might conclude that we should stop investigating more replication patterns.
- However, if `RP4` is modified by adding one active replica to the first tier and jointly remove one at the second, we end up with `RP5`, which also satisfies the SLA requirement.

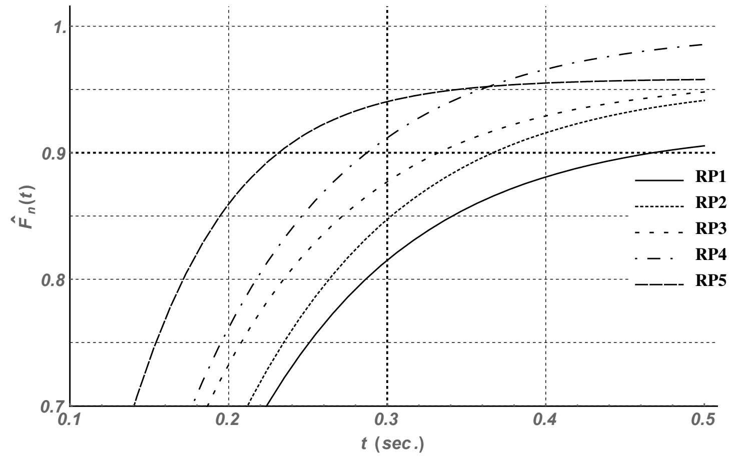

**Figure 9:** Graphical comparison of the estimated response time distribution of the replications pattern. The origin of the plot is in (0.1,0.7).

**Table 7:** Statistics of the replication patterns (times in seconds).

| | Mean response time | Percentiles 90th | Percentiles 95th | $p_B$ |
|---|---|---|---|---|
| `RP1` | 0.168 | 0.468 | N/A | 0.0805 |
| `RP2` | 0.168 | 0.367 | 0.577 | 0.0440 |
| `RP3` | 0.149 | 0.332 | 0.519 | 0.0429 |
| `RP4` | 0.149 | 0.287 | 0.359 | 0.00498 |
| `RP5` | 0.112 | 0.231 | 0.344 | 0.0418 |

In order for the SP to decide whether one should configure the service according `RP4` or `RP5`, one could try to compute the service provisioning cost of both replication patterns and select the cheapest. However, this might not be straight forward. An SP might offer several services that interrelate in some way, e.g., by running on the same infrastructure, so that the cost of a replication pattern of a service is dependent on other services. In this case, it might be possible to use an optimisation model to simultaneously select the optimal replication pattern for each service in the service portfolio of the provider. An optimisation problem that simultaneously considers the placement and replication of service components is discussed and solved in Gullhav and Nygreen (2016).

## 5 Conclusion

In this paper, we have presented a simulation approach used to estimate the response time distribution of multi-tier services while considering failures. This kind of combined performance and dependability analysis is valuable in many applications, including cloud service provisioning as shown herein. In our opinion, this form of analysis should have received more attention by the research community, and by this paper, we hope to encourage more research in this direction. However, in many cases, a fundamental difficulty with this type of analysis is due to the relatively long time between failures, compared to the time between service completions. We overcome this issue by basing our approach on decomposing the problem into two subproblems analysed by independent submodels, specifically an analytic dependability model and a simulation-based performance model.

To verify the approach, we compared the response time distribution obtained by the simulation with the ones obtained by the analytic approach of Gullhav et al. (2013). As expected, the results showed that with exponential inter-arrival and service times, the obtained distributions were indistinguishable. For a non-exponential service time distribution, exemplified by the log-normal distribution, we could see that the simulation approach was required to obtain accurate estimates for the response time distribution. However, the results also showed that the analytic approach for some cases with log-normal service times produced safe bounds on the 90th and 95th percentiles of the response time distribution. In such cases, if the faster analytic approach is used as support in the service configuration decisions, the service configurations would at least be more than good enough with respect to the considered SLA requirements.

Moreover, we demonstrated how the simulation approach can be used as decision support for cloud service providers, and we think that this might form the basis of tool that can help cloud service providers to offer services with higher QoS as requested by Marston et al. (2011).

## References

Al-Kuwaiti M, Kyriakopoulos N and Hussein S (2009). A comparative analysis of network dependability, fault-tolerance, reliability, security, and survivability. *IEEE Communications Surveys Tutorials* 11(2):106–124.

Alam F, Mohan S, Fowler J W and Gopalakrishnan M (2012). A discrete event simulation tool for performance management of web-based application systems. *Journal of Simulation* 6(1): 21–32.

Banks J, Carson J S, Nelson B L and Nicol D M (2010). *Discrete-Event System Simulation.* Pearson: Upper Saddle River, NJ, USA 5th edition.

Bhatti N and Friedrich R (1999). Web server support for tiered services. *Network, IEEE* 13(5): 64–71.

Birtwistle G M (2003). *DEMOS - a System for Discrete Event Modelling on Simula.* University of Leeds: University of Leeds.

Calheiros R N, Ranjan R, Beloglazov A, De Rose C A and Buyya R (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience* 41(1):23–50.

Grottke M, Apte V, Trivedi K and Woolet S (2011). Response time distributions in networks of queues. In Boucherie R J and van Dijk N M (eds), *Queueing Networks* volume 154 of *International Series in Operations Research & Management Science* pp 587–641. Springer US New York, USA.

Gullhav A N and Nygreen B (2015). Deployment of replicated multi–tier services in cloud data centres. *International Journal of Cloud Computing* 4(2):130–149.

Gullhav A N and Nygreen B (2016). A branch and price approach for deployment of multi-tier software services in clouds. *Computers & Operations Research* 75:12 – 27.

Gullhav A N, Nygreen B and Heegaard P E (2013). Approximating the response time distribution of fault-tolerant multi-tier cloud services. In *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing.* IEEE Computer Society: Los Alamitos, CA, USA, pp 287–291.

Jain R (1990). *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling.* John Wiley & Sons: New York, USA.

Lewis P A and Orav E J (1989). *Simulation methodology for statisticians, operations analysts, and engineers* volume 1. Wadsworth & Brooks/Cole: Pacific Grove, CA, USA.

Marston S, Li Z, Bandyopadhyay S, Zhang J and Ghalsasi A (2011). Cloud computing — the business perspective. *Decision Support Systems* 51(1):176 – 189.

Meyer J (1980). On evaluating the performability of degradable computing systems. *IEEE Transactions on Computers* C-29(8):720–731.

Trivedi K S (2002). *Probability & statistics with reliability, queuing and computer science applications.* John Wiley & Sons: New York, USA.

Trivedi K S, Ciardo G, Malhotra M and Sahner R A (1993). Dependability and performability analysis. In Donatiello L and Nelson R (eds), *Performance Evaluation of Computer and Communication Systems* volume 729 of *Lecture Notes in Computer Science* pp 587–612. Springer Berlin Heidelberg.

Urgaonkar B, Pacifici G, Shenoy P J, Spreitzer M and Tantawi A N (2005). An analytical model for multi-tier internet services and its applications. In Eager D L, Williamson C L, Borst S C and Lui J C S (eds), *SIGMETRICS.* ACM. pp 291–302.

Wu L and Buyya R (2012). Service level agreement (SLA) in utility computing systems. In *Grid and Cloud Computing: Concepts, Methodologies, Tools and Applications* pp 286–310. Information Resources Management Association.

Xiong K and Perros H (2009). Service performance and analysis in cloud computing. In *2009 World Conference on Services - I.* IEEE Computer Society: Los Alamitos, CA, USA, pp 693–700.