Babylonian Journal of Networking
Vol.2024, **pp**. 86–98
DOI: https://doi.org/10.58496/BJN/2024/011; ISSN: 3006-5372
https://mesopotamian.press/journals/index.php/BJN

Research Article

Mapping Web Service Characteristics to Queueing Theory Models for Performance Analysis

Raed Abdulkareem Hasan [1,*], Omar Ibrahim Obaid [2], Ans Ibrahim Mahameed AlQassab [3], Husniyah Jasim [4], Saad Ahmed Dheyab [5]

[1] *Renewable Energy Research Unit, Northern Technical University,34002 Mosul, Iraq.*
[2] *Department of Computer, College of Education, AL-Iraqia University, Iraq.*
[3] *Departments of mathematics, College of Education for Pure Science, Tikrit University, Tikrit, Iraq.*
[4] *Department of Electronics, Al-Hawija Technical Institute, Northern Technical University,34002 Mosul, Iraq.*
[5] *College of Engineering, University of Information Technology and Communications, Baghdad, Iraq.*

ARTICLE INFO

Article History
Received 02 May 2024
Revised 12 Jun 2024
Accepted 01 Jul 2024
Published 20 Jul 2024

Keywords
Web Service
SLA
QoS
Network Computing
WSDL

ABSTRACT

The article presents a comprehensive analysis of mapping the characteristics of Web services to queueing theory models, specifically the M/M/1 queueing model. The authors aim to establish a theoretical foundation for modeling the performance of Web services and deriving response time formulae. The key points covered in the article are as follows: 1. Analyzing the characteristics of Web services: The article examines the six fundamental characteristics of queueing systems, including arrival patterns, service patterns, queue discipline, system capacity, and the number of servers, and maps them to the Web service environment. 2. Representation using Kendall's notation: The Web service characteristics are represented using Kendall's notation, resulting in the M/M/1/∞/FCFS model, where M denotes exponential distributions for arrival and service patterns, 1 represents a single server, ∞ represents infinite system capacity, and FCFS represents the first-come, first-served queue discipline. 3. Derivation of response time formulae: The article provides a detailed derivation of the response time formulae for the M/M/1 model using stochastic processes, Markov chains, and the birth-death process. The derivation incorporates steady-state conditions and results in a formula that calculates the probability of completing a request within a user-specified response time. 4. Practical applications: The derived response time formula can be used by service providers to determine the probability of completing user requests within specified Service Level Agreements (SLAs). By setting a threshold value (filter value), the service provider can select requests with a higher probability of completion for further processing. Overall, the article contributes to the understanding of Web service performance modeling by establishing a theoretical foundation based on queueing theory. The mapping of Web service characteristics to the M/M/1 model and the derivation of response time formulae provide valuable insights for service providers to analyze and optimize the performance of their Web services while adhering to SLAs

1. INTRODUCTION

In this section, we delve into the details of ordinary web services and find their matching queueing theory model. Along with the model, we also provide the symbols for representing it. Queueing systems are defined by six main features as outlined in [1] customer arrival pattern: this is the field that studies customer arrivals, and some of the most essential aspects of these processes are: The average number of arrivals per unit of time or the average time between successive arrivals are the ways in which arrival processes are expressed. Based on what is known about its arrival pattern, the pattern can be either deterministic or stochastic. A system is considered deterministic when arrivals occur at regular intervals and stochastic when there is some degree of uncertainty[2].

Upon arrival, clients might be categorized as either patient or impatient. Customers who are patient stay in queue until they are served; customers who are impatient may even leave the queue before they are served. In a Web service environment, the arrivals are shown as stochastic rather than deterministic in the prior description.. Due to the unpredictable nature of user requests, the times it takes for them to arrive can vary[3]. The user's request also comes from a separate entity, typically other

*Corresponding author. Email: Raed.isc.sa@ntu.edu.iq

users or organizations, and is best described by a Poisson distribution. Users' patience while their requests are processed is assumed in this study.;

– Service patterns can be as predictable or stochastic as arrival processes. A random pattern can be described by a probability distribution. The service can be designed to handle a single customer at a time or to handle multiple consumers simultaneously using parallel processing. Also, the service might change based on how many people use it. If the service changes depending on the amount of users, we say that it is state dependent; otherwise, we say that it is state independent.;

Applying the aforementioned features of queueing theory to the context of web services results in discrete processing of each request and, according to an exponential distribution, service times for user requests [4]. In some cases, it is not appropriate to use the exponential distribution of service time [5]. The work of shown that the e-commerce system's service time for user requests follows an exponential distribution [6]. In addition, the majority of the service times for requests on e-commerce websites are quite tiny, and there is a great deal of variety in these times [7]. This is in agreement with the findings of [8], which assert that, according to exponential distribution, there ought to be a high number of requests with short service times and a small number with huge service times.. This proves that the exponential distribution is a reasonable assumption for the time it takes for Web services to respond. In addition, the service pattern is state independent since it is unaffected by the quantity of participants..

queue discipline: it indicates how the system chooses a client to be handled by the server. The standard practice for managing queues is known as first come, first served (FCFS). Aside from random service selection (RSS) and last come, first served (LCFS), there are other queue disciplines.

Various queuing disciplines can coexist in a web service environment. However, this information is essential for Web service modelling since it may be used in the filter to calculate the completion probability based on server load and arrival times. It follows that FCFS is the presumed queue discipline.

system capacity: the amount of room that can physically accommodate clients who are also waiting in queue. We can talk about infinite or finite here. If it is limited, then after the system reaches its capacity, no more customers will be able to join. There can be an unlimited amount of clients waiting in queue if it's endless.

The system capacity is taken for granted to be boundless in the context of the Web service environment. This is due to the fact that, in the event that it is limited, high-priority requests will be inaccessible after the system reaches its capacity. To sidestep this problem, we'll pretend the system's capacity is infinite.

quantity of servers: this quantity can be one server or several servers. At this time, our investigation is limited to a single server.

2. MODEL DERIVATION

One concise way to express the models in queueing theory is using Kendal's notation [9] Queue discipline, number of servers, capacity, service pattern, and arrival pattern When the attributes of the web services are transformed into the form given above, the resulting expression is M/M/1/∞/FCFS. M refers exponential distribution.

1 refers to single server.

∞ refers to capacity of the system.

FCFS pertains to the first come, first served queue discipline.

Service providers can use the generated response time formula to estimate the likelihood of fulfilling user requests within the defined SLAs. The service provider can prioritize requests with a greater completion probability by establishing a threshold value, also known as a filter value. In sum, the essay adds to our knowledge of how to model the performance of web services by providing a theoretical groundwork grounded in queueing theory..

2.1 QoS in Server

Efforts directed at the network layer have the potential to substantially improve QoS. Overloaded servers cause users to wait longer for a response as demand for a resource increases. Consequently, under heavy demand, FIFO scheduling will nullify any gain from network QoS efforts [10]. This emphasises how important it is for servers to have QoS provisioning capabilities. Improving server QoS has been the subject of a lot of recent research and development.

Web Service Description Language (WSDL) doesn't permit non-functional components, according to the author of [11]. The development of a QoS-specific language that can express latency, acceptable round-trip, and various QoS requirements is

crucial for Web services to enable QoS.. According to the author, there is a different layer in IBM's Web service architecture design that allows you to specify QoS attributes. The user can choose the service provider according to the quality of service (QoS) thanks to research in these areas.. The authors of [12] state that while choosing a service provider, reputation should be the primary factor, and that the quality of service offered should serve as a benchmark. The primary idea behind this project is that truth should be one of the criteria used to evaluate service providers, alongside availability, performance, cost, and response time. Defined as "the ability to maintain the lowest difference between projected and achieved level of service metrics," verity is a newly-introduced standard. Customers can find out how trustworthy and dependable service providers are by using verity. For a comparable purpose, [13] details how to incorporate quality of service (QoS) and provisioning pricing into the foundational WS protocol stack, and more especially, the WSDL definition, so that users can choose service providers with excellent QoS. Here are the attributes that were the main focus of the research:

- Service reaction time (SRT) is the amount of time it takes for a service to respond after receiving and processing a request.
- Under heavy traffic, the web service is unlikely to respond within the maximum SRT specified for this kind of request; this is known as the probability of failure.
- When a single request is fulfilled, the service provider charges a one-time transaction price (STP), and when the service is accessible for the entire term, they charge a flat subscription fee (FSP)..

A relevant study on service provider selection and quality of service (QoS) puts forth a method to map QoS to the server and network in [22]. As a result, customers can pick service providers based on the quality of their offerings. The offered solution significantly simplifies the process of integrating Quality of Service into web services[27]. In order to do this, the service provider and the user are both given the chance to configure QoS settings. We introduce a web service broker (WSB) that updates the service's quality of service parameters[23]. Instead of using the UDDI, users can now search up the WSB to retrieve the offered service based on their requirements.

In continuation of previous efforts to let customers choose service providers according to the quality of service (QoS) they offer, researchers have also looked into ways for service providers to tailor their responses to individual users' requirements. Users' expectations and needs fluctuate according to their financial capacity and readiness to pay, thus it's crucial to classify them separately. The amount paid directly correlates to the expected quality. One name for this process is providing distinct services [14]. In [15], differentiated services are provided by the use of WebQoS architecture for admission control, scheduling, and classification. Classification is based on the user and the requested resource.. Two types of users are created as a result of the categorization: basic and premium[24]. Admission control follows classification. As soon as the request count exceeds a certain threshold, admission control is activated and the user's basic requests are rejected[25]. After the scheduling policy has been applied, the user requests are chosen for processing[26]. The following are some of the scheduling policies that were mentioned:

Two scheduling methods exist: tight priority, which gives higher-class requests precedence over lower-class ones, and weighted wriority, which uses weight to determine scheduling.

As an example, if one class is twice as heavy as another, it will be processed twice as often.

One option is to schedule each class to a set capacity, and any unused capacity can be used by another class.

The other option is to schedule each class to a fixed capacity, which cannot be shared.

The third option is to schedule requests based on their earliest deadline. Guarantees of the anticipated response time can be provided by this.

The research's performance was assessed by creating a prototype that was based on Apache. The results show that premium users can indeed receive differentiated service with respect to throughput, reaction time, and error rates. A basic approach for regulating the amount of requests is shown to be sufficient to deliver differentiated service in [16]. It happens at the user level as well as in the kernel (system)[28]. To convert request priority to system priority, two new system calls are added to the Apache and Linux kernels at the kernel-level. A scheduler process is added to the Apache server at the user-level[29]. We create two distinct kinds of scheduling policies:

- A work-conserving approach, where lower-priority requests can be executed if there are insufficient higher-priority requests;
- A non-work-conserving approach, where lower-priority requests cannot be executed even if there are insufficient higher-priority requests.

Experiments show that high-priority requests perform better when the amount of low-priority requests is controlled. Studies pertaining to [17] regulate the quantity of requests are detailed in [18]. By controlling the rate of particular types of requests, the writers are able to provide differentiated service.. The functionality of an architecture called smartware can be separated into three parts: an interceptor, a scheduler, and a dispatcher[30]. The requests are received by the interceptor, which then determines the composite priority. After that, the scheduler sorts the requests into their respective queues[31]. The dispatcher receives the requests that have been chosen using a particular algorithm. At the end of the service, the dispatcher passes the signal on[32]. Based on the aforementioned method, the authors have demonstrated experimentally that differentiated QoS may be delivered with changing throughput.

2.2 Derivation of response time

Presented here are the formulae for the response time in the M/M/1 model. We assume that the system has been operating under steady-state conditions for a sufficient amount of time to warrant studying it [19]. When everything is running smoothly, the average arrival rate shouldn't be higher than the average service rate; otherwise, the line will continue to grow indefinitely[33]. We can find the M/M/1 model's steady-state reaction time by using stochastic process and Markov chain concepts.

Mathematical abstractions of real-world processes whose evolution is controlled by probabilistic principles are known as stochastic processes [20]. The definition of a stochastic process over the parameter space T (time) is {X (t), t T}, a set of random variables[34]. Therefore, the process state at time t is represented by X (t). The following is the definition of T in its discrete-parameter form; it is also defined in its continuous-parameter form:

A discrete-parameter process describes a stochastic process where T is either countable or endlessly countable.

The stochastic process is called a continuous-parameter process if T is an interval.

A stochastic process is called a "Markov process" if the outcomes are unrelated to the past or the present.. This can be explained mathematically as follows. If the stochastic process has been through states $x_1, x_2, \dots, x_n$, then the conditional distribution of $X(t_n)$, depends only on $X(t_n - 1)$ and is expressed mathematically as:

$$\Pr\{X(t_n) \le x_n | \, X(t_1) = x_1, \dots, X(t_n - 1) = x_{n-1}\} = \Pr\{X(t_n) \le x_n | \, X(t_{n-1}) = x_{n-1}\} \quad (1)$$

There are a number of ways to categorise Markov processes according to their time range and state space, or the collection of potential states. There are four distinct Markov processes that can be considered since the time range and state space can take on either a discrete or continuous value.:

- Discrete-state-space Markov chain with a continuous-parameter form;

- Discrete-state-space Markov chain with a discrete-parameter form;

For example, discrete-parameter Markov processes, which have a continuous state space but a discrete temporal range;

-Where the state space and the time range are both continuous, we get Markov processes with continuous parameters.

Being a continuous parameter The Markov chain works well in this setting because time is continuous but web service state spaces are discrete. Given that separate clients are likely to be in separate states of Web services, we say that the state space is discrete. The state space can be defined as discrete given a countable client count. The standard way to depict time is as a continuous variable.

The representation of the continuous-parameter Markov chain is {X,t.,t∈T}, where T is defined as {t:0⪦t≤∞}. The probability of transitioning from state i to state j within the time interval starting at u and ending at s, given a time s>t>u≥0 and two states i and j, can be calculated by adding up all the states from i to j. The following is the mathematical representation of this:

$$p_{ij}(u,s) = \sum_r p_{ir}(u,t) p_{rj}(t,s) \quad (2)$$

– $p_{ij}(u,s)$ - Rate of change from state i to state j within the time interval u–s;

– $p_{ir}(u,t)$ - Expected time required to go from state i to state r, starting at u and ending at t;

– $p_{rj}(t,s)$ - The likelihood of transitioning from state r to state j within the time interval starting at time t and ending at time s.

The Chapman-Kolmogorov (CK) equation is the name given to Equation (1). Here is an alternative way to express equation (1) using matrix notation::

P(u,s) = P(u,t) P(t,s)

By utilising the CK equation, two differential equations—the forward and reverse Kolmogorov equations—are generated:

$$\Pr\{\text{ a change of state in } (t,t+\Delta t)\}=1-p_{ii}(t,t+\Delta t)=q_i(t)\Delta t+o(\Delta t) \qquad (2a)$$

The deviation from one from the chance of remaining in the same state throughout time intervals t and t+Δt represents the likelihood of changing the state during those intervals. It follows that the chance of transitioning from state i to state j during the time interval t and t+Δt is given by,q-i.,t.Δt, and the probability of moving from state i to state j is given as:

$$p_{ij}(t,t+\Delta t)=q_{ij}(t)\Delta t+o(\Delta t) \qquad (2b)$$

– $p_{ij}(t, t+\Delta t)$ Probability to move from state i to j during time interval t and $t+\Delta t$;

– $q_{ij}(t)\Delta t$ Probability to move from state i to j during time interval t and $t+\Delta t$;

– $o(\Delta t)$ Probability to move to state i from states other than j.

Equation (1) is partially differentiated with respect to time t, yielding

$$\frac{\partial}{\partial t} p_{ij}(u,t)=-q_j(t)p_{ij}(u,t)+\sum_{r\neq j} p_{ir}(u,t)q_{rj}(t) \qquad (3a)$$

and

$$\frac{2}{2u}p_{ij}(u,t)=q_i(u)p_{ij}(u,t)-\sum_{r} q_{ir}(u)\, p_{rj}(u,t) \qquad (3b)$$

Both equation (3a) and (3b) are called Kolmogorov's forward and backward equations, respectively..

By making the assumption that the process is homogeneous and unaffected by the number of transitions between states, we may write,q-i.,t.=,q-i. and,q-ij.,t.=,q-ij. in Kolmogorov's forward equation by substituting u = 0. Because of this, we get (3a),

$$\frac{dp_{ij}(0,t)}{dt}=-q_jp_{ij}(0,t)+\sum_{r\neq j} p_{ir}(0,t)q_{rj} \qquad (3C)$$

When you add together all the terms in the previous equation and multiply them by,p-i.(0), you get

$$\frac{dp_j(t)}{dt}=-q_jp_j(t)+\sum_{r\neq j} p_r(t)q_{rj} \qquad (3D)$$

One way to express this in matrix notation is as

$$p'(t) = p(t)Q \qquad (4)$$

$$p(t) = (p_0(t), p_1(t), p_2(t) \ldots) \quad (4a)$$

$$p(t) = (dp_1(t)/dt, dp_1(t)/dt, \ldots) \quad (4b)$$

and

$$Q = \begin{bmatrix} -q_0 & q_{01} & q_{02} & q_{03} & \cdots \\ q_{10} & -q_1 & q_{12} & q_{13} & \cdots \\ q_{20} & q_{21} & q_2 & q_{23} & \cdots \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{bmatrix}$$

A continuous-parameter Markov chain is a good way to describe the Web service environment, as mentioned above. This model also has the birth-death property. As per the birth-death property, at every particular moment the change in state can be either a progress, a regression, or remain unchanged. You can express the change as -1, 0, or +1 using:

Pr {n ⟶ n+1 in (t, t + Δt)} = λnΔt + o (Δt) (n ≥ 0),

Pr {n ⟶ n-1 in (t, t + Δt)} = μnΔt + o (Δt) (n ≥ 1),

Pr {no change in (t, t + Δt)} = 1 – (λn + μn) Δt + o (Δt)

thus

qn,n+1 = λn

qn,n-1 = μn (μn ≠ 0),

qrj = 0 (elsewhere),

qn = λn + μn (q0 = λ0).

The matrix Q is transformed by inserting the values of qi and qij.

$$Q = \begin{bmatrix} -\lambda_0 & \lambda_0 & 0 & 0 & \ldots \\ \mu_1 & -(\lambda_1 + \mu_1) & \lambda_1 & 0 & \ldots \\ 0 & \mu_2 & q_2 & \lambda_2 & \ldots \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{bmatrix}$$

Consequently, the Kolmogorov forward equation (3a) can be used to create a system of differential-difference equations that describe the birth-death process.

$$\frac{dp_j(t)}{dt} = -\left(\lambda_j + \mu_j\right) p_j(t) + \lambda_{j-1} p_{j-1}(t) + \mu_{j+1} p_{j+1}(t) \quad \frac{dp_j(t)}{dt} = -\lambda_0 p_0(t) + \mu_1 p_1(t) \qquad (j \geq 1) \qquad (5)$$

Switching p(,t.=0 in equation (4) to 0 = pQ yields a steady state. .

By substituting $\frac{dp_j(t)}{dt} = 0$ and $p_j(t) = p_j$ in equation (5) we get

$$0 = -\left(\lambda_j+\mu_j\right)p_j+\lambda_{j-1}p_{j-1}+\mu_{j+1}p_{j+1}0 = -\lambda_0p_0+\mu_0p_1 \qquad (j \geq 1) \qquad (6)$$

Like a birth and a death, arrival and departure in the M/M/1 model or Web service environment are interconnected. The reason behind this is that the quantity of consumers present is used to measure the system's state. When something new enters the system, it will raise its state to one, and when something old leaves, it will lower it by one. For all values of n in equation (6), we may obtain the following equation for the birth-death process by substituting,λ-n.=λ and,μ-n.=μ.:

$$0 = -(\lambda+\mu)p_n + \mu p_{n+1}+\lambda p_{n-1}0 = -\lambda p_0 + \mu p_1 \qquad (n\geq 1), \qquad (7a)$$

from the above equation we get

$$p_{n+1}=\frac{\lambda+\mu}{\mu}\ p_n-\frac{\lambda}{\mu}\ p_{n-1}p_1=\frac{\lambda}{\mu}p_0 \qquad (n\geq 1) \qquad , (7b)$$

The values for p2 can be obtained by solving equation (7b) with n = 1.

$$p_2=\frac{m+n}{n}p_1-\frac{m}{n}p_0$$

We obtain by plugging the value of,p1. into the previous equation from equation (7b).

$$p_2=\frac{m+n}{n}\ \frac{m}{n}p_0-\frac{m}{n}p_0$$

$$p_2 = \frac{m}{n}p_0\frac{m+n}{n}1$$

$$= \frac{m}{n}p_0\ \frac{m}{n}+\ 1-1$$

$$= \left(\frac{m}{n}\right)^2 p_0$$

similarly for p_3 we get

$$p_3=\left(\frac{m}{n}\right)^3 p_0$$

thus the value of p_n is

$$p_n=\left(\frac{m}{n}\right)^n p_0$$

Here is the product version of the above equation::

$$= p_0 \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i} \ , \ = p_0 \left(\frac{\lambda}{\mu}\right)^n \qquad (n \geq 1) \qquad (8)$$

The idea that probabilities must sum to one allows us to derive the value of p0. Therefore, the equation given above yields,

$$1 = \sum_{n=0}^{\infty} \left(\frac{\lambda}{\mu}\right)^n p_0$$

let $\rho = \frac{\lambda}{\mu}$, The value p0 can be obtained by inserting it into the previous equation. according to what follows:

$$p_0 = \frac{1}{\sum_{n=0}^{3} t^n} \qquad (9)$$

The term $\sum_{n=0}^{\infty} \rho^n$ can be expanded as a geometric series of $1+\rho+\rho^2+\rho^3+\ldots$Only when $\rho<1$ will the series converge. In order for steady state ($\rho<1$) to take place, it is evident that $\mu<\mu$. The rationale behind this is that the wait will eventually reach infinity if the arrival rate exceeds the service rate. Therefore, in order for the steady state to exist, the arrival rate must be lower than the service rate.

In order to find p0, one way to look at the summation term is as the total of all the geometric progression terms.:

$$\sum_{n=0}^{\infty} \rho^n = \frac{1}{1-\rho} \qquad (\rho < 1), \qquad (10)$$

inserting the given value into the aforementioned equation yields;

$$p_0 = 1 - \rho \qquad \left(\rho = \frac{\lambda}{\mu} < 1\right), \qquad (11)$$

And so, the following function gives the steady-state solution for the M/M/1 system.:

$$p_n = (1 - \rho) \qquad \left(\rho = \frac{\lambda}{\mu} < 1\right), \qquad (12)$$

Using the steady state method, we can determine the client's response time in an M/M/1 system, assuming there are n clients waiting for service when the customer arrives. Assuming n clients are in queue, the response time can be expressed as R(n). R1 represents the receiving client and Si represents the ith client's service time. This is a graphic depicting the amount of time it takes to respond to the nth customer, who is essentially the sum of all the customers in front of them in queue:

$$R(n) = R1 + S1 + S2 + S3 + \ldots + Sn+1 \qquad (13)$$

where

R(n) – Customer nth's response time;

R1 – Time remaining when the client really gets the service;

Si – Customer ith's service duration.

R1 will consistently equal its service time rather than the remaining time because of the memory less attribute of the exponential distribution, which means that it "has no memory of the past" [21]. The sum of all n+1 independent exponentially distributed random variables, where the distribution is given by the (n+1)th stage of the Erlang distribution, is R(n), as shown in equation (13). The possibility of completing the nth request within time t is shown by the following form of the Cumulative Distribution Function (CDF):

$$W(t) = \sum_{n=0}^{3} \Pr\{n+1 \text{ completions in } \# \text{ t} \mid \text{arrival found n in system}\} \cdot p^n$$

by inserting in the values of n+1 stage Erlang and pn into the previous equation, we obtain

$$W(t) = (1-t)\sum_{n=0}^{3} t^n \int_0^t \frac{n(nx)^n}{n!}\, e^{-nx}\, dx$$

and when we simplify the previous equation even further, we obtain:

$$W(t) = n(1-t)\int_0^t \sum_{n=0}^{3} t^n\, \frac{t^n (nx)^n}{n!}\, e^{-nx}\, dx$$

after entering in the tn value, we obtain:

$$= n(1-t)\int_0^t \sum_{n=0}^{3} \frac{(mx)^n}{n!}\, e^{-nx}\, dx$$

$$= n(1-t)\int_0^t e^{mx}\, e^{-nx}\, dx$$

$$= n(1-t)\int_0^t e^{-(n-m)x}\, dx$$

and on integration of the above equation gives

$$= n(1-t)\frac{e^{(n-m)x}}{(n-m)}\Big]_0^t$$

$$= n(1-t)\frac{e^{(n-m)t}}{n-m} + \frac{1}{n-m}$$

$$= 1 - e^{-(n-m)t}$$

thus the probability to complete on or before time t is given by

$$W(t) = 1 - e^{-(n-m)t} \quad (14)$$

where,

– λ arrival rate;

– μ service rate;

– t user specified response time.

Equation (14) is modified by substituting the user-specified response time, the rate at which requests are processed, and the rate at which requests arrive as values for (, (and t). The likelihood of finishing within the time the user has set is the output of the equation. The service provider establishes a filter value, such as a likelihood greater than 0.7, based on their operational expertise of the Web services in the past. The second layer receives all user requests whose likelihood is higher than the filter value.

3. RESULT

Users can influence the algorithm's performance by setting response time parameters, such as the maximum waiting time, the penalty per unit time, and the amount they are prepared to pay for the service. If the user chooses an acceptable waiting time, the BES algorithm will try to execute the request before that time. If the service provider is unable to fulfil the user's request within the allotted waiting time, it will be required to purchase time from the user by paying a certain amount for each unit of delay, which is referred to as penalty per unit of time, within the specified limit. The user may experience an infinite waiting period as a potential downside of this strategy. The BES algorithm gets around this by making sure user requests are processed before the maximum waiting time.

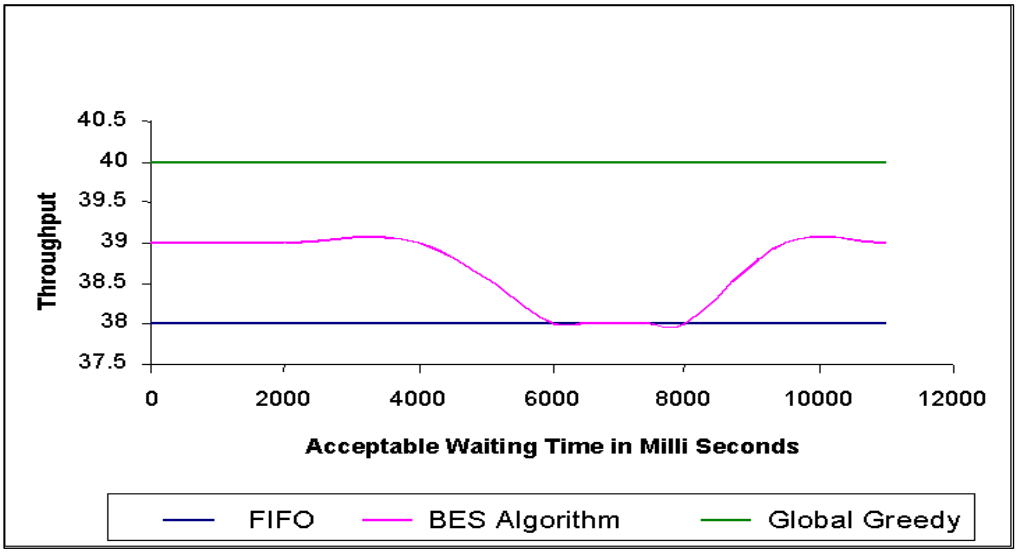

Fig. 1. Changes to the permissible waiting time have an effect on throughput

Across all trials, global greedy search proved to be more effective than BES and FIFO. In equation (4.21), the BES algorithm outperforms FIFO scheduling within the 0–4 second range. This range outperforms FIFO in terms of revenue generated (0.75%) and throughput achieved (2.63%). On the other hand, it mimics FIFO scheduling behavior between 4 and 8 seconds. A more precise estimate would place the allowable waiting time between 25% and 50% of the maximum waiting time. Here, most user requests are handled inside the allotted time, and reordering a new request results in a higher penalty. Consequently, throughput will fall during this period. Crucially, as demonstrated in Figure 2, the amount of requests that can be processed

within the acceptable waiting time for the BES algorithm and the FIFO scheduling stays the same, even though there has been a drop.

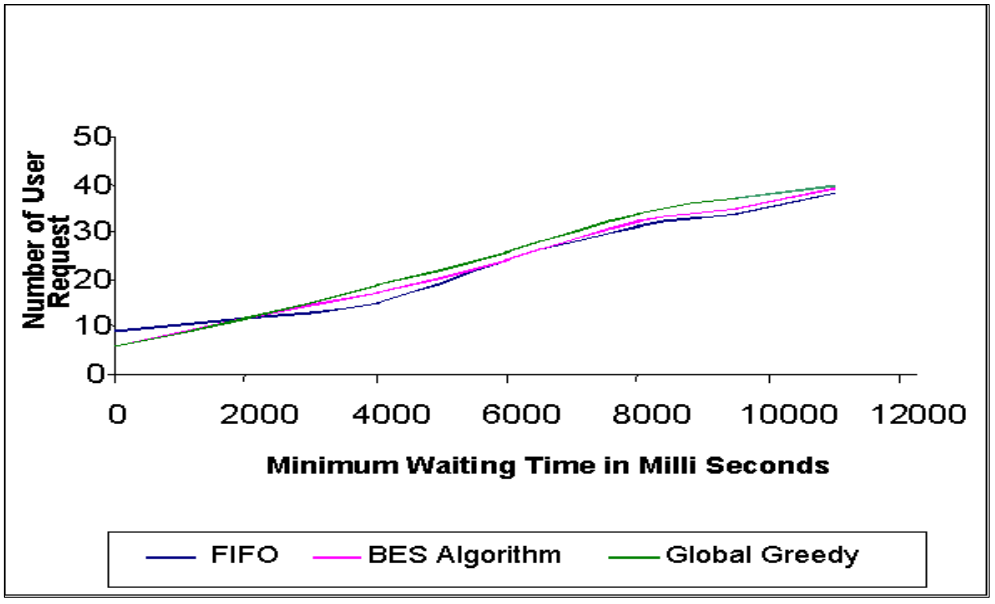

Fig. 2. Effects of changing the allowed waiting time on the volume of user requests

4. CONCLUSION

This essay concludes with a thorough mapping of Web service features to queueing theory models, namely the M/M/1 queueing model. The authors successfully apply Kendall's notation as M/M/1/∐/FCFS to depict the system in Web service contexts by examining the arrival patterns, service patterns, queue discipline, system capacity, and number of servers. Using a thorough mathematical study that incorporates stochastic processes, Markov chains, and the birth-death process, the article subsequently develops the response time equations for the M/M/1 model. Incorporating steady-state circumstances into the derivation yields a formula that determines the likelihood of fulfilling a request within a response time that the user specifies..

There are substantial real-world consequences for service providers stemming from this theoretical basis and the response time formula that was derived. This formula can be used by service providers to calculate the likelihood of fulfilling customer requests within the stated SLAs, provided a threshold value (filter value) is set. To make sure resources are being allocated efficiently and SLAs are being met, requests with a better chance of completion are chosen for further processing. The article fills a need in the literature on Web service performance modelling by describing how queueing theory models can be applied to Web service characteristics. Service providers can gain useful insights and tools for analyzing and optimizing the performance of their Web services while preserving acceptable Quality of Service (QoS) levels through the mapping and derivation of response time equations

Conflicts of Interest

The author's paper explicitly states that there are no conflicts of interest to be disclosed.

Funding

The paper does not disclose any collaborations with funded projects or institutions, indicating the author had no external financial support.

Acknowledgements

The author extends gratitude to the institution for fostering a collaborative atmosphere that enhanced the quality of this research.

References

[1] P. Iyappan and P. Jamuna, "Hybrid Simulated Annealing and Spotted Hyena Optimization Algorithm-Based Resource Management and Scheduling in Cloud Environment," Wireless Personal Communications, vol. 133, no. 2, pp. 1123-1147, 2023.

[2] O. A. Hammood et al., "An effective transmit packet coding with trust-based relay nodes in VANETs," Bulletin of Electrical Engineering and Informatics, vol. 9, no. 2, pp. 685-697, Apr. 2020, doi: 10.11591/eei.v9i2.1653.

[3] M. A. Mohammed, A. A. Kamil, R. A. Hasan, and N. Tapus, "An effective context sensitive offloading system for mobile cloud environments using support value-based classification," Scalable Computing, vol. 20, no. 4, pp. 687-698, Dec. 2019, doi: 10.12694/scpe.v20i4.1570.

[4] T. Mzili, I. Mzili, M. E. Riffi, and G. Dhiman, "Hybrid Genetic and Spotted Hyena Optimizer for Flow Shop Scheduling Problem," Algorithms, vol. 16, no. 6, p. 265, 2023.

[5] R. A. Hasan, M. N. Mohammed, M. A. Bin Ameedeen, and E. T. Khalaf, "Dynamic load balancing model based on server status (DLBS) for green computing," Advanced Science Letters, vol. 24, no. 10, pp. 7777-7782, Oct. 2018, doi: 10.1166/asl.2018.13016.

[6] R. A. Hasan, S. S. Najim, and M. A. Ahmed, "Correlation with the fundamental PSO and PSO modifications to be hybrid swarm optimization," Iraqi Journal for Computer Science and Mathematics, pp. 25-32, Jul. 2021, doi: 10.52866/ijcsm.2021.02.02.004.

[7] S. I. Jasim, M. M. Akawee, and R. A. Hasan, "A spectrum sensing approaches in cognitive radio network by using cloud computing environment," Bulletin of Electrical Engineering and Informatics, vol. 11, no. 2, pp. 750-757, 2022, doi: 10.11591/eei.v11i2.3162.

[8] R. A. Hasan, H. W. Abdulwahid, and A. S. Abdalzahra, "Using Ideal Time Horizon for Energy Cost Determination," Iraqi Journal For Computer Science and Mathematics, vol. 2, no. 1, pp. 9-13, 2021.

[9] L. Sun, W. Shi, J. Wang, H. Mao, J. Tu, and L. Wang, "Research on production scheduling technology in knitting workshop based on improved genetic algorithm," Applied Sciences, vol. 13, no. 9, p. 5701, 2023.

[10] R. A. Hasan, R. A. I. Alhayali, M. A. Mohammed, and T. Sutikno, "An improved fish swarm algorithm to assign tasks and cut down on latency in cloud computing," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 20, no. 5, pp. 1103-1108, 2022.

[11] N. Pholdee, V. K. Patel, S. M. Bureerat, S. M. Sait, and A. R. Yıldız, "Hybrid spotted hyena–Nelder-Mead optimization algorithm for selection of optimal machining parameters in grinding operations," Materials Testing, vol. 63, no. 3, pp. 293-298, 2021.

[12] N. M. Saleh, A. M. Saleh, R. A. Hasan, and H. H. Mahdi, "The Renewable Sustainable and Clean Energy in Iraq Between Reality and Ambition According to the Paris Agreement on Climate Change," Mesopotamian Journal of Big Data, 2022, pp. 36-43.

[13] W. A. Hammood, A. Aminuddin, O. A. Hammood, K. H. Abdullah, D. Sofyan, and M. Rahardi, "Conceptual model of internet banking adoption with perceived risk and trust factors," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 21, no. 5, pp. 1013-1019, 2023.

[14] A. Saleh, N. Saleh, O. Ali, R. A. Hasan, O. Ahmed, A. Alias, and K. Yassin, "Green Building Techniques: Under The Umbrella of the Climate Framework Agreement," Babylonian Journal of Machine Learning, 2024, pp. 1-14.

[15] R. A. Hasan, M. M. Akawee, and T. Sutikno, "Improved GIS-T model for finding the shortest paths in graphs," Babylonian Journal of Machine Learning, 2023, pp. 7-16.

[16] A. H. Ali, M. A. Mohammed, R. A. Hasan, M. N. Abbod, M. S. Ahmed, and T. Sutikno, "Big data classification based on improved parallel k-nearest neighbor," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 21, no. 1, pp. 235-246, 2023.

[17] I. Bala, M. M. Mijwil, G. Ali, and E. Sadıkoğlu, "Analysing the Connection Between AI and Industry 4.0 from a Cybersecurity Perspective: Defending the Smart Revolution," Mesopotamian Journal of Big Data, 2023, pp. 63-69.

[18] S. Abdulrahman and M. Useng, "Blockchain and Distributed Ledger Technologies for IoT Security: A Survey paper," Mesopotamian Journal of Computer Science, 2022, pp. 5-8.

[19] A. L. Hameed, M. Hameed, and R. A. Hasan, "A New Technology for Reducing Dynamic Power Consumption in 8-Bit ALU Design," Iraqi Journal of Industrial Research, vol. 9, no. 3, pp. 12-22, 2022.

[20] M. A. Mohammed, M. M. Akawee, Z. H. Saleh, R. A. Hasan, A. H. Ali, and T. Sutikno, "The effectiveness of big data classification control based on principal component analysis," Bulletin of Electrical Engineering and Informatics, vol. 12, no. 1, pp. 427-434, 2023.

[21] R. A. Hasan, T. Sutikno, and M. A. Ismail, "A Review on Big Data Sentiment Analysis Techniques," Mesopotamian Journal of Big Data, 2021, pp. 6-13.

[22] H. D. K. Al-janabi, H. D. K. Al-janabi, and R. A. H. Al-Bukamrh, "Impact of Light Pulses Generator in Communication System Application by Utilizing Gaussian Optical Pulse," in 2019 22nd International Conference on Control Systems and Computer Science (CSCS), May 2019, pp. 459-464.

[23] Q. Luo, J. Li, and Y. Zhou, "Spotted hyena optimizer with lateral inhibition for image matching," Multimedia Tools and Applications, vol. 78, pp. 34277-34296, 2019.
[24] M. Aljanabi, M. A. Ismail, R. A. Hasan, and J. Sulaiman, "Intrusion Detection: A Review," Mesopotamian Journal of CyberSecurity, 2021, pp. 1-4.
[25] M. A. Mohammed, I. A. Mohammed, R. A. Hasan, N. Ţăpuş, A. H. Ali, and O. A. Hammood, "Green energy sources: issues and challenges," in 2019 18th RoEduNet Conference: Networking in Education and Research (RoEduNet), Oct. 2019, pp. 1-8.
[26] D. Tang, M. Dai, M. A. Salido, and A. Giret, "Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization," Computers in Industry, vol. 81, pp. 82-95, 2016.
[27] C. Zhang, J. Tan, K. Peng, L. Gao, W. Shen, and K. Lian, "A discrete whale swarm algorithm for hybrid flow-shop scheduling problem with limited buffers," Robotics and Computer-Integrated Manufacturing, vol. 68, 102081, 2021.
[28] Y. Z. Li, Q. K. Pan, R. Ruiz, and H. Y. Sang, "A referenced iterated greedy algorithm for the distributed assembly mixed no-idle permutation flowshop scheduling problem with the total tardiness criterion," Knowledge-Based Systems, vol. 239, 108036, 2022.
[29] S. Mahmud, A. Abbasi, R. K. Chakrabortty, and M. J. Ryan, "A self-adaptive hyper-heuristic based multi-objective optimisation approach for integrated supply chain scheduling problems," Knowledge-Based Systems, vol. 251, 109190, 2022.
[30] A. Gümüşçü, S. Kaya, M. E. Tenekeci, İ. H. Karaçizmeli, and I. B. Aydilek, "The impact of local search strategies on chaotic hybrid firefly particle swarm optimization algorithm in flow-shop scheduling," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 8, pp. 6432-6440, 2022.
[31] M. Vali, K. Salimifard, A. H. Gandomi, and T. Chaussalet, "Chaotic-SCA Salp Swarm Algorithm Enhanced with Opposition Based Learning: Application to Decrease Carbon Footprint in Patient Flow," in Handbook of nature-inspired optimization algorithms: the state of the art: volume I: solving single objective bound-constrained real-parameter numerical optimization problems, Cham: Springer International Publishing, 2022, pp. 1-29.
[32] S. A. Ahmed, H. Desa, H. K. Easa, A. S. T. Hussain, T. A. Taha, S. Q. Salih, and P. S. J. Ng, "Advancements in UAV image semantic segmentation: A comprehensive literature review," Multidisciplinary Reviews, vol. 7, no. 6, 2024118-2024118, 2024.
[33] M. A. Mohammed, R. A. Hasan, M. A. Ahmed, N. Tapus, M. A. Shanan, M. K. Khaleel, and A. H. Ali, "A Focal load balancer based algorithm for task assignment in cloud environment," in 2018 10th International Conference on Electronics Computers and Artificial Intelligence (ECAI), June 2018, pp. 1-4.
[34] R. A. Hasan and M. N. Mohammed, "A krill herd behaviour inspired load balancing of tasks in cloud computing," Studies in Informatics and Control, vol. 26, no. 4, pp. 413-424, 2017.