doi:10.1016/j.cor.2008.12.008 • www.stefancreemers.be • info@stefancreemers.be

An advanced queueing model to analyze appointment-driven service systems

Stefan Creemers
Marc Lambrecht

Abstract - **Many service systems are appointment-driven. In such systems, customers make an appointment and join an external queue (also referred to as the "waiting list"). At the appointed date, the customer arrives at the service facility and receives service. Important measures of interest include the size of the waiting list as well as the time spent in the waiting list. We develop a model to assess these performance measures. The model may be used to support strategic decisions concerning server capacity (e.g. how often should a server be online, how many customers should be served during each service session, ...). The model is a vacation model that uses efficient algorithms and matrix analytical techniques to obtain waiting list performance measures.**

Keywords - **appointment system, vacation model, waiting list, queueing system, algorithm**

1 Introduction

Many service systems require customers to make an appointment prior to receiving service. After making an appointment, customers are issued an appointment date (which takes place at some future service session) and join an external queue which is referred to as the "waiting list". Upon the appointment date, the customer is removed from the waiting list and receives service at a service facility (e.g. a doctors office). We will refer to these systems as appointment-driven systems. They may be found in healthcare, legal services, administration and many other service sectors. Often appointment-driven systems are characterized by a chronic backlog of customers to be served. The long waiting times involved, result in reneging behavior and missed company profits. The root of these problems may be found in a mismatch between capacity and demand.

In this article we develop a Customer Assignment System (CAS) that allows to assess the trade-off between capacity and demand in appointment-driven systems. CAS have been studied in [4]. In their work, [4] propose a CAS (which they refer to as an AMQ; an Appointment Making Queue) in which arrivals are allowed to take place at any moment in time. Moreover, they assume interarrival times to be exponentially distributed. In this article we develop a more general model that allows arrivals to take place only during predefined arrival sessions (e.g. during office hours). We assume phase type (PH) interarrival time distributions of customers that may differ over arrival sessions (e.g. the arrival process at an arrival session during the weekend is allowed to differ from the arrival process at an arrival session during a regular working day); allowing for a time dependent arrival process. In addition, we significantly improve computational performance and model accuracy.

The contribution of this article is twofold: (1) the model allows the assessment of detailed waiting list performance measures; (2) the model may serve as a tool to address several strategic issues (i.e. the trade-off between capacity and demand, the frequency and characteristics of service sessions as well as the evaluation of appointment-booking practices).

The remainder of this article is organized as follows. Section 2 gives a detailed problem description. The CAS is presented in section 3. Section 4 discusses model accuracy and computational performance by means of a numerical example. Section 5 concludes.

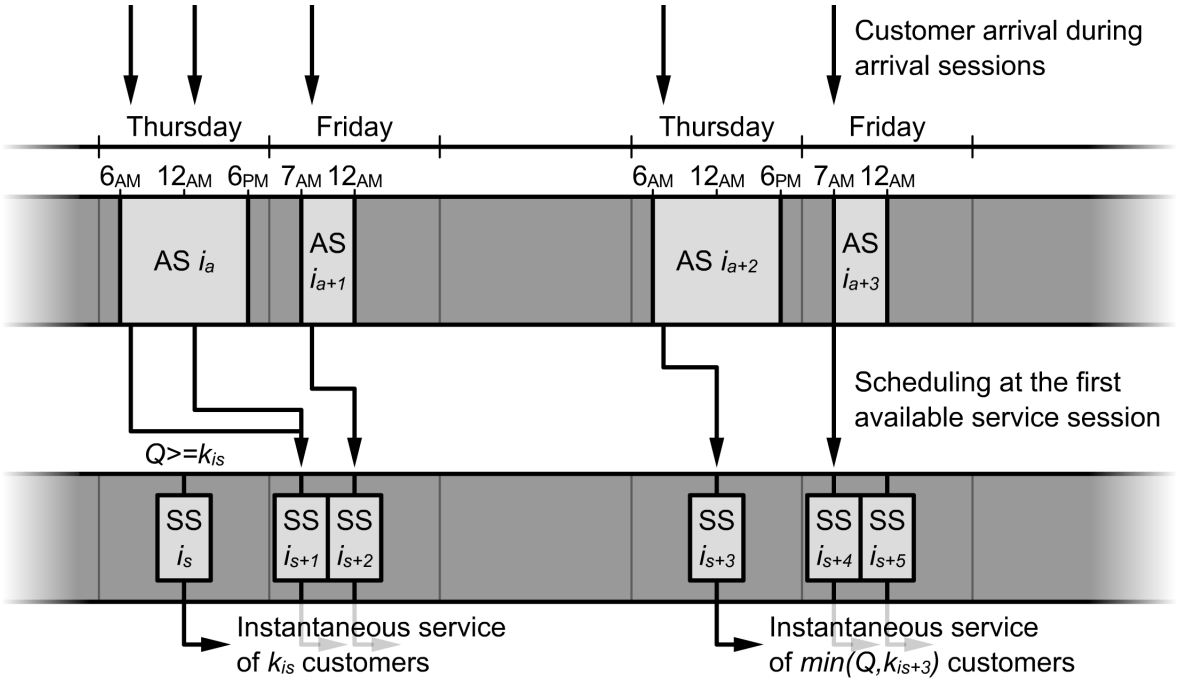
2 Problem Description

In appointment-driven systems customers make an appointment, are assigned an appointment date and are introduced into a waiting list. Upon the appointed date the customer is removed from the waiting list and receives the actual service at the service facility during a predefined service session (e.g. during the opening hours of a doctors office). The CAS observes the queueing behavior of a customer from the making of an appointment (i.e. the arrival) until the start of the service session in which the customer will receive service. As such, the CAS does not observe the time spent waiting in service (i.e. during the service session) nor the actual service process itself (e.g. a patient receiving treatment at the doctors office). The CAS serves only the purpose of obtaining waiting list performance measures.

The CAS has the following properties:

- Customers are only removed from the queue (i.e. the waiting list) at the start of a service session (i.e. at a service instant).
- Customers are removed instantaneously.
- Customers are removed in batches (i.e. we have a batch service process).
- The number of customers removed from the queue depends on: (1) the maximum number of customers allowed to receive service during the upcoming service session; (2) the number of customers in the waiting list at the start of the service session.
- Customers are removed from the queue as soon as possible (i.e. customers are assigned to the first service session in which excess capacity is still available).
- Arrivals at the CAS are allowed to occur only during arrival sessions (e.g. on office hours, during working days).
- Service sessions have fixed starting times (e.g. a doctor always receives patients on Thursday at 12 AM) and arrival sessions have fixed starting and ending times (e.g. appointments may be made on Thursday from 6 AM until 6 PM). As such intervals between these time instances are fixed as well (i.e. these intervals are deterministic).

We illustrate the dynamics of the CAS by means of an example. Figure 1 serves as a guide throughout the text. Suppose we have an appointment-driven system with service sessions on Thursday (at 12 AM) and on Friday (at 7 AM and at 12 AM). Arrivals are allowed to take place during arrival sessions on Thursday (from 6 AM until 6 PM) and on Friday (from 7 AM



AS : Arrival Session
SS : Service Session
Q : Number of customers in queue
k_{is} : Maximum number of customers served during service session i_s

Figure 1: Dynamics of the CAS

until 12 AM). Over the period observed, five customers arrive and are scheduled for service during the first available service session. Note that the capacity of the first service session has already been depleted (i.e. the number of customers in the queue at the start of the first service session is at least equal to the number of customers that is allowed to receive service during that service session). Therefore, the first arriving customer is scheduled at the second service session (in which capacity is still available). At the start of a service session, either all customers are removed from the queue or the maximum number of customers allowed to receive service during the upcoming service session is removed.

The performance measures of interest are: (1) the distribution of the number of customers in the queue at the start of a service session; (2) the average number of customers in the queue; (3) the average waiting time of a customer. Of particular interest is the distribution of the number of customers in the queue at the start of a service session because it indicates the probability of having to serve a certain number of customers during a given service session. [4] link this information to an "appointment system" (for a review of appointment systems refer to [3] among others) and create an appointment-driven queueing system that allows the analysis of appointment-driven systems as a whole. More specifically, an appointment-driven queueing system analyzes: (1) waiting list performance measures (by means of the CAS); (2) server performance (e.g. server overtime, idle time, ...); (3) customer queueing behavior during a service session itself. The CAS studied in this article may be considered as a submodel of an appointment-driven queueing system.

In what follows we discuss the modeling approach, define the basic processes that govern the CAS, characterize the PH distributions used to model system processes and establish a counting process to determine the distribution of the number of arriving customers during a vacation.

2.1 Modeling Approach

The CAS described below may be considered as a vacation model. Vacation models have received a lot of attention in queueing literature over the past decades. Vacation models observe the queueing behavior of systems in which the server takes a vacation (i.e. becomes unavailable) when certain conditions are met. During server vacations, arriving customers are stored in the queue. Once the server returns, service begins once more. A wide variety of vacation models exists. For a general overview we refer to Doshi [5], Takagi [13] and Tian et al. [14].

The CAS has some unique features rendering the modeling exercise rather complex:

- Because customers are served in batches, we have a batch service vacation model.
- At the start of each service session, there is a maximum number of customers allowed to receive service (i.e. there is a maximum number of customers that is removed from the queue). In addition, service occurs instantaneously (indicating that no arrivals take place during service itself). As such, the vacation model has a gated, k-limited service discipline (also referred to as a G-limited service discipline).
- The maximum number of customers removed from the queue, depends on the service session that is about to start (e.g. the maximum number of patients served during a

service session on Thursday is allowed to differ from the maximum number of patients served during a service session on Friday). As such, the vacation model features time-dependent values of k.

- A vacation is initiated at the following time instances (a detailed description of the vacation process is provided in the upcoming sections): (1) the start of a service session (even when no customers are present in the queue); (2) the start of an arrival session; (3) the end of an arrival session. Since these time instances are fixed moments in time, the intervals in between (i.e. the vacation durations) are of fixed (i.e. deterministic) length as well.

- The deterministic length of a vacation depends on the time in the system (e.g. a vacation initiated after a service session on Thursday is allowed to have a different length compared to a service session that is initiated on Friday).

To summarize, the CAS may be modeled as a batch service vacation model featuring: (1) a gated, k-limited service discipline; (2) vacations of deterministic length; (3) time-dependent values of k as well as time-dependent vacation lengths. Because of the complexity involved, more straightforward queueing models (such as deterministic queueing models) do not suffice to accurately assess performance measures.

2.2 Basic Processes

The service process of an appointment-driven system is a succession of service sessions during which customers are served. Each service session i_s is characterized by the maximum number of customers k_{i_s} allowed to receive service. We assume recurring cycles to be present in the succession of service sessions (e.g. a doctor receiving patients every Thursday and Friday). A cycle of service sessions has length T_{c_s}.

Similar to the service process, the arrival process is a succession of arrival sessions i_a during which customers are allowed to make appointments. An arrival session i_a is fully characterized by: (1) the length T_{i_a}; (2) the mean interarrival time $1/\lambda_{i_a}$; (3) the variance of interarrival times $\sigma^2_{i_a}$. We assume recurring cycles to be present in the succession of arrival sessions. A cycle of arrival sessions has length T_{c_a}. An illustration of the cyclic nature of a service and arrival processes is provided in Figure 2. In building the CAS, we will fully exploit the repetitive structure of the service and arrival process.

The vacation process is obtained when superimposing both the service and the arrival process. The vacation process is the continuous (i.e. uninterrupted; service occurs instantaneously) succession of vacations i_v, of deterministic length T_{i_v}. A new vacation i_v is initiated at each instance in time at which (1) a service session starts; (2) an arrival session starts; (3) an arrival session ends. This observation is used to determine T_{i_v}. We illustrate this procedure in Figure 3. Because service and arrival processes are assumed to be cyclic, the vacation process is cyclic as well. The cycle length of the vacation process T_{c_v} equals the least common multiple of T_{c_s} and T_{c_a} (assuming the ratio of T_{c_s} and T_{c_a} is a rational number). A cycle of vacations contains J vacations (for the remainder of the text, index j is defined as $j \in \{1, 2, \ldots, J\}$). We illustrate these principles in Figure 4. Note that, due to the cyclic nature of the vacation process, a vacation of type $j + (iJ)$ is also a vacation of type j (for

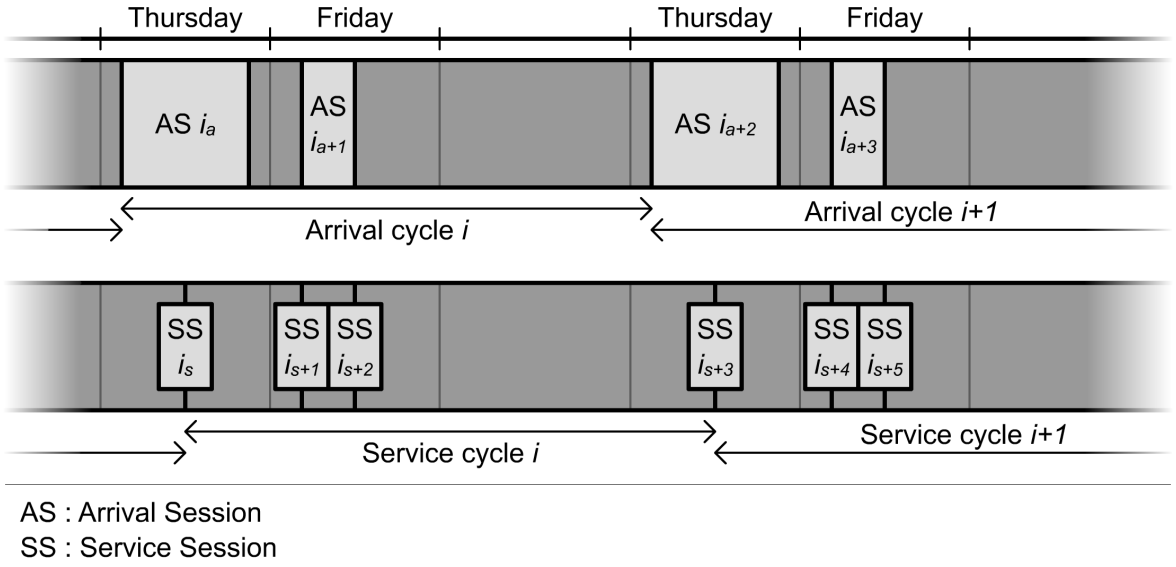


Figure 2: Cyclic nature of the service and arrival process

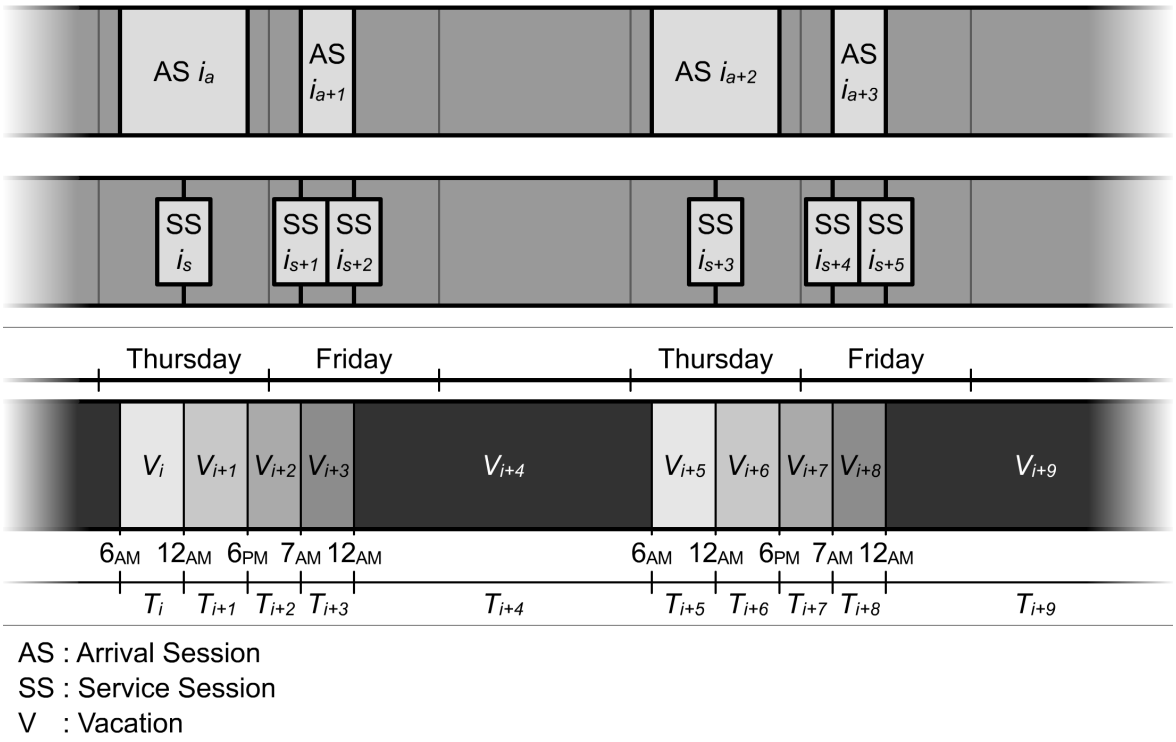


Figure 3: The vacation process at the CAS

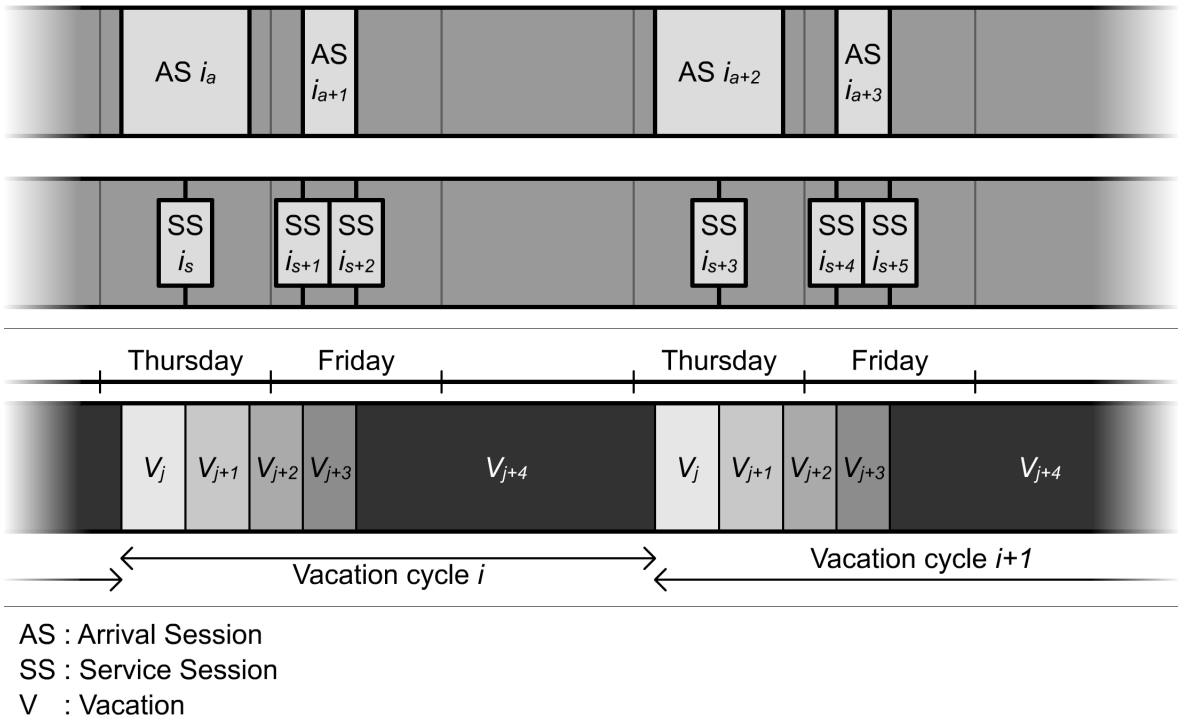


Figure 4: Cyclic nature of the vacation process

the remainder of the text, index i is defined as $i \in \{0, 1, \ldots\}$). In addition, vacations may be divided into different classes (e.g. arrivals are allowed to take place only during vacations of a particular class). A definition of the different vacation classes is presented in section 3.1.1.

2.3 Phase Type Distributions

In order to model a general i.i.d. arrival process and a deterministic vacation process, we make use of continuous time PH distributions. Continuous time PH distributions use exponentially distributed building blocks to approximate (with arbitrary precision) any positive valued continuous distribution. PH distributions are widely implemented in the queueing literature. For a review on the literature and an introduction on PH distributions refer to [9], [7] and [10] among others. A PH distribution is the distribution of time until absorption in a Markov chain with absorbing state 0 and state space $\{0, 1, \ldots, z, z+1\}$. It is fully characterized by parameters $\boldsymbol{\tau}$ and $\mathbf{T}$. $\boldsymbol{\tau}$ is the vector of initial probabilities to start the process in any of the $(z+1)$ transient states and $\mathbf{T}$ is the matrix containing the transition rates between transient states. The infinitesimal generator of the Markov chain representing the PH distribution is presented below:

$$\mathbf{Q} = \left| \begin{array}{cc} 0 & \mathbf{0} \\ \mathbf{t} & \mathbf{T} \end{array} \right|,$$

where $\mathbf{0}$ is a matrix of appropriate dimension containing only zeros and $\mathbf{t} = -\mathbf{T}\mathbf{e}$ (with $\mathbf{e}$ a vector of ones of appropriate size).

In this article we make use of simple phase type approximations of the arrival and vacation process. With respect to the vacation process, an Erlang distribution of sufficient phases provides a sound approximation of the deterministic vacation lengths. A vacation j of deterministic length T_j may be modeled as an Erlang distribution of V phases, each phase having an exponentially distributed duration with mean $1/\omega_j$:

$$1/\omega_j = \frac{T_j}{V}, \tag{1}$$

where V is some number sufficiently large as to minimize the variance of the resulting Erlang distribution of parameters V and ω_j (more specifically, as V approaches infinity, the variance of the corresponding Erlang distribution approaches zero). For the Erlang distribution, $\boldsymbol{\tau}_j$ and $\mathbf{T}_j$ are given by:

$$\boldsymbol{\tau}_j = \begin{array}{c|c} 1 & 1 \\ 2 & 0 \\ \cdots & 0 \\ V-1 & 0 \\ V & 0 \end{array}, \quad \mathbf{T}_j = \begin{array}{c|cccccc} & 1 & 2 & \cdots & V-1 & V \\ \hline 1 & -\omega_j & \omega_j & \cdots & 0 & 0 \\ 2 & 0 & -\omega_j & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ V-1 & 0 & 0 & \cdots & -\omega_j & \omega_j \\ V & 0 & 0 & \cdots & 0 & -\omega_j \end{array}.$$

With respect to the arrival process, we limit ourselves to the matching of the first two moments of the interarrival time distribution (i.e. $1/\lambda_{i_a}$ and $\sigma^2_{i_a}$; the respective mean and variance of the interarrival time distribution are matched by the PH distribution). For notational convenience, let $1/\lambda_j$ and σ_j^2 denote the mean and variance of the interarrival time distribution of arrivals during a vacation of type j. The two-moment matching procedure developed in this section minimizes M_j, the number of phases required to approximate the arrival process at a vacation of type j. Define $\mathbf{M}_j$; the set containing the different arrival phases of the arrival process at a vacation of type j (as such, $|\mathbf{M}_j| = M_j$). Of course, if no arrivals are allowed to occur during a vacation of type j, $M_j = 0$ and $\mathbf{M}_j = \{\emptyset\}$. If arrivals are allowed to occur, we make a distinction between three cases: (1) $C_j^2 = 1$; (2) $C_j^2 > 1$; (3) $C_j^2 < 1$ (where $C_j^2 = \sigma_j^2 \lambda_j^2$ denotes the squared coefficient of variation of interarrival times at a vacation j). In the first case, a simple exponential distribution of parameter λ_j suffices to model the arrival process. $\boldsymbol{\tau}_j$ and $\mathbf{T}_j$ are given by:

$$\boldsymbol{\tau}_j = 1\,, \quad \mathbf{T}_j = -\lambda_j.$$

In the second case, we model the arrival process using a convex mixture of 2 exponential distributions (i.e. using a hyperexponential distribution). The parameters of the hyperexponential distribution matching the interarrival time distribution with rate λ_j and variance σ_j^2 are given by:

$$\beta_{j_1} = \frac{2}{2 - C_j^4 + C_j^6}, \tag{2}$$

$$\beta_{j_2} = 1 - \beta_{j_1}, \tag{3}$$

$$1/\lambda_{j_1} = \frac{1}{2\lambda_j}\left(2 - C_j^2 + C_j^4\right), \tag{4}$$

$$1/\lambda_{j_2} = \frac{1}{\lambda_j} - \frac{1}{\lambda_j C_j^2}, \tag{5}$$

where β_{j_1}, β_{j_2}, λ_{j_1} and λ_{j_2} denote the probability of having an interarrival time that is exponentially distributed with parameter λ_{j_1}, the probability of having an interarrival time that is exponentially distributed with parameter λ_{j_2}, the parameter of the first exponential distribution and the parameter of the second exponential distribution respectively. $\boldsymbol{\tau}_j$ and $\mathbf{T}_j$ are defined as:

$$\boldsymbol{\tau}_j = \begin{array}{c|c} 1 & \beta_{j_1} \\ 2 & \beta_{j_2} \end{array}, \quad \mathbf{T}_j = \begin{array}{c|cc} & 1 & 2 \\ 1 & -\lambda_{j_1} & 0 \\ 2 & 0 & -\lambda_{j_2} \end{array}.$$

With respect to the third case, we model the arrival process using a hypoexponential distribution (a series of exponential distributions whose parameters are allowed to differ; a generalization of the Erlang distribution). The parameters of the hypoexponential distribution matching the interarrival time distribution with rate λ_j and variance σ_j^2 are given by:

$$z_j = \lfloor C_j^{-2} \rfloor - \lfloor C_j^2 \lfloor C_j^{-2} \rfloor \rfloor, \tag{6}$$

$$1/\lambda_{j_1} = \frac{\frac{z_j}{\lambda_j} + \sqrt{-\frac{z_j}{\lambda_j^2} + \frac{C_j^2 z_j}{\lambda_j^2} + \frac{C_j^2 z_j^2}{\lambda_j^2}}}{z_j + z_j^2}, \tag{7}$$

$$1/\lambda_{j_2} = \frac{1}{\lambda_j} - \frac{z_j^2}{\lambda_j \left(z_j + z_j^2\right)} - \frac{z_j \sqrt{-\frac{z_j}{\lambda_j^2} + \frac{C_j^2 z_j}{\lambda_j^2} + \frac{C_j^2 z_j^2}{\lambda_j^2}}}{z_j + z_j^2}, \tag{8}$$

where z_j, λ_{j_1} and λ_{j_2} denote the number of phases of exponential duration of parameter λ_{j_1} that occur prior to the last phase, the parameter of the exponentially distributed interarrival times at the first z_j phases, the parameter of the exponentially distributed interarrival time at the last phase. $\boldsymbol{\tau}_j$ and $\mathbf{T}_j$ are presented below:

$$\boldsymbol{\tau}_j = \begin{array}{c|c} 1 & 1 \\ 2 & 0 \\ \vdots & \vdots \\ z_j & 0 \\ z_j+1 & 0 \end{array}, \quad \mathbf{T}_j = \begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & z_j+1 \\ 1 & -\lambda_{j_1} & \lambda_{j_1} & \cdots & 0 & 0 \\ 2 & 0 & -\lambda_{j_1} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & 0 & 0 & \cdots & -\lambda_{j_1} & \lambda_{j_1} \\ z_j+1 & 0 & 0 & \cdots & 0 & -\lambda_{j_2} \end{array}.$$

For the three cases, M_j equals 1,2 and $z_j + 1$ respectively. A summary of the PH distributions, used to model the CAS, is provided in Figure 5.

2.4 Counting process

A counting process is established in order to obtain the distribution of the number of customers arrived during a vacation of type j. Define $\psi[i, d|j, 0, c]$; the probability of having i arrivals during a vacation of type j and an arrival process at final phase d ($d \in \mathbf{M}_j$) given: (1) a PH distribution of parameters $\mathbf{T}_j$ and $\boldsymbol{\tau}_j$; (2) an arrival process at initial phase c ($c \in \mathbf{M}_j$).

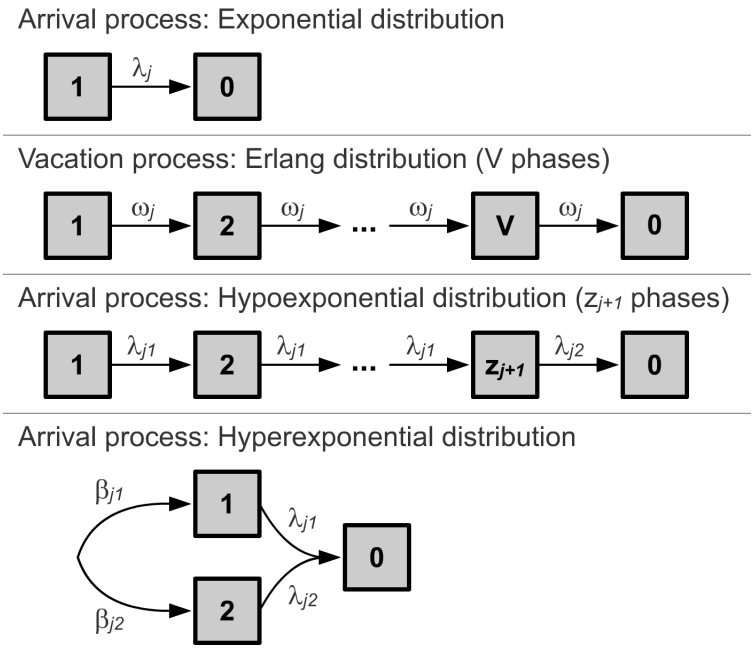


Figure 5: Overview of PH distributions used at the CAS

The distribution of the number of arrivals may be obtained through a counting process of the MAP (Markovian Arrival Process) characterized by $\mathbf{C}_{j_0}$ and $\mathbf{C}_{j_1}$. The counting process has a continuous time rate matrix:

$$\mathbf{Q}_j = \begin{vmatrix} \mathbf{C}_{j_0} & \mathbf{C}_{j_1} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{C}_{j_0} & \mathbf{C}_{j_1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_{j_0} & \mathbf{C}_{j_1} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{C}_{j_0} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \ddots \end{vmatrix},$$

where $\mathbf{C}_{j_0} = \mathbf{T}_j$ and $\mathbf{C}_{j_1} = \mathbf{t}_j \boldsymbol{\tau}_j^{\top}$. The transition probabilities of the counting process during a vacation of deterministic length T_j are given by:

$$\mathbf{C}_j\left(T_j\right) = e^{T_j \mathbf{Q}_j} = \sum_{i=0}^{\infty} \frac{T_j^i}{i!} \mathbf{Q}_j^i. \tag{9}$$

In order to avoid numerical problems and to enhance computational performance, we apply a uniformization argument to the counting process. More specifically, define:

$$\mathbf{P}_j = \frac{\mathbf{Q}_j}{\lambda_j^{max}} + \mathbf{I}, \tag{10}$$

where $\mathbf{I}$ is an identity matrix of appropriate dimension and λ_j^{max} is the largest arrival rate of the PH distribution of parameters $\mathbf{T}_j$ and $\boldsymbol{\tau}_j$. We have [15]:

$$\mathbf{C}_j\left(T_j\right) = e^{-T_j \lambda_j^{max}} \sum_{i=0}^{\infty} \frac{\left(T_j \lambda_j^{max}\right)^i}{i!} \mathbf{P}_j^i. \tag{11}$$

Since we are only interested in transitions moving from states with queue size zero, we only need to observe the first block row of $\mathbf{C}_j(T_j)$. More specifically, the first block row of $\mathbf{C}_j(T_j)$ holds the distribution of the number of arrivals during a vacation of type j (i.e. probabilities $\psi[i,d|j,0,c]$). In order to obtain the first block row of $\mathbf{C}_j(T_j)$, it suffices to compute $\mathbf{P}_{j_1}^{(i)}$; the first block row of $\mathbf{P}_j^i$ (for all $i \geq 0$). $\mathbf{P}_{j_1}^{(i)}$ may be obtained through the simple recursive relationship:

$$\mathbf{P}_{j_1}^{(i)} = \left(\frac{\mathbf{C}_{j_0}}{\lambda_j^{max}} + \mathbf{I}\right)\mathbf{P}_{j_1}^{(i-1)} + \left[\mathbf{0} \quad \frac{\mathbf{C}_{j_1}}{\lambda_j^{max}}\mathbf{P}_{j_1}^{(i-1)}\right]. \tag{12}$$

3 A queueing model to analyze appointment-driven systems

The CAS is not a straightforward queueing model. One possible approach to model the CAS is to construct a Markov chain of four dimensions: (1) The queue size $Q : Q \in \{0,1,2,\ldots\}$; (2) the vacation type j; (3) the phase of the arrival process $m : m \in \{1,\ldots,M_j\}$; (4) the phase of the vacation process $v : v \in \{1,\ldots,V\}$.

Unfortunately, the use of multidimensional Markov chains is in general not advisable since it is clear that, as J, M_j or V increase, the resulting statespace grows rapidly. When modeling real life problems, memory and computational constraints are quickly met. In order to efficiently assess performance measures, we divide the problem into two sets of Discrete Time Markov Chains (DTMC):

- A first set of DTMC, $X_j = \{X_j(t) : t \geq 0\}$ monitors the queueing process of customers only at the start of a vacation of type j, prior to removing k_j customers from the queue.
- A second set of DTMC, $X_j^* = \left\{X_j^*(t) : t \geq 0\right\}$ monitors the queueing process of only those customers that arrive during a vacation of type j.

An illustration of the dynamics of both sets of DTMC is provided in Figure 6. The combination of both sets of DTMC allows the description of the queueing behavior of customers in the waiting list. Decomposing the problem significantly improves computational efficiency due to: (1) dimensional reduction (X_j and X_j^* are two dimensional and three dimensional DTMC respectively); (2) avoiding unnecessary computations.

In what follows we use these sets of DTMC to describe the queueing behavior of customers in the waiting list of an appointment-driven system. We provide a detailed description of the DTMC X_j and X_j^*. Next, we demonstrate how to obtain relevant performance measures and in a final section we combine these performance measures to obtain general results.

3.1 The DTMC X_j

The DTMC X_j observes the queueing behavior of customers at the start of a vacation of type j prior to the removal of k_j customers from the queue. Therefore, observation moments occur only at the start of a vacation. The actions taking place in between two successive observation moments (i.e. the start of a vacation of type j and the start of the next vacation of type j) are left unobserved. We refer to Figure 7 for an illustration. In order to take

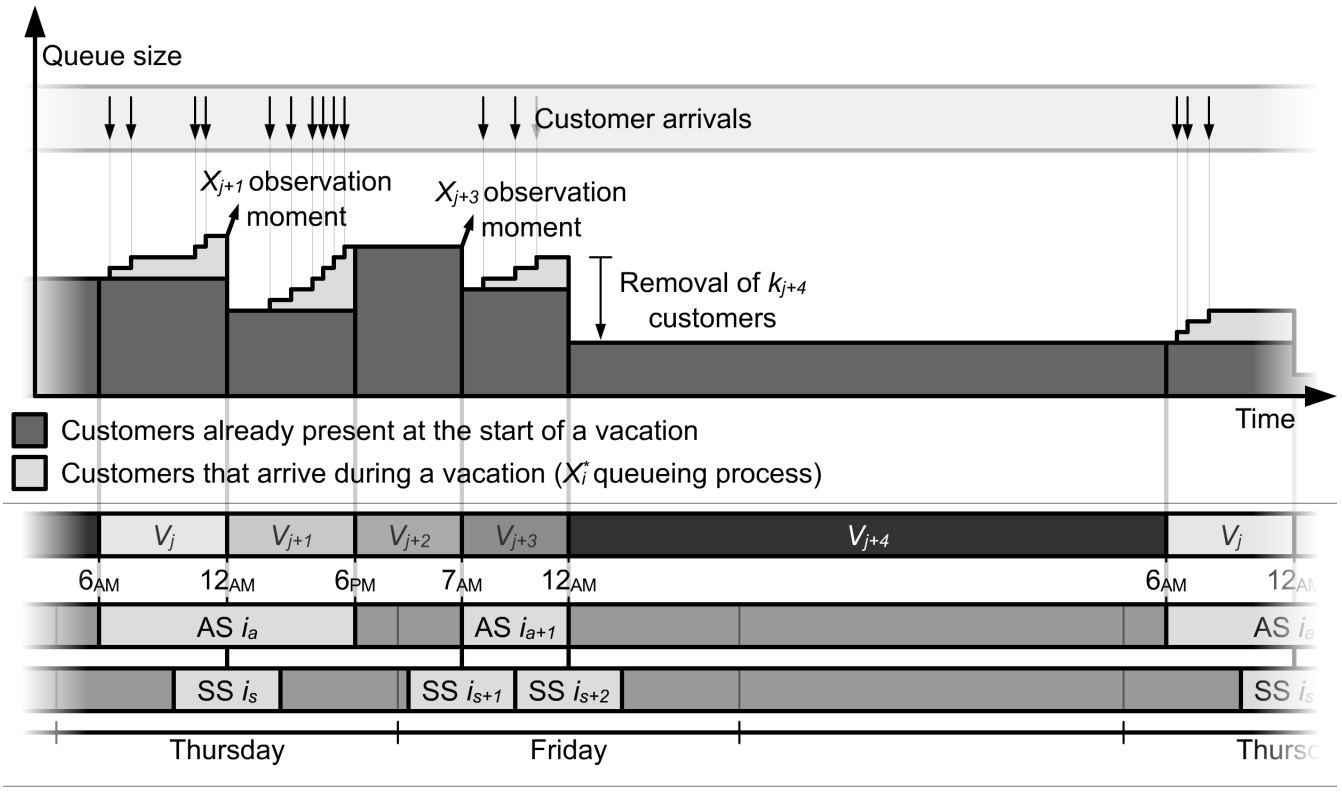


Figure 6: Division of the problem into two sets of DTMC

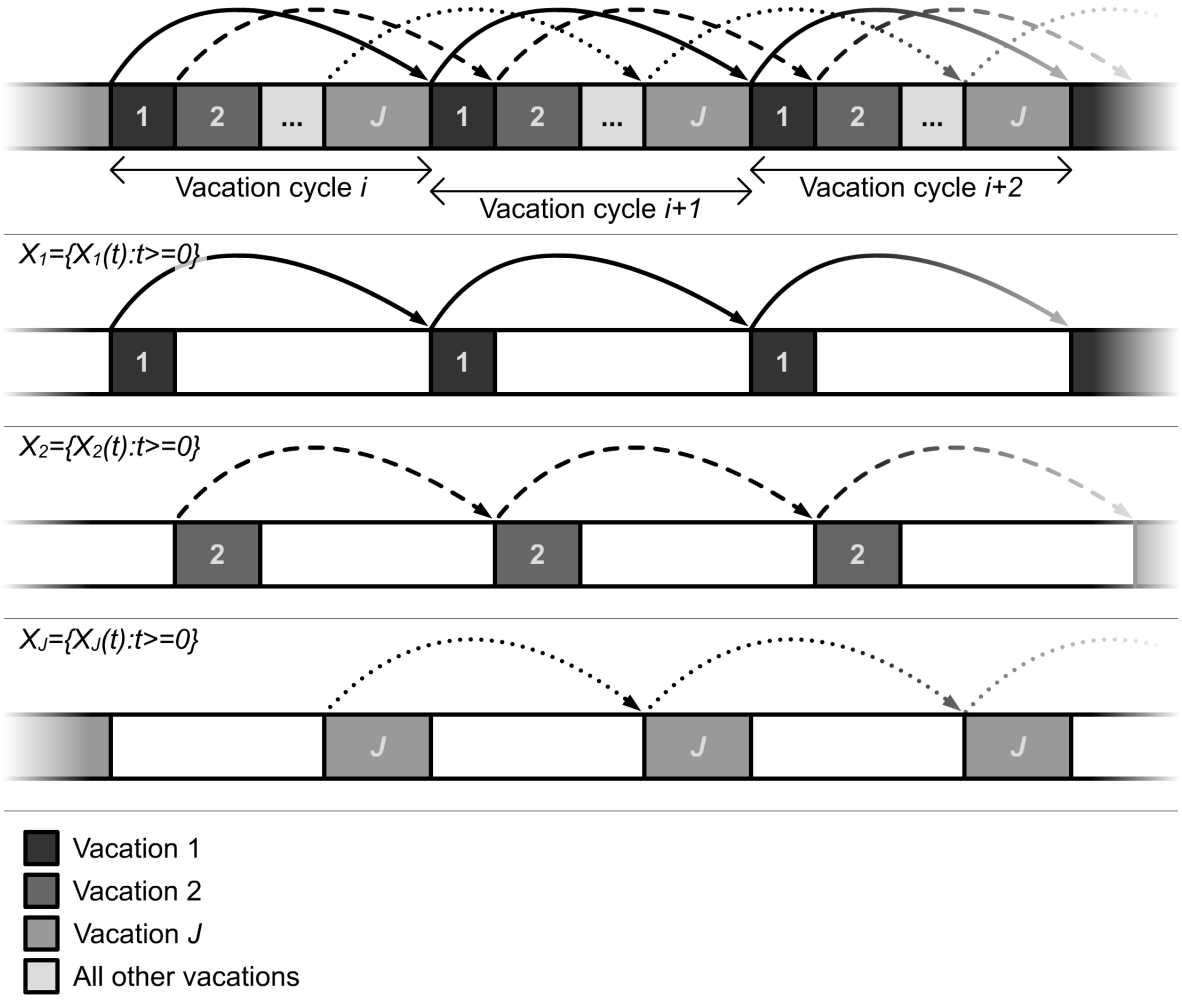


Figure 7: The DTMC X_j

the unobserved alterations of the queueing process into account, one needs to compute all possible outcomes (i.e. resulting queue sizes) and their corresponding probabilities. More specifically, one wants to know the probability of having t customers in the queue at the beginning of a vacation of type j, given that, at the beginning of the previous vacation of type j, s customers were present in the queue.

The DTMC X_j may be defined as a two dimensional stochastic process whose statespace can be represented by pairs $(Q, m)_j$. Define $\phi[n, t, d|j, s, c]$; the probability of moving from a state with queue size s and arrival phase c at the start of a vacation j towards a state with queue size t and arrival phase d at the start of vacation $j + n$ ($c \in \mathbf{M}_j \wedge d \in \mathbf{M}_{j+n}$). In order to obtain $\phi[J, t, d|j, s, c]$ (i.e. the transition probabilities of the DTMC X_j), we first determine $\phi[1, t, d|j, s, c]$ by means of the counting process outlined in section 2.4. These latter probabilities serve as the input of an iterative algorithm that yields the required transition probabilities $\phi[J, t, d|j, s, c]$.

In what follows we first provide a classification of the different vacation classes and show how to obtain $\phi[1, t, d|j, s, c]$ for each vacation class. Next we propose an algorithm to compute probabilities $\phi[J, t, d|j, s, c]$ and use these probabilities to construct the DTMC X_j. We use matrix analytical techniques to obtain the stationary distribution of the number of customers in the queue at the start of a vacation of type j, prior to the removal of k_j customers from the queue. Finally, we present an alternative algorithm that further improves computational performance.

3.1.1 A Classification of Vacations

A distinction between five classes of vacations may be made:

- Class 1 vacations coincide with the end of an arrival session but do not coincide with the start of a service session.
- Class 2 vacations coincide with the start of a service session, do not coincide with the start of an arrival session and no arrival session is still in progress.
- Class 3 vacations coincide with the start of an arrival session but do not coincide with the start of a service session.
- Class 4 vacations coincide with both the start of an arrival session and the start of a service session.
- Class 5 vacations coincide with the start of a service session, do not coincide with the start of an arrival session and an arrival session is still in progress.

We do not allow for (multiple) adjacent arrival sessions. The different classes of vacations are illustrated in Figure 8. Let $c_j : c_j \in \{1, 2, 3, 4, 5\}$ denote the class of a vacation j. Depending on the class, a vacation j is characterized by:

- a deterministic length T_j ($\forall j : c_j \in \{1, 2, 3, 4, 5\}$),
- a number of customers k_j that is served instantaneously at the start of a vacation j ($\forall j : c_j \in \{2, 4, 5\}$ and $k_j = 0\ \forall j : c_j \in \{1, 3\}$),

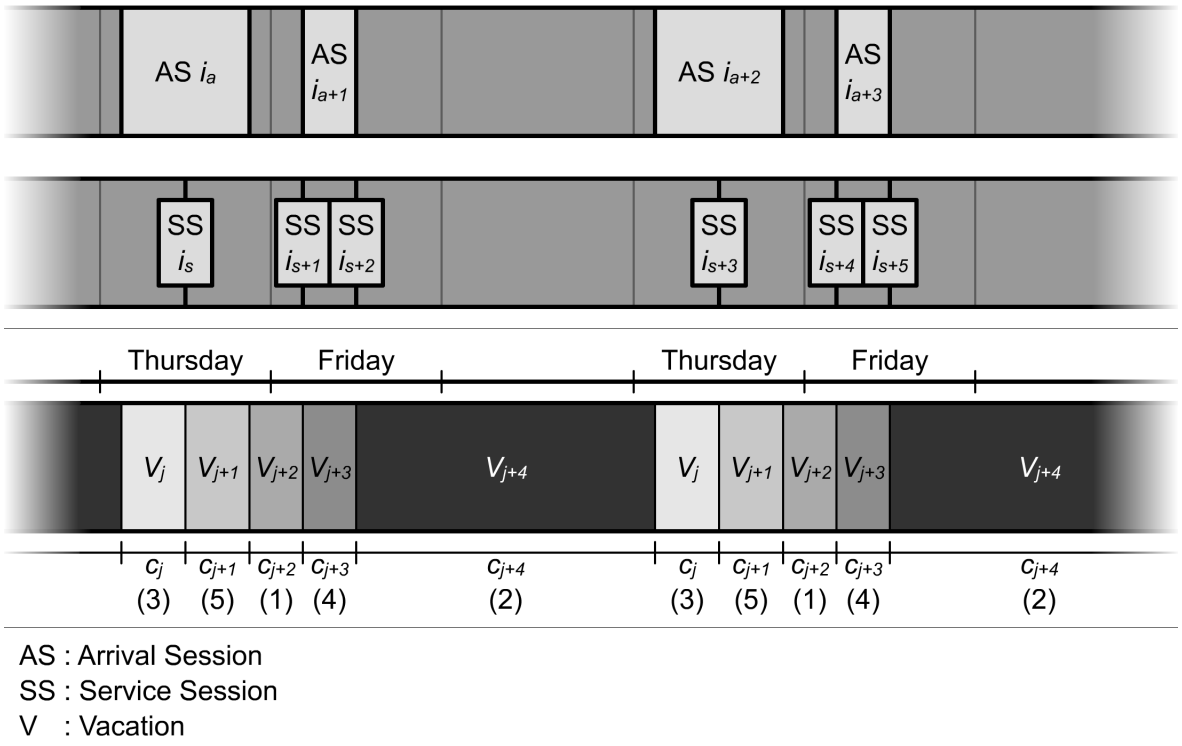


Figure 8: Overview of different vacation classes

- a mean interarrival time $1/\lambda_j$ and variance σ_j^2 ($\forall j : c_j \in \{3,4,5\}$). Note that $M_j = 0$ and $\mathbf{M}_j = \{\emptyset\}$ for all vacations $j : c_j \in \{1,2\}$.

Each of the vacation classes requires a distinct modeling approach.

Class 1 Vacation

Since no arrivals nor service takes place, the queueing process remains unaltered during class 1 vacations. If vacation $j+1$ is of class 2, we have (note that a vacation j of class $c_j \in \{1,2\}$ can never be succeeded by a vacation $j+1$ of class $c_{j+1} \in \{1,5\}$):

$$\phi[1,t,\emptyset|j,s,\emptyset] = \begin{cases} 1 & \forall s,t : s = t, \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

If $c_{j+1} \in \{3,4\}$, we have:

$$\phi[1,t,d|j,s,\emptyset] = \begin{cases} \boldsymbol{\tau}_{j+1}(d) & \forall s,t : s = t, \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

where $\boldsymbol{\tau}_{j+1}(d)$ indicates the probability of starting the arrival process of a vacation $j+1$ at an arrival phase d (refer to section 2.3 for a definition of $\boldsymbol{\tau}_j$).

Class 2 Vacation

Since no arrivals are allowed to take place, the queueing process remains unaltered during class 2 vacations. However, at the start of a class 2 vacation j, a maximum of k_j customers is removed from the queue. If $c_{j+1} = 2$, we have:

$$\phi[1,t,\emptyset|j,s,\emptyset] = \begin{cases} 1 & \forall s,t : s > k_j \wedge t = s - k_j, \\ 1 & \forall s,t : s \leq k_j \wedge t = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

If $c_{j+1} \in \{3,4\}$, we have:

$$\phi[1,t,d|j,s,\emptyset] = \begin{cases} \boldsymbol{\tau}_{j+1}(d) & \forall s,t : s > k_j \wedge t = s - k_j, \\ \boldsymbol{\tau}_{j+1}(d) & \forall s,t : s \leq k_j \wedge t = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

Class 3 Vacation

For a given vacation j we use the counting process developed earlier to obtain the distribution of the number of customers arrived. No services takes place at a class 3 vacation. If $c_{j+1} \in \{1,2\}$, we have:

$$\phi[1,t,\emptyset|j,s,c] = \begin{cases} \sum\limits_{d \in \mathbf{M}_j} \psi[i,d|j,0,c] & \forall s,t : t - s = i, \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

If $c_{j+1} = 5$, we have (note that a vacation j of class $c_j \in \{3,4,5\}$ can never be succeeded by a vacation $j+1$ of class $c_{j+1} \in \{3,4\}$):

$$\phi[1,t,d|j,s,c] = \begin{cases} \psi[i,d|j,0,c] & \forall s,t : t - s = i, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Note that class 3 vacations are not preceded by a vacation during which arrivals are allowed to take place. As such, the initial arrival phase probabilities are given by $\boldsymbol{\tau}_j$. This observation also holds for class 4 vacations.

Class 4 Vacation

Similar to class 3 vacations, we obtain the distribution of the number of customers arrived by means of the counting process discussed previously. At the start of a class 4 vacation j, a maximum of k_j customers is removed from the queue. If $c_{j+1} \in \{1,2\}$, we have:

$$\phi[1,t,\emptyset|j,s,c] = \begin{cases} \sum\limits_{d \in \mathbf{M}_j} \psi\left[i,d|j,0,c\right] & \forall s,t : s > k_j \wedge t = s + i - k_j, \\ \sum\limits_{d \in \mathbf{M}_j} \psi[i,d|j,0,c] & \forall s,t : s \leq k_j \wedge t = i, \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

If $c_{j+1} = 5$, we have:

$$\phi[1,t,d|j,s,c] = \begin{cases} \psi[i,d|j,0,c] & \forall s,t : s > k_j \wedge t = s + i - k_j, \\ \psi[i,d|j,0,c] & \forall s,t : s \leq k_j \wedge t = i, \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

Class 5 Vacation

Class 5 vacations are identical to class 4 vacations except for their definition of the initial arrival phase probabilities. Whereas the start of a class 4 vacation coincides with the start of a new arrival session (i.e. initial arrival phase probabilities are given by $\boldsymbol{\tau}_j$), the start of a class 5 vacation occurs while an arrival process is already in progress. As such, the initial arrival probabilities should reflect the phase of the arrival process at the start of the class 5 vacation (i.e. the initial arrival process phase equals the final arrival process phase of the previous vacation).

3.1.2 Algorithm

The algorithm developed in this section serves the purpose of computing $\phi[J,t,d|j,s,c]$; the probability of moving from a state $(s,c)_j$ at the start of a vacation of type j towards a state $(t,d)_{j+J}$ at the start of the next vacation of type j. Define $\phi[n,u,e|j,(s,t),(c,d)]$; the probability to depart from a state $(s,c)_j$ at the start of a vacation j, to visit state $(t,d)_{j+n-1}$ at the start of vacation $j+n-1$ and to end up in state $(u,e)_{j+n}$ at the start of vacation $j+n$ ($c \in \mathbf{M}_j \wedge d \in \mathbf{M}_{j+n-1} \wedge e \in \mathbf{M}_{j+n}$). We assume at least two vacations to be present in a vacation cycle (i.e. $J \geq 2$; if $J=1$ the requested probabilities $\phi[J,t,d|j,s,c]$ are given by $\phi[1,t,d|j,s,c]$).

Before advancing to the algorithm itself, a number of important properties are established:

Property 1 *When moving from the start of a vacation j to the start of a vacation $j+J$, no more than $Q_c = \sum_{j=1}^{J} k_j$ customers can be removed from the queue.*

Property 2 $\phi[J,t,d|j,s,c] = \phi[J,(t+i),d|j,(s+i),c] \quad \forall s : s \geq Q_c$.

One of the practical implication of these properties is that only states with queue sizes up to Q_c have to be evaluated by the algorithm; resulting in a significant reduction of memory and computational requirements.

The algorithm consists of two main steps: (1) iteration; (2) evaluation. In the upcoming sections, we discuss these steps in detail. A final section provides a general outline of the algorithm.

Step 1: Iteration

Prior to starting the first iteration step we initialize a counter $n=2$ and compute all probabilities $\phi[1,t,d|j,s,c]$ for all vacations j and all possible queue sizes and arrival phases. In a first phase of the iteration step, compute all probabilities $\phi[n,u,e|j,(s,t),(c,d)]$ as follows:

$$\phi[n,u,e|j,(s,t),(c,d)] =$$
$$\begin{cases} \phi[(n-1),t,d|j,s,c]\phi[1,u,\emptyset|(j+n-1),t,\emptyset] & \forall c_{j+n-1} \in \{1,2\} \wedge \forall c_{j+n} = 2, \\ \phi[(n-1),t,d|j,s,c]\phi[1,u,e|(j+n-1),t,\emptyset] & \forall c_{j+n-1} \in \{1,2\} \wedge \forall c_{j+n} \in \{3,4\}, \\ \phi[(n-1),t,d|j,s,c]\phi[1,u,\emptyset|(j+n-1),t,d] & \forall c_{j+n-1} \in \{3,4,5\} \wedge \forall c_{j+n} \in \{1,2\}, \\ \phi[(n-1),t,d|j,s,c]\phi[1,u,e|(j+n-1),t,d] & \forall c_{j+n-1} \in \{3,4,5\} \wedge \forall c_{j+n} = 5, \end{cases} \tag{21}$$

where $\phi[(n-1), t, d|j, s, c]$ is either computed in a previous iteration step or is given by $\phi[1, t, d|j, s, c]$.

In the second phase of the iteration step, we aggregate all resulting probabilities over t and d and obtain probabilities $\phi[n, u, e|j, s, c]$ as follows:

$$\phi[n, u, e|j, s, c] = \sum_{t=0}^{\infty} \sum_{d \in \mathbf{M}_{j+n-1}} \phi[n, u, e|j, (s, t), (c, d)]. \tag{22}$$

After aggregation, we proceed to the evaluation step.

Step 2: Evaluation

At the evaluation step we evaluate if $n = J$. If the condition holds, we have obtained the required probabilities $\phi[J, t, d|j, s, c]$. If the condition does not hold, we increment the counter n and proceed to another iteration step.

Discussion

A general outline of the algorithm is provided in Algorithm 1. Due to the aggregation of

Algorithm 1 Global algorithmic structure

for all Vacations j **do**
  for all s, t, c, d **do**
    Compute $\phi[1, t, d|j, s, c]$
  end for
end for
for all Vacations j **do**
  Set $n = 2$
  while $n < J$ **do**
    for all s, t, u, c, d, e **do**
      $\phi[n, u, e|j, (s, t), (c, d)] = \phi[(n-1), t, d|j, s, c]\phi[1, u, e|(j+n-1), t, d]$
    end for
    for all s, u, c, e **do**
      $\phi[n, u, e|j, s, c] = \sum_{t=0}^{\infty} \sum_{d \in \mathbf{M}_{j+n-1}} \phi[n, u, e|j, (s, t), (c, d)]$
    end for
    Increment n
  end while
end for

probabilities $\phi[n, u, e|j, (s, t), (c, d)]$ and due to the limitation of initial queue size $s : s \leq Q_c$, the number of computations required to obtain probabilities $\phi[J, t, d|j, s, c]$ reduces to a polynomial function.

3.1.3 Stationary distribution of the DTMC X_j

To obtain the stationary distribution at a vacation j, we define $\boldsymbol{\mathcal{P}}_j$, the transition matrix of the DTMC X_j. Following directly from Property 1 and 2 we have that:

$$\begin{array}{rcll} \phi[J,t,d|j,s,c] & = & 0 & \forall s > Q_c \wedge t < (s - Q_c)\,, \\ \phi[J,t,d|j,s,c] & = & \phi[J,(t+i),d|j,(s+i),c] & \forall s \geq Q_c. \end{array} \tag{23}$$

In other words, if $s > Q_c$, it is impossible to deplete the queue when moving from a state $(s,c)_j$ at the start of a vacation of type j towards a state $(t,d)_{j+J}$ at the start of the next vacation of type j. In addition, if $s \geq Q_c$, transition rates moving from states at the start of a vacation of type j towards the start of the next vacation of type j are equal given: (1) equal arrival phases (c,d); (2) equal difference in queue sizes (s,t) and $(s+i,t+i)$. These properties endow the Markov chain $\boldsymbol{\mathcal{P}}_j$ with a special, repetitive structure. More specifically, $\boldsymbol{\mathcal{P}}_j$ may be represented as a non-skip-free $M/G/1$ Markov chain [6]:

$$\boldsymbol{\mathcal{P}}_j = \left| \begin{array}{ccccc} \mathbf{B}_{j,0,0} & \mathbf{B}_{j,0,1} & \mathbf{B}_{j,0,2} & \mathbf{B}_{j,0,3} & \cdots \\ \mathbf{B}_{j,1,0} & \mathbf{B}_{j,1,1} & \mathbf{B}_{j,1,2} & \mathbf{B}_{j,1,3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \\ \mathbf{B}_{j,Q_c-1,0} & \mathbf{B}_{j,Q_c-1,1} & \mathbf{B}_{j,Q_c-1,2} & \mathbf{B}_{j,Q_c-1,3} & \cdots \\ \mathbf{A}_{j,0} & \mathbf{A}_{j,1} & \mathbf{A}_{j,2} & \mathbf{A}_{j,3} & \cdots \\ \mathbf{0} & \mathbf{A}_{j,0} & \mathbf{A}_{j,1} & \mathbf{A}_{j,2} & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_{j,0} & \mathbf{A}_{j,1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array} \right|,$$

where $\mathbf{A}_{j,i}$ and $\mathbf{B}_{j,s,t}$ are $M_j \times M_j$ matrices that depend on the PH distribution used to model the arrival process at a vacation of type j. If no arrivals are allowed to occur during a vacation of type j (i.e. $c_j \in \{1,2\}$), $\mathbf{A}_{j,i}$ and $\mathbf{B}_{j,s,t}$ are scalars and are given by:

$$\begin{array}{rcl} \mathbf{B}_{j,s,t} & = & \phi[J,t,\emptyset|j,s,\emptyset], \\ \mathbf{A}_{j,i} & = & \phi[J,Q_c,\emptyset|j,i,\emptyset]. \end{array}$$

If arrivals are allowed to occur, we once more make a distinction between 3 cases. In the case of $C_j^2 = 1$, $\mathbf{A}_{j,i}$ and $\mathbf{B}_{j,s,t}$ are scalars and are given by:

$$\begin{array}{rcl} \mathbf{B}_{j,s,t} & = & \phi[J,t,1|j,s,1], \\ \mathbf{A}_{j,i} & = & \phi[J,Q_c,1|j,i,1]. \end{array}$$

In the case of $C_j^2 > 1$ we have:

$$\mathbf{B}_{j,s,t} = \begin{array}{c} \\ 1 \\ M_j \end{array} \left| \begin{array}{cc} 1 & M_j \\ \phi[J,t,1|j,s,1] & \phi[J,t,M_j|j,s,1] \\ \phi[J,t,1|j,s,M_j] & \phi[J,t,M_j|j,s,M_j] \end{array} \right|,$$

$$\mathbf{A}_{j,i} = \begin{array}{c} \\ 1 \\ M_j \end{array} \left| \begin{array}{cc} 1 & M_j \\ \phi[J,Q_c,1|j,i,1] & \phi[J,Q_c,M_j|j,i,1] \\ \phi[J,Q_c,1|j,i,M_j] & \phi[J,Q_c,M_j|j,i,M_j] \end{array} \right|.$$

With respect to the hypoexponential case (i.e. $C_j^2 < 1$), $\mathbf{B}_{j,s,t}$ is defined as follows:

$$\mathbf{B}_{j,s,t} = \left|\begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & M_j \\ 1 & \phi[J,t,1|j,s,1] & \phi[J,t,2|j,s,1] & \cdots & \phi[J,t,z_j|j,s,1] & \phi[J,t,M_j|j,s,1] \\ 2 & \phi[J,t,1|j,s,2] & \phi[J,t,2|j,s,2] & \cdots & \phi[J,t,z_j|j,s,2] & \phi[J,t,M_j|j,s,2] \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & \phi[J,t,1|j,s,z_j] & \phi[J,t,2|j,s,z_j] & \cdots & \phi[J,t,z_j|j,s,z_j] & \phi[J,t,M_j|j,s,z_j] \\ M_j & \phi[J,t,1|j,s,M_j] & \phi[J,t,2|j,s,M_j] & \cdots & \phi[J,t,z_j|j,s,M_j] & \phi[J,t,M_j|j,s,M_j] \end{array}\right|.$$

and $\mathbf{A}_{j,i}$ is given by:

$$\mathbf{A}_{j,i} = \left|\begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & M_j \\ 1 & \phi[J,Q_c,1|j,i,1] & \phi[J,Q_c,2|j,i,1] & \cdots & \phi[J,Q_c,z_j|j,i,1] & \phi[J,Q_c,M_j|j,i,1] \\ 2 & \phi[J,Q_c,1|j,i,2] & \phi[J,Q_c,2|j,i,2] & \cdots & \phi[J,Q_c,z_j|j,i,2] & \phi[J,Q_c,M_j|j,i,2] \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & \phi[J,Q_c,1|j,i,z_j] & \phi[J,Q_c,2|j,i,z_j] & \cdots & \phi[J,Q_c,z_j|j,i,z_j] & \phi[J,Q_c,M_j|j,i,z_j] \\ M_j & \phi[J,Q_c,1|j,i,M_j] & \phi[J,Q_c,2|j,i,M_j] & \cdots & \phi[J,Q_c,z_j|j,i,M_j] & \phi[J,Q_c,M_j|j,i,M_j] \end{array}\right|.$$

The repetitive structure observed in $\boldsymbol{\mathcal{P}}_j$ may be exploited using matrix analytical techniques (also referred to as matrix analytical methodology or matrix geometric techniques). Matrix analytical techniques have been studied for several decades and have attracted the attention of many researchers in the queueing field. For an overview of literature and an introduction to matrix analytical techniques, refer to [7], [12], [10] and [1] among others. In short, matrix analytical techniques allow the (numerically) exact analysis of a wide variety of queueing systems featuring some repetitive structure (more specifically, $M/G/1$, $GI/M/1$ and quasi-birth-death processes).

The matrix $\boldsymbol{\mathcal{P}}_j$ can be reblocked into blocks $\boldsymbol{\mathcal{B}}_{j,i}$ and $\boldsymbol{\mathcal{A}}_{j,i}$ of dimensions $M_jQ_c \times M_jQ_c$ as follows:

$$\boldsymbol{\mathcal{P}}_j = \left|\begin{array}{ccccc} \boldsymbol{\mathcal{B}}_{j,0} & \boldsymbol{\mathcal{B}}_{j,1} & \boldsymbol{\mathcal{B}}_{j,2} & \boldsymbol{\mathcal{B}}_{j,3} & \cdots \\ \boldsymbol{\mathcal{A}}_{j,0} & \boldsymbol{\mathcal{A}}_{j,1} & \boldsymbol{\mathcal{A}}_{j,2} & \boldsymbol{\mathcal{A}}_{j,3} & \cdots \\ \mathbf{0} & \boldsymbol{\mathcal{A}}_{j,0} & \boldsymbol{\mathcal{A}}_{j,1} & \boldsymbol{\mathcal{A}}_{j,2} & \cdots \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\mathcal{A}}_{j,0} & \boldsymbol{\mathcal{A}}_{j,1} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}\right|,$$

which can be solved as a traditional $M/G/1$ Markov chain. More specifically, this involves the computation of an auxilliary matrix $\mathbf{G}_j$ which is obtained as the solution of [1]:

$$\mathbf{G}_j = \sum_{i=0}^{\infty} \boldsymbol{\mathcal{A}}_{j,i} \mathbf{G}_j^i. \tag{24}$$

However, as is indicated in [6], such reblocking might increase computational requirements (since it involves computations of $M_jQ_c \times M_jQ_c$ matrices instead of computations of $M_j \times M_j$ matrices). For non-skip-free $M/G/1$ Markov chains, [6] have shown that $\mathbf{G}_j$ may be obtained as follows:

$$\mathbf{G}_j = \mathbf{C}\left(\mathbf{g}_j\right)^{Q_c}, \tag{25}$$

where $\mathbf{g}_j = (\mathbf{G}_{j,0}, \mathbf{G}_{j,1}, \ldots, \mathbf{G}_{j,Q_c-1})$ is the first block row of $\mathbf{G}_j$ and $\mathbf{C}(\mathbf{g}_j)^{Q_c}$ is referred to as the companion matrix and holds the probabilities of moving Q_c steps left in the Markov chain. $\mathbf{C}(\mathbf{g}_j)$ may be represented as:

$$\mathbf{C}(\mathbf{g}_j) = \begin{vmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} & \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} & \\ \vdots & \vdots & \vdots & \ddots & \vdots & \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & & \cdots & \mathbf{I} \\ \mathbf{G}_{j,0} & \mathbf{G}_{j,1} & \mathbf{G}_{j,2} & \cdots & \mathbf{G}_{j,Q_c-1} & \end{vmatrix}.$$

As such, once the first block row of $\mathbf{G}_j$ is known, it is a simple matter to compute $\mathbf{G}_j$. Using the functional iteration algorithm outlined in [6], we compute $\mathbf{G}_j$. Once we obtained $\mathbf{G}_j$, we are able to compute $\boldsymbol{\pi}[i|j]$ using the recursive formula developed in [11]:

$$\boldsymbol{\pi}[n|j] = \left(\boldsymbol{\pi}[0|j]\boldsymbol{\mathcal{B}}_j^{(n)} + \sum_{i=1}^{n-1} \boldsymbol{\pi}[i|j]\boldsymbol{\mathcal{A}}_j^{(n-i)} \right) \left(\mathbf{I} - \boldsymbol{\mathcal{A}}_j^{(0)} \right)^{-1}, \ \forall n : n \geq 1, \tag{26}$$

where:

$$\boldsymbol{\mathcal{A}}_j^{(n)} = \sum_{i=n}^{\infty} \boldsymbol{\mathcal{A}}_{j,i+1} \mathbf{G}_j^{i-n}, \ \forall n : n \geq 0, \tag{27}$$

$$\boldsymbol{\mathcal{B}}_j^{(n)} = \sum_{i=n}^{\infty} \boldsymbol{\mathcal{B}}_{j,i} \mathbf{G}_j^{i-n}, \ \forall n : n \geq 0, \tag{28}$$

and $\boldsymbol{\pi}[0|j]$ is the solution of:

$$\boldsymbol{\pi}[0|j] = \boldsymbol{\pi}[0|j]\boldsymbol{\mathcal{B}}_j^{(0)}, \tag{29}$$

$$-\mu_j = \boldsymbol{\pi}[0|j]\mathbf{b_j} - \mu_j \boldsymbol{\pi}[0|j]\mathbf{e} + \boldsymbol{\pi}[0|j] \left(\mathbf{I} - \boldsymbol{\mathcal{B}}_j \right) \left(\mathbf{I} - \boldsymbol{\mathcal{A}}_j \right)^{\sharp} \mathbf{a}_j, \tag{30}$$

where:

- $\boldsymbol{\mathcal{A}}_j = \sum_{i=0}^{\infty} \boldsymbol{\mathcal{A}}_{j,i}$,
- $\boldsymbol{\mathcal{B}}_j = \sum_{i=0}^{\infty} \boldsymbol{\mathcal{B}}_{j,i}$,
- $\mathbf{a}_j = \sum_{i=0}^{\infty} (i-1) \boldsymbol{\mathcal{A}}_{j,i} \mathbf{e}$,
- $\mathbf{b}_j = \sum_{i=0}^{\infty} i \boldsymbol{\mathcal{B}}_{j,i} \mathbf{e}$,
- $\mu_j = \boldsymbol{\alpha}_j^{\top} \mathbf{a}_j$ and $\boldsymbol{\alpha}_j$ is the stationary distribution vector of $\boldsymbol{\mathcal{A}}_j$,
- operator $(\cdot)^{\sharp}$ denotes the group inverse operation.

$\boldsymbol{\pi}[i|j]$ is the vector of stationary probabilities associated with a queue size $s : s \in \{iQ_c, \ldots, ((i+1)Q_c - 1)\}$. More specifically, $\boldsymbol{\pi}[i|j]$ holds the stationary distribution of states $(s,c)_j : \forall s, c : s \in \{iQ_c, \ldots, ((i+1)Q_c - 1)\}$
$\wedge c \in \mathbf{M}_j$. From $\boldsymbol{\pi}[i|j]$ we obtain $\pi[s,c|j]$, the stationary distribution of being in a state $(s,c)_j$; $\forall s, c : s \in \{0, 1; \ldots\} \wedge c \in \mathbf{M}_j$.

Using the stationary distribution $\pi[s,c|j]$, various performance measures may be derived. In this article, we limit ourselves to the average queue size at the start of a vacation j. Remark that $\pi[s,c|j]$ holds the stationary distribution of the queue size prior to the removal of k_j customers from the queue. Because we are interested in the average number of customers already present in the queue during a vacation of type j (i.e. the number of customers in the queue after the removal of k_j customers), a shifting operation is required:

$$\mathcal{Q}_j = \sum_{s=k_j}^{\infty} \sum_{c \in \mathbf{M}_j} (s - k_j)\pi[s,c|j]. \tag{31}$$

Combining the above result with the average queue size at the DTMC X_j^*, the average size of the waiting list is obtained.

Note that the stationary distribution $\pi[s,c|j]$ (and all performance measures derived thereof) is computed in a numerically exact manner. The stationary distribution $\pi[s,c|j]$ is a key element in determining (1) waiting list performance measures; (2) performance measures of the appointment-driven system as a whole (when linked to an appointment system). Therefore, its exact computation results in a significant improvement of model accuracy.

3.1.4 Alternative computation of the stationary distribution of the DTMC X_j

In this section, we develop an algorithm that uses the stationary distribution $\pi[s,c|j]$ of a single DTMC X_j to determine the stationary distribution of all other DTMC X_{j+n} : $n \in \{1, \ldots, J-1\}$. Together with probabilities $\phi[1,t,d|j,s,c]$, the stationary distribution $\pi[s,c|j]$ serves as the input of the algorithm outlined below. The algorithm is a simple iterative procedure that consists of two steps: (1) iteration; (2) evaluation. In what follows, we discuss these steps in detail. In a final section, we provide a discussion and give a general outline of the algorithm.

Step 1: Iteration

Prior to starting the first iteration step we initialize a counter $n = 1$ and compute the stationary distribution $\pi[s,c|j]$ for a single vacation j. In the iteration step, the stationary

distribution $\pi[t,d|(j+n)]$ is computed as follows:

$$\pi[t,d|(j+n)] = \begin{cases} \sum_{s=0}^{\infty} \sum_{c\in\mathbf{M}_{j+n-1}} \pi[s,c|(j+n-1)]\phi[1,t,\emptyset|(j+n-1),s,\emptyset] & \forall c_{j+n-1}\in\{1,2\}\wedge\forall c_{j+n}=2, \\ \sum_{s=0}^{\infty} \sum_{c\in\mathbf{M}_{j+n-1}} \pi[s,c|(j+n-1)]\phi[1,t,d|(j+n-1),s,\emptyset] & \forall c_{j+n-1}\in\{1,2\}\wedge\forall c_{j+n}\in\{3,4\}, \\ \sum_{s=0}^{\infty} \sum_{c\in\mathbf{M}_{j+n-1}} \pi[s,c|(j+n-1)]\phi[1,t,\emptyset|(j+n-1),s,c] & \forall c_{j+n-1}\in\{3,4,5\}\wedge\forall c_{j+n}\in\{1,2\}, \\ \sum_{s=0}^{\infty} \sum_{c\in\mathbf{M}_{j+n-1}} \pi[s,c|(j+n-1)]\phi[1,t,d|(j+n-1),s,c] & \forall c_{j+n-1}\in\{3,4,5\}\wedge\forall c_{j+n}=5, \end{cases} \tag{32}$$

where $\pi[s,c|(j+n-1)]$ is either computed in a previous iteration step or is given by $\pi[s,c|j]$. After the iteration step, we proceed to the evaluation step.

Step 2: Evaluation

At the evaluation step, we evaluate if $n = J$. If the condition holds, we have computed all the required stationary distributions. If the condition does not hold, we increment the counter n and proceed to another iteration step.

Discussion

A general outline of the algorithm is provided in Algorithm 2.

Algorithm 2 Global algorithmic structure

```
for all Vacations j do
  for all s, t, c, d do
    Compute φ[1, t, d|j, s, c]
  end for
end for
For a single vacation j compute π[s, c|j]
Set n = 1
while n < J do
  for all t, d do
    π[t, d|(j + n)] = Σ_{s=0}^{∞} Σ_{c∈M_{j+n-1}} π[s, c|(j + n − 1)]φ[1, t, d|(j + n − 1), s, c]
  end for
  Increment n
end while
```

Using Algorithm 2, the stationary distribution of all DTMC $X_{j+n} : n \in \{1,\ldots,J-1\}$ can efficiently be obtained, thereby avoiding the use of more complex matrix analytical techniques. As such, computational performance is further enhanced.

3.2 The DTMC X_j^*

The DTMC X_j^* observes the queueing behavior of customers that arrive during a vacation of type j. As such, the DTMC X_j^* does not take into account the queueing process of customers that were already present in the queue at the start of a vacation of type j (the DTMC X_j deals with the queueing behavior of these customers). Note that the DTMC X_j^* does not exist for vacations $j : c_j \in \{1, 2\}$.

In what follows, we present a description of the DTMC X_j^* and use matrix analytical techniques to obtain the average queue size of customers that arrived during a vacation of type j.

3.2.1 Analysis of the DTMC X_j^*

The DTMC X_j^* is a three dimensional stochastic process whose statespace can be represented by triplets $(Q, m, v)_j$. $\boldsymbol{\mathcal{P}}_j^*$ is the transition matrix that corresponds to DTMC X_j^* and is presented below:

$$\boldsymbol{\mathcal{P}}_j^* = \begin{vmatrix} \hat{\mathbf{L}}_j^* & \mathbf{F}_j^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{B}_j^* & \mathbf{L}_j^* & \mathbf{F}_j^* & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{B}_j^* & \mathbf{0} & \mathbf{L}_j^* & \mathbf{F}_j^* & \mathbf{0} & \cdots \\ \mathbf{B}_j^* & \mathbf{0} & \mathbf{0} & \mathbf{L}_j^* & \mathbf{F}_j^* & \cdots \\ \mathbf{B}_j^* & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{L}_j^* & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \ddots \end{vmatrix},$$

where $\hat{\mathbf{L}}_j^*$, $\mathbf{L}_j^*$, $\mathbf{F}_j^*$ and $\mathbf{B}_j^*$ are the respective "local", "forward" and "backward" transition probability matrices. An outline of these matrices is provided below:

$$\mathbf{L}_j^* = \begin{array}{c|cccccc} & 1 & 2 & \cdots & V-1 & V \\ 1 & \boldsymbol{\Lambda}_j^{(1)} & \boldsymbol{\Omega}_j & \cdots & \mathbf{0} & \mathbf{0} \\ 2 & \mathbf{0} & \boldsymbol{\Lambda}_j^{(1)} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ V-1 & \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Lambda}_j^{(1)} & \boldsymbol{\Omega}_j \\ V & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{\Lambda}_j^{(1)} \end{array},$$

$$\mathbf{F}_j^* = \begin{array}{c|cccccc} & 1 & 2 & \cdots & V-1 & V \\ 1 & \boldsymbol{\Lambda}_j^{(2)} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ 2 & \mathbf{0} & \boldsymbol{\Lambda}_j^{(2)} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ V-1 & \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Lambda}_j^{(2)} & \mathbf{0} \\ V & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{\Lambda}_j^{(2)} \end{array},$$

$$\mathbf{B}_j^* = \begin{array}{c|cccccc} & 1 & 2 & \cdots & V-1 & V \\ 1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ 2 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ V-1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ V & \mathbf{\Omega}_j & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \end{array}.$$

and $\hat{\mathbf{L}}_j^* = \mathbf{L}_j^* + \mathbf{B}_j^*$. $\mathbf{\Lambda}_j^{(1)}$, $\mathbf{\Lambda}_j^{(2)}$ and $\mathbf{\Omega}_j$ are the characterizing matrices of $\boldsymbol{\mathcal{P}}_j^*$ and depend on the PH distribution used to model the arrival process. In the case where $C_j^2 = 1$, each of the characterizing matrices can be represented as a scalar:

$$\mathbf{\Lambda}_j^{(2)} = \frac{\lambda_j}{\lambda_j + \omega_j}, \quad \mathbf{\Omega}_j = \frac{\omega_j}{\lambda_j + \omega_j},$$

and $\mathbf{\Lambda}_j^{(1)} = 0$. When using a hyperexponential interarrival time distribution (i.e. $C_j^2 > 1$) we have:

$$\mathbf{\Lambda}_j^{(2)} = \begin{array}{c|cc} & 1 & M_j \\ 1 & \beta_{j_1}\frac{\lambda_{j_1}}{\lambda_{j_1}+\omega_j} & \beta_{j_2}\frac{\lambda_{j_1}}{\lambda_{j_1}+\omega_j} \\ M_j & \beta_{j_1}\frac{\lambda_{j_2}}{\lambda_{j_2}+\omega_j} & \beta_{j_2}\frac{\lambda_{j_2}}{\lambda_{j_2}+\omega_j} \end{array}, \quad \mathbf{\Omega}_j = \begin{array}{c|cc} & 1 & M_j \\ 1 & \frac{\omega_j}{\lambda_{j_1}+\omega_j} & 0 \\ M_j & 0 & \frac{\omega_j}{\lambda_{j_2}+\omega_j} \end{array},$$

and $\mathbf{\Lambda}_j^{(1)} = \mathbf{0}$. In the case where $C_j^2 < 1$, the characterizing matrices are given by:

$$\mathbf{\Lambda}_j^{(1)} = \begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & M_j \\ 1 & 0 & \frac{\lambda_{j_1}}{\lambda_{j_1}+\omega_j} & \cdots & 0 & 0 \\ 2 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & 0 & 0 & \cdots & 0 & \frac{\lambda_{j_1}}{\lambda_{j_1}+\omega_j} \\ M_j & 0 & 0 & \cdots & 0 & 0 \end{array},$$

$$\mathbf{\Lambda}_j^{(2)} = \begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & M_j \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 2 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & 0 & 0 & \cdots & 0 & 0 \\ M_j & \frac{\lambda_{j_2}}{\lambda_{j_2}+\omega_j} & 0 & \cdots & 0 & 0 \end{array},$$

$$\mathbf{\Omega}_j = \begin{array}{c|ccccc} & 1 & 2 & \cdots & z_j & M_j \\ 1 & \frac{\omega_j}{\lambda_{j_1}+\omega_j} & 0 & \cdots & 0 & 0 \\ 2 & 0 & \frac{\omega_j}{\lambda_{j_1}+\omega_j} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ z_j & 0 & 0 & \cdots & \frac{\omega_j}{\lambda_{j_1}+\omega_j} & 0 \\ M_j & 0 & 0 & \cdots & 0 & \frac{\omega_j}{\lambda_{j_2}+\omega_j} \end{array}.$$

The queueing behavior observed by the DTMC X_j^* can best be described as a regenerative process. Customers arrive at a queue until the end of the vacation j (of length T_j) is reached, the queue empties and the process repeats itself. As such, the stationary distribution vector $\boldsymbol{\pi}^*[i|j]$ effectively captures the queueing process of only those customers that arrive during a vacation of type j (where $\boldsymbol{\pi}^*[i|j]$ is the vector of stationary probabilities associated with states $(i, c, v)_j : c \in \mathbf{M}_j \wedge v \in \{1, \ldots, V\}$).

The special structure observed in the transition matrix $\boldsymbol{\mathcal{P}}_j^*$ (that defines the DTMC X_j^*) corresponds to a $GI/M/1$ process. As such matrix analytical techniques can once more be used to efficiently derive performance measures. With respect to the DTMC X_j^* the only measure of interest is the average queue size $\mathcal{Q}_j^*$. The average queue size may be determined as a function of $\boldsymbol{\pi}^*[1|j]$ and $\mathbf{R}_j^*$; the vector of stationary probabilities of having a single customer in the queue and an auxiliary matrix respectively. As such we are not required to compute the entire stationary distribution of the DTMC X_j^*, yielding a significant gain in computational performance.

With respect to the DTMC X_j^*, $\mathbf{R}_j^*$ may be computed explicitly:

$$\mathbf{R}_j^* = \mathbf{F}_j^* \left(\mathbf{I} - \mathbf{L}_j^*\right)^{-1}, \tag{33}$$

and $\boldsymbol{\pi}^*[1|j]$ is obtained as follows:

$$\boldsymbol{\pi}^*[1|j] = \boldsymbol{\pi}^*[0|j]\mathbf{R}_j^*, \tag{34}$$

where $\boldsymbol{\pi}^*[0|j]$ is the solution of:

$$\begin{array}{lcl} \boldsymbol{\pi}^*[0|j] & = & \boldsymbol{\pi}^*[0|j] \left(\mathbf{I}\hat{\mathbf{L}}_j^* + \sum\limits_{n=1}^{\infty} \left(\mathbf{R}_j^*\right)^n \mathbf{B}_j^*\right), \\ 1 & = & \boldsymbol{\pi}^*[0|j] \left(\mathbf{I} - \mathbf{R}_j^*\right)^{-1} \mathbf{e}. \end{array} \tag{35}$$

The average queue size at the DTMC X_j^* is given by [8]:

$$\mathcal{Q}_j^* = \boldsymbol{\pi}^*[1|j] \left(\mathbf{I} - \mathbf{R}_j^*\right)^{-2} \mathbf{e}. \tag{36}$$

In the upcoming section, we demonstrate how to use these results in order to obtain general performance measures.

3.3 Aggregation of results

From the previous sections we obtained the average queue sizes: (1) at the start of a vacation of type j, after removal of k_j customers from the queue; (2) during a vacation of type j. We can combine these results to obtain the average waiting list size (defined as $\mathcal{Q}$) and the expected waiting time of a customer at the waiting list (defined as $\mathcal{W}$).

The average waiting list size is given by (note that $\mathcal{Q}_j^* = 0$ for all vacations $j : c_j \in \{1, 2\}$):

$$\mathcal{Q} = \sum_{j=1}^{J} p_j \left(\mathcal{Q}_j + \mathcal{Q}_j^*\right), \tag{37}$$

Table 1: Summary of the input parameters at the appointment-driven system

j	c_j	T_j	k_j	C_j^2	λ_{j_1}	λ_{j_2}	β_{j_1}	β_{j_2}
1	3	6	0	2.00	$1/6$	$2/3$	$1/3$	$2/3$
2	5	6	3	2.00	$1/6$	$2/3$	$1/3$	$2/3$
3	1	13	0					
4	4	5	1	0.68	$1/4$	1		
5	2	138	2					

where p_j is the probabilities of finding oneself at a vacation of type j:

$$p_j = \frac{T_j}{\sum_{j=1}^{J} T_j}. \tag{38}$$

Using Little's law, we can compute the expected waiting time of a customer at the CAS:

$$\mathcal{W} = \frac{\mathcal{Q}}{\sum_{j=1}^{J} \frac{\eta_j}{T_j}}, \tag{39}$$

where η_j is the expected number of arrivals during a vacation of type j and is computed as follows:

$$\eta_j = \sum_{i=0}^{\infty} \sum_{c \in \mathbf{M}_j} \sum_{d \in \mathbf{M}_j} i\psi[i, d|j, 0, c]. \tag{40}$$

4 Discussion and model performance

We illustrate model accuracy and computational performance by means of a numerical example. For this purpose, we revert to the setting discussed in section 2. More specifically, we assume an arrival cycle consisting of two arrival sessions (on Thursday and on Friday) and a service cycle of three service sessions (one on Thursday and two on Friday). The resulting vacation cycle consists of five vacations (i.e. $J = 5$) and customers are removed from the queue at the start of vacations $j : j \in \{2, 4, 5\}$. We assume arrivals to take place during vacations $j : j \in \{1, 2, 4\}$. The interarrival time distribution of customers at $j : j \in \{1, 2\}$ is highly variable and has mean and squared coefficient of variation $1/\lambda_j = 3$ and $C_j^2 = 2$ respectively. At vacation $j = 4$, customers arrivals are distributed with a mean $\lambda_j = 5$ and a squared coefficient of variation $C_j^2 = 0.68$. Using the equations developed in section 2.3 we obtain the respective PH distribution parameters. We summarize all key data in Table 1 (all time-related parameters are expressed in hours).

Using the CAS developed in this article, we assess waiting list performance measures of the example system. We use a simulation study to validate the model and to assess model

Table 2: Results of the DTMC X_j

j	$\mathcal{Q}_j$	Simulation
1	4.7401	4.7428
2	4.5235	4.5620
3	6.5528	6.5848
4	5.6633	5.6661
5	4.7401	4.7428

Table 3: Results of the DTMC X_j^*

j	$\mathcal{Q}_j^*$				Simulation
	$V = 10$	$V = 50$	$V = 100$	$V = 200$	
1	1.4017	1.3082	1.2961	1.2900	1.2838
2	1.1619	1.0498	1.0346	1.0269	1.0386
4	0.4260	0.3784	0.3720	0.3688	0.3655

accuracy. The simulation runlength (50.000.000 vacation cycles) ensures the accuracy of the simulation results.

We first discuss the results of the DTMC X_j (which monitors the number of customers in the queue at the start of a vacation of type j). More specifically, we have a look at the average queue size $\mathcal{Q}_j$ at each of the five vacations. The results are reported in Table 2. One may observe that the results of the CAS and the simulation model match almost seamlessly, which should not come as a surprise in view of the fact that the DTMC X_j provides numerically exact results.

With respect to the DTMC X_j^* (which observes the queueing behavior of customers arriving during a vacation of type j) we approximate the deterministic vacation lengths using an Erlang distribution of V phases. In order to assess the impact on model accuracy, we run the model using different values of V. The results (i.e. the average queue size $\mathcal{Q}_j^*$ at each of the vacations in which arrivals are allowed to occur) are reported in Table 3. It is clear that as V increases, the accuracy of the model increases as well. Even for small values of V (i.e. $V = 50$) accurate results are obtained. For large values of V, only small differences exist between the results of the CAS and the results of the simulation model.

When looking at the performance measures of the CAS in general, one may observe an expected number of customers in the waiting list that equals $\mathcal{Q} = 4.9941$ customers (simulation reports 4.9989 customers). The expected waiting time at the waiting list amounts to $\mathcal{W} = 158.25$ hours (simulation reports 157.53 hours).

With respect to computational performance, the algorithms are implemented in Visual C while all other computations are performed using the Matlab software package and the structured Markov chain solver developed by [2]. The computations are performed on an

AMD Athlon with 2.0 GHz CPU-speed and 768 MB of RAM. Computation times for the example amount to a few seconds (most of which are spent interfacing between the the Visual C implemented algorithms and the Matlab software package).

We can conclude that: (1) the CAS provides valid results; (2) the results obtained at the CAS are highly accurate; (3) computational performance allows the study of complex real-life problems.

5 Conclusion

In this article we focus on appointment-driven systems. In such systems, customers make an appointment, are issued an appointment date and receive service at some future service session. Appointment-driven systems are often characterized by a chronic backlog of customers to be served (i.e. the waiting list). We develop a model that allows the detailed analysis of waiting list performance measures. We refer to this model as the Customer Assignment System (CAS). The CAS may be considered as a complex vacation model that has various unique properties. In order to efficiently obtain the waiting list performance measures, we divide the CAS into two sets of Discrete Time Markov Chains (DTMC). Each set of DTMC requires a distinct modeling approach. The stationary distribution and other measures of interest of both sets of DTMC are computed by means of efficient algorithms and matrix analytical techniques.

The CAS developed in this article, is able to provide a very accurate analysis of waiting list performance measures. In addition, the computational performance indicates that real-life problems may be analyzed. Using these performance measures, strategically important issues may be addressed: (1) the trade-off between capacity and demand; (2) the distribution of capacity over time and space (e.g. when should a server be online and how many customers should be served at each of the service sessions); (3) the evaluation of appointment-booking practices (e.g. when and for how long should customers be allowed to make appointments).

Future research on CAS should focus on: (1) the time-dependent arrival of customers; (2) the correlated arrival of customers; (3) the use of more general PH approximations of the arrival process.

References

[1] Bini, D., Meini, B., Steffe, S., Van Houdt, B. Structured Markov chains solver: algorithms. ACM international conference proceeding series, proceedings of SMCtools, 2006.

[2] Bini, D., Meini, B., Steffe, S., Van Houdt, B. Structured Markov chains solver: software tools. ACM international conference proceeding series, proceedings of SMCtools, 2006.

[3] Cayirli, T., Veral, E. Outpatient scheduling in health care: a review of literature. Production and Operation Management 2003;12(4):519–549.

[4] Creemers, S., Lambrecht, M. Queueing models for appointment-driven systems. Annals of Operations Research 2008, to appear.

[5] Doshi, BT. Queueing systems with vacations - a survey. Queueing Systems 1986;1(1):29–66.

[6] Gail, HR., Hantler, SL., Taylor, BA. Non-skip-free M/G/1 and G/M/1 type Markov chains. Advances in Applied Probability 1997;29(3):733–758.

[7] Latouche, G., Ramaswami, V. Introduction to Matrix Analytic Methods in Stochastic Modeling. Philadelphia: ASA-SIAM Series on Statistics and Applied Probability; 1999.

[8] Nelson, R. Matrix geometric solutions in Markov models: a mathematical tutorial, IBM Research Report RC 16777, 1991.

[9] Neuts, MF. Matrix-geometric solutions in stochastic models. Baltimore: Johns Hopkins University Press; 1981.

[10] Osogami, T. Analysis of multiserver systems via dimensionality reduction of Markov chains. PhD thesis, Carnegie Mellon University, 2005.

[11] Ramaswami, V. A stable recursion for the steady state vector in Markov chains of M/G/1 type. Stochastic Models 1988;4(1):183–189.

[12] Riska, A. Aggregate matrix analytic techniques and their applications. PhD thesis, The College of William and Mary, 2002.

[13] Takagi, H. Queueing analysis of polling models. ACM Computing Surveys 1988;20(1):5–28.

[14] Tian, N., Zhang, Z. Vacation queueing models. New York: Springer Science; 2006.

[15] Tijms, HC. A first course in stochastic models. Chichester: John Wiley & Sons; 2003.