Applied Mathematics, 2012, 3, 1689-1696
http://dx.doi.org/10.4236/am.2012.311234 Published Online November 2012 (http://www.SciRP.org/journal/am)

Exact Distributions of Waiting Time Problems of Mixed Frequencies and Runs in Markov Dependent Trials

Bruce J. Chaderjian, Morteza Ebneshahrashoob*, Tangan Gao
Department of Mathematics and Statistics, California State University, Long Beach, USA
Email: *morteza.ebneshahrashoob@csulb.edu

Received September 7, 2012; revised October 7, 2012; accepted October 15, 2012

ABSTRACT

We study waiting time problems for first-order Markov dependent trials via conditional probability generating functions. Our models involve α frequency cells and β run cells with prescribed quotas and an additional γ slack cells without quotas. For any given $\bar{\alpha} \le \alpha$ and $\bar{\beta} \le \beta$, in our Model I we determine the waiting time until at least $\bar{\alpha}$ frequency cells and at least $\bar{\beta}$ run cells reach their quotas. For any given $\tau \le \alpha + \beta$, in our Model II we determine the waiting time until τ cells reach their quotas. Computer algorithms are developed to calculate the distributions, expectations and standard deviations of the waiting time random variables of the two models. Numerical results demonstrate the efficiency of the algorithms.

Keywords: Inverse Sampling; Multinomial Stopping Problem; Soonest through Latest Waiting Time Variable; Probability Generating Function; First-Order Markov Dependent Trial

1. Introduction

Over the past few decades numerous studies have been made concerning waiting time random variables with stopping rules involving frequencies, runs, and patterns (e.g., [1-3]). The book [1] provides a thorough overview of many waiting time problems and their applications up to 2001. The book [2] uses the finite Markov chain imbedding technique to deal with certain waiting time problems involving frequency, run, and pattern quotas. The compilation [3] contains papers that use various techniques to deal with waiting time problems and their applications. Sooner and later waiting time problems as well as Markov dependent trials are discussed in many articles (e.g., [4-8]).

A model which incorporates many specific models in the above research was proposed by [9] for independent multinomial trials. The Dirichlet methodology was used as a computational tool in [9], but in general the Dirichlet method is not computationally efficient. The main goal of this paper is to introduce two efficient algorithms which use conditional probability generating functions (pgf's) to solve certain generalizations of the model in [9] to the case of first-order Markov dependent trials.

The first-order Markov dependent (α, β, γ) models studied in this paper involve $\alpha + \beta + \gamma$ disjoint *cells*. Each cell tracks exclusively the number of occurrences of a specific outcome in a sequence of first-order Markov dependent trials. The first α cells are designated as *frequency* cells and are prescribed integer frequency quotas $f_1, \cdots, f_\alpha$. Each frequency cell tracks the total number of times (frequency count) that its associated outcome has occurred. The cell is said to have reached its quota if its frequency count has reached its prescribed quota value. The next β cells are *run* cells and are prescribed integer run quotas $r_1, \cdots, r_\beta$. Each run cell tracks the number of *consecutive* times (run count) that its associated outcome has occurred during the current run. A run cell is said to have reached its quota if its run count has reached its prescribed quota value. The last γ cells are *slack* cells that have no prescribed quotas. These cells may be used if some of the outcomes are not of interest for a specific experiment. For certain special cases, such as independent multinomial trials, the γ slack cells may be reduced into one single slack cell.

The models discussed in this paper are the following:

Model I: The scheme is to stop sampling when at least $\bar{\alpha}$ $(\le \alpha)$ frequency cells and at least $\bar{\beta}$ $(\le \beta)$ run cells have reached their given quotas. Let $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ denote the waiting time until at least $\bar{\alpha}$ frequency quotas and at least $\bar{\beta}$ run quotas have been reached.

Model II: The scheme is to stop sampling when any combination of τ $(\le \alpha + \beta)$ frequency or run cells have reached their given quotas. Let $WT_{(\alpha,\beta,\gamma)}^{\tau}$ denote

*Corresponding author.

the waiting time until a total of τ frequency or run quotas have been reached. This model includes all cases from the *soonest* ($\tau = 1$) through the *latest* ($\tau = \alpha + \beta$).

Our algorithms calculate the exact distributions, expectations, and standard deviations of $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ of Model I and $WT_{(\alpha,\beta,\gamma)}^{\tau}$ of Model II. Our work generalizes [9] in the following ways: 1) Model I uses stopping rules that distinguish between frequency quotas and run quotas as in [9], but for the case of first-order Markov dependent trials; 2) Model II introduces stopping rules that do not distinguish between frequency quotas and run quotas, again for first-order Markov dependent trials; 3) Although *specific* examples have been solved for the models in [9], we believe that our computer programs are the first that are capable of solving the *general* models; 4) Our models allow for multiple slack cells, whereas only one slack cell is necessary in the case of independent multinomial trials.

Various special cases of our models have been discussed in the literature. For example, $T_{k_1,k_2}^{(S)}$ and $T_{k_1,k_2}^{(L)}$ in Chapter 6 of [1] are the special cases of Models I and II with $\bar{\alpha} = \alpha = 0$, $\bar{\beta} = \beta = 2$, and no slack cell.

Remark 1: Due to the similarity of Model I and Model II, the algorithm for Model II can be adapted from that of Model I, and thus the details of the algorithm for Model II are omitted in this paper. Numerical results for Model II are presented in Section 4.

Recently, the use of sparse matrix computational methods applied to the pgf method has opened a new phase for the method as a computational tool for solving various problems (e.g., [10-12]). In Section 2, we briefly describe the pgf method for solving Model I. In Section 3, we outline the details of our algorithm for Model I. Numerical results for both Model I and Model II are presented in Section 4. Monte Carlo simulation algorithms are also developed for both models to demonstrate the efficiency of our algorithms.

2. PGF Method for Model I

For the first-order Markov dependent trials of Model I, let $\left(p_1,\cdots,p_\alpha,q_1,\cdots,q_\beta,o_1,\cdots,o_\gamma\right)$ be the initial probabilities that the first outcome occurs in the corresponding frequency, run, or slack cell, with $\sum_{i=1}^{\alpha}p_i+\sum_{j=1}^{\beta}q_j+\sum_{l=1}^{\gamma}o_l=1$. If the current outcome is in the *k*-th cell, $1\le k\le\alpha+\beta+\gamma$, let $\left(p_{k1},\cdots,p_{k\alpha},q_{k1},\cdots,q_{k\beta},o_{k1},\cdots,o_{k\gamma}\right)$ be the transition probabilities that the *next* outcome occurs in the corresponding cell, with $\sum_{i=1}^{\alpha}p_{ki}+\sum_{j=1}^{\beta}q_{kj}+\sum_{l=1}^{\beta}o_{kl}=1$.

We now describe the states of Model I.

Definition 1: Let $f_1,\cdots,f_\alpha$ and $r_1,\cdots,r_\beta$ be, respectively, the integer quotas prescribed to the α frequency cells and the β run cells of Model I. Suppose that in Model I the current outcome is in cell k, $1\le k\le\alpha+\beta+\gamma$, the current frequency counts are m_i, $0\le m_i\le f_i$ for $i=1,\cdots,\alpha$, and the current run counts are n_j, $0\le n_j\le r_j$ for $j=1,\cdots,\beta$. We denote this state by

$$\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]. \tag{1}$$

The initial state is denoted by $[\varnothing;0,\cdots,0;0,\cdots,0]$ with $\varnothing$ indicating "the initial state".

Definition 2: We define a frequency (or run) cell that has not reached its quota to be *incomplete*. If the cell has reached its quota we say it is *complete*. Given $\bar{\alpha}\le\alpha$ and $\bar{\beta}\le\beta$ in Model I, if a state *s* of Model I contains fewer than $\bar{\alpha}$ complete frequency cells or fewer than $\bar{\beta}$ complete run cells, we say *s* is an *incomplete state*. Otherwise, *s* is a *complete state*.

For simplicity, since the actual subsequent count becomes irrelevant in a complete cell, we use its prescribed quota value to represent a complete cell's subsequent count. For this reason we had in (1) that $0\le m_i\le f_i$ for $i=1,\cdots,\alpha$, and $0\le n_j\le r_j$ for $j=1,\cdots,\beta$. It will be seen in Section 3 that all non-initial states of Model I can be represented by a (possibly proper) subset of elements of the form (1).

Let $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$ denote the (unconditional) pgf of $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ at the initial state $[\varnothing;0,\cdots,0;0,\cdots,0]$ and $\phi\left(t\middle|k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right)$ denote the conditional pgf of $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ at the state $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$, where *t* is the parameter of the pgf's. If a pgf is expanded in a standard power series in *t*, say $\phi\left(t\middle|k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right)=a_0+a_1t+a_2t^2+a_3t^3+\cdots$, the coefficient a_n equals the probability that at least $\bar{\alpha}$ frequency quotas and at least $\bar{\beta}$ run quotas will be reached in *n* steps given that the experiment is currently at the state $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$ (see the Remark on page 464 in [12]). Therefore, the set of coefficients of the power series of $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$ gives exactly the probability distribution of the waiting time random variable $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ that we wish to solve for.

The system of equations for the pgf's of Model I comes from Equations (2) and (3) with the boundary conditions in (4) applied. Equations (2) and (3) are based on the well-known total probability formula and the boundary conditions (4) simply mean that a pgf is constant when at least $\bar{\alpha}$ frequency quotas and at least $\bar{\beta}$ run quotas are satisfied.

Beginning with the initial state we have

$$\begin{aligned}&\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)\\&=p_1t\phi\left(t\middle|1;1,0,\cdots,0;0,\cdots,0\right)+\cdots\\&\quad+p_\alpha t\phi\left(t\middle|\alpha;0,\cdots,0,1;0,\cdots,0\right)\\&\quad+q_1t\phi\left(t\middle|\alpha+1;0,\cdots,0;1,0,\cdots,0\right)+\cdots\\&\quad+q_\beta t\phi\left(t\middle|\alpha+\beta;0,\cdots,0;0,\cdots,0,1\right)\\&\quad+o_1t\phi\left(t\middle|\alpha+\beta+1;0,\cdots,0;0,\cdots,0\right)+\cdots\\&\quad+o_\gamma t\phi\left(t\middle|\alpha+\beta+\gamma;0,\cdots,0;0,\cdots,0\right).\end{aligned}\tag{2}$$

To develop a similar equation for the other incomplete states, observe that the count in a run cell is 0 if and only if both the cell is incomplete and the current outcome is not in that run cell. Observe also, from our earlier conventions, that no cell can have a count that exceeds its quota. Let $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$ be any non-initial incomplete state. Bearing in mind our above observations, for $i=1,\cdots,\alpha$, define $M_i\equiv\min\left\{m_i+1,\,f_i\right\}$, and for $j=1,\cdots,\beta$, define $N_j\equiv\min\left\{n_j+1,r_j\right\}$, and define $\tilde{n}_j\equiv r_j$ if $n_j=r_j$ (the j-th run cell is complete) and $\tilde{n}_j\equiv 0$ otherwise. We then have

$$\begin{aligned}&\phi\left(t\middle|k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right)\\&=p_{k1}t\phi\left(t\middle|1;M_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right)+\cdots\\&\quad+p_{k\alpha}t\phi\left(t\middle|\alpha;m_1,\cdots,M_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right)\\&\quad+q_{k1}t\phi\left(t\middle|\alpha+1;m_1,\cdots,m_\alpha;N_1,\tilde{n}_2,\cdots,\tilde{n}_\beta\right)+\cdots\\&\quad+q_{k\beta}t\phi\left(t\middle|\alpha+\beta;m_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_{\beta-1},N_\beta\right)\\&\quad+o_{k1}t\phi\left(t\middle|\alpha+\beta+1;m_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right)+\cdots\\&\quad+o_{k\gamma}t\phi\left(t\middle|\alpha+\beta+\gamma;m_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right).\end{aligned}\tag{3}$$

The boundary conditions which correspond to constant pgf's are defined by

$$\phi\left(t\middle|k;m_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right)\equiv 1\tag{4}$$

if $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$ is a complete state.

Let N be the total number of non-constant pgf's of Model I (or the total number of equations in (2) and (3)). We will see in Section 3.3 how to calculate the value of N by (17). Let $\Phi(t)$ be the N-dimensional vector of the N non-constant pgf's arranged in a prescribed order with $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$ as its first entry. Then the system of equations in (2) and (3) with the boundary conditions (4) applied can be written as

$$\Phi(t)=tA\Phi(t)+tb\tag{5}$$

where $A\in\mathbf{R}^{N\times N}$ is a constant matrix whose nonzero entries are the initial or transition cell probabilities (coefficients of the non-constant pgf's) and $b\in\mathbf{R}^N$ is a constant vector made up from sums of cell probabilities (from the coefficients of the constant pgf's).

It is well-known (e.g., Theorem 3.4.1 in [13]) that

$$\begin{aligned}&\phi^{(k)}\left(0\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)=k!P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right),\\&k=0,1,2,\cdots,\end{aligned}\tag{6}$$

where the left-hand side is the k-th derivative of $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$ at $t=0$. Note that $\Phi(0)=0$. By repeatedly taking derivatives in (5), we have

$$\Phi'(0)=b\quad\text{and}\quad\Phi^{(k)}(0)=kA\Phi^{(k-1)}(0),\quad k=2,3,\cdots$$

and thus

$$\Phi^{(k)}(0)=k!A^{k-1}b,\quad k=1,2,3,\cdots.\tag{7}$$

Since the pgf $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$ is the first entry of the vector $\Phi(t)$, by (6) and (7),

$$\begin{aligned}&P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=0\right)=0\\&P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right)=\text{the first component of}\\&A^{k-1}b,\text{for all } k=1,2,\cdots.\end{aligned}\tag{8}$$

Instead of obtaining symbolically the pgf of $\phi\left(t\middle|\varnothing;0,\cdots,0;0,\cdots,0\right)$, our algorithm uses the simple formula (8) to calculate the exact distribution of the waiting time variable $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$. Therefore, the primary focus of our algorithm is to efficiently generate A and b. The details of how we generate A and b will be discussed in Section 3.

Since the matrix A is very sparse with each row having no more than $\alpha+\beta+\gamma$ nonzero entries, the calculation of Ab involves no more than $N\times(\alpha+\beta+\gamma)$ multiplications of real numbers. Since A^kb can be calculated from $A\left(A^{k-1}b\right)$ and $P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right)$ equals the first component of $A^{k-1}b$, the calculation of $P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right)$ for all $k=0,1,\cdots,n$ (*i.e.*, $P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}\le n\right)$) involves no more than $N\times(n-1)\times(\alpha+\beta+\gamma)$ multiplications of real numbers. By the nature of the problem, it can be shown that the spectral radius $\rho(A)$ of the matrix A is less than 1 which ensures the stability of calculating A^kb, $k=1,\cdots,n-1$.

3. Generating *A* and *b* for Model I

In this section we will discuss how to efficiently generate the matrix A and the vector b in (5). To do this, we will generate and order the initial state $\left[\varnothing;0,\cdots,0;0,\cdots,0\right]$ and all incomplete states of Model I of the form

$\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$ in (1) which correspond to the non-constant pgf's at the left-hand sides of the equations in (2) and (3). (Recall that $0\le m_i\le f_i$ for $i=1,\cdots,\alpha$, $0\le n_j\le r_j$ for $j=1,\cdots,\beta$, and the current event occurs in the *k*-th cell for some *k*, $1\le k\le\alpha+\beta+\gamma$.) Our process will first generate all necessary arrangements $\left[m_1,\cdots,m_\alpha\right]$, called *frequency states*, and all necessary arrangements $\left[n_1,\cdots,n_\beta\right]$, called *run states*, for the states in (1). Then the frequency states, run states, and possible values of *k* will be combined to form all incomplete states of Model I.

3.1. Generating Frequency States

The efficiency of our algorithm ultimately depends on its ability to identify the element in $\Phi(t)$ in (5) that corresponds to each state. This efficiency is facilitated by the ordering of the elements in $\Phi(t)$. In this section we will generate and order all the frequency states $\left[m_1,\cdots,m_\alpha\right]$ needed to construct the states in Model I.

The frequency states are first grouped into disjoint sets whose elements have in common precisely the same complete cells. Each set corresponds to exactly one binary *base vector* $(v_1,\cdots,v_\alpha)$ in which if $\left[m_1,\cdots,m_\alpha\right]$ is a frequency state in the set, then $v_i=1$ if $m_i=f_i$ (a complete cell) and $v_i=0$ otherwise.

To generate all the frequency states, we first generate all the base vectors necessary to form a one-to-one correspondence between the base vectors and the sets of frequency states. By the nature of Model I, once the goal of reaching $\bar{\alpha}$ frequency quotas has been achieved, the actual subsequent counts become irrelevant in the frequency cells. All frequency states containing at least $\bar{\alpha}$ complete cells are thus reduced to a single frequency state representing "at least $\bar{\alpha}$ frequency quotas reached". For simplicity, we use $\left[f_1,\cdots,f_\alpha\right]$ to denote this frequency state and we associate it with the base vector $(1,\cdots,1)$. Thus, only the base vectors which have less than $\bar{\alpha}$ 1's and the base vector $(1,\cdots,1)$ are needed for Model I and there are

$$\binom{\alpha}{0}+\binom{\alpha}{1}+\cdots+\binom{\alpha}{\bar{\alpha}-1}+1$$

such base vectors.

We now order the base vectors, followed by an ordering of the frequency states associated with each base vector. The base vectors are grouped according to their number of 1's. The groups themselves are then numbered and arranged in ascending order according to the number of 1's present in each of the base vectors within the groups. The base vectors within each group are then arranged by the lexicographic order. For example, from the leftmost column of Example 1 in Section 3.2 with $\bar{\alpha}=\alpha=2$, *Group* $0\equiv\{(0,0)\}$ (only the zero vector), *Group* $1\equiv\{(0,1),(1,0)\}$ *in this order*, and *Group* $2\equiv\{(1,1)\}$. As a second example, for $\alpha=4$ and $\bar{\alpha}>2$, *Group* 2 contains $\binom{4}{2}=6$ base vectors which contain exactly two 1's. The six base vectors in *Group* 2 have the lexicographic order $(0,0,1,1)$, $(0,1,0,1)$, $(0,1,1,0)$, $(1,0,0,1)$, $(1,0,1,0)$, and $(1,1,0,0)$. Let V_f be the vector containing all the necessary base vectors of the frequency states, arranged in the order just described. Standard back-tracking techniques are used to generate V_f.

We now generate and order the frequency states. Let $v=(v_1,\cdots,v_\alpha)$ be a given base vector, $v\in V_f$. Let S_f^v be the set of all frequency states associated with v. Note that the frequency states $\left[m_1,\cdots,m_\alpha\right]$ in S_f^v all satisfy $0\le m_i<f_i$ if $v_i=0$ and $m_i=f_i$ if $v_i=1$. We order these frequency states by the lexicographic order of their values m_i for which $m_i<f_i$ (and thus the complete cells with $v_i=1$ have no role in the lexicographic order). Standard back-tracking techniques are used to generate all frequency states in S_f^v in the lexicographic order just described. In the same way, we generate all the frequency states of Model I by repeating this generating process as we proceed through V_f sequentially to each base vector in V_f. Let *FS* be the vector of all frequency states arranged in the order in which they were generated.

As an example of our ordering of the frequency states, see the "*FS*" column of Example 1 in Section 3.2.

Let $v=(v_1,\cdots,v_\alpha)$, $v\in V_f$, with not all $v_i=1$. The *local position* of a given frequency state $\left[m_1,\cdots,m_\alpha\right]$ in S_f^v can be calculated by

$$\sum_{j=1:v_j=0,m_j>0}^{l-1}\left(m_j\prod_{i=j+1:v_i=0}^{l}f_i\right)+(m_l+1),\qquad(9)$$

where l is the largest index with $v_l=0$. There is a total of $\prod_{i=1:v_i=0}^{\alpha}f_i$ frequency states in S_f^v.

Similarly, the vector *FS* of all frequency states contains a total of

$$N_f=\sum_{v\in V_f}\prod_{i=1:v_i=0}^{\alpha}f_i$$

frequency states, where we adopt the convention that $\prod_{i=1:v_i=0}^{\alpha}f_i=1$ when $v=(1,\cdots,1)$ (which corresponds to the single frequency state $\left[f_1,\cdots,f_\alpha\right]$). Thus,

$$FS=\left(FS_1,FS_2,\cdots,FS_{N_f}\right).\qquad(10)$$

For any frequency state $\left[m_1,\cdots,m_\alpha\right]\ne\left[f_1,\cdots,f_\alpha\right]$, its

global position in the vector FS can be calculated by

$$\sum_{v\in V_f:\, v\prec\bar{v}}\prod_{i=1:\bar{v}_i=0}^{\alpha} f_i + \sum_{j=1:\bar{v}_j=0,m_j>0}^{l-1}\left(m_j \prod_{i=j+1:\bar{v}_i=0}^{l} f_i\right) \quad (11)$$
$$+(m_l+1),$$

where $\bar{v}=(\bar{v}_1,\cdots,\bar{v}_\alpha)$ is the base vector associated with $[m_1,\cdots,m_\alpha]$, $v\prec\bar{v}$ means that the base vector $v=(v_1,\cdots,v_\alpha)$ precedes $\bar{v}$ in the vector V_f, and l is the largest index with $\bar{v}_l=0$. The second part of this formula is from (9). The frequency state $[f_1,\cdots,f_\alpha]$ is naturally at the last position N_f in FS.

Our ordering of the frequency states and the validity of Equations (9) and (11) are illustrated in the four leftmost columns of Example 1 in Section 3.2.

3.2. Generating Run States

All necessary run states of Model I can be generated and ordered similarly to the frequency states. For run states $[n_1,\cdots,n_\beta]$, base vectors $(v_1,\cdots,v_\beta)$ are defined by $v_j=1$ if $n_j=r_j$ (a complete cell) and $v_j=1$ otherwise. Thus, each base vector corresponds to a set of run states which have in common precisely the same complete run cells. Once $\bar{\beta}$ run cells have reached their given quotas, all subsequent run states are reduced to one single state representing "at least $\bar{\beta}$ run quotas reached". This run state, denoted by $[r_1,\cdots,r_\beta]$, is associated with the base vector $(1,\cdots,1)$. Thus, there are

$$\binom{\beta}{0}+\binom{\beta}{1}+\cdots+\binom{\beta}{\bar{\beta}-1}+1$$

base vectors needed to generate the run states of Model I. Let V_r be the vector containing these base vectors arranged in the same manner as the base vectors in Section 3.1, *i.e.* they are collected into groups which are arranged in ascending order of their number of 1's, and then lexicographically ordered within their group.

To facilitate the description of our ordering of the run states, we make the following definitions:

Definition 3: Let $rs=[n_1,\cdots,n_\beta]$ be a given run state. The j-th run cell for the state rs is called *active* if its current run count n_j satisfies $0<n_j<r_j$; otherwise, the run cell is *inactive* ($n_j=0$ or $n_j=r_j$). If rs contains an active run cell, rs is an *active run state*. Otherwise, rs is *inactive*.

Let $v=(v_1,\cdots,v_\beta)$ be a given base vector, $v\in V_r$, and let S_r^v be the set of run states $[n_1,\cdots,n_\beta]$ associated with v. Note that since no more than one run can be in progress at any one time, exactly one run cell is active in an active run state. Also, recall that once a run cell becomes complete (has reached its quota) its run count is fixed at its quota value. Thus, the only inactive run state in S_r^v is given by $n_j=0$ if $v_j=0$ and $n_j=r_j$ if $v_j=1$. All other run states in S_r^v are active (with one active cell). The run states in S_r^v are arranged in the lexicographic order of all the values $n_j<r_j$ for all the values of j of the incomplete run cells (and thus the complete run cells with $v_j=1$ have no role in the lexicographic order). Note that the first run state in this arrangement is the inactive run state in S_r^v. For any given active run state $[n_1,\cdots,n_\beta]$ in S_r^v, its *local position* in S_r^v can be calculated by

$$1+n_{i_a}+\sum_{j=i_a+1:v_j=0}^{\beta}(r_j-1), \quad (12)$$

where i_a is the index of the active run cell. There is a total of $1+\sum_{j=1:v_j=0}^{\beta}(r_j-1)$ run states in S_r^v. Standard back-tracking techniques are used to generate all run states in S_r^v in the lexicographic order described above.

In the same way as in Section 3.1, we generate all the run states of Model I by repeating the above generating process as we proceed through V_r sequentially to each base vector in V_r. Let RS be the vector of all run states arranged in the order in which they are generated.

It can be verified that RS contains a total of

$$N_r=\sum_{v\in V_r}\left(1+\sum_{j=1:v_j=0}^{\beta}(r_j-1)\right)$$

run states. These are the run states needed for generating all the states of Model I. We have

$$RS=(RS_1,RS_2,\ldots,RS_{N_r}). \quad (13)$$

Note that for the last run state in RS, $[r_1,\cdots,r_\beta]$, associated with $v=(1,\cdots,1)$, the second sum in the formula above for N_r is zero vacuously. It can be verified that for any run state $[n_1,\cdots,n_\beta]$, its *global position* in RS can be calculated by

$$\sum_{v\in V_r:\, v\prec\bar{v}}\left(1+\sum_{j=1:v_j=0}^{\beta}(r_j-1)\right)+1, \quad (14)$$

if $[n_1,\cdots,n_\beta]$ is inactive, or

$$\sum_{v\in V_r:\, v\prec\bar{v}}\left(1+\sum_{j=1:v_j=0}^{\beta}(r_j-1)\right) + \left(1+n_{i_a}+\sum_{j=i_a+1:\bar{v}_j=0}^{\beta}(r_j-1)\right) \quad (15)$$

if $[n_1,\cdots,n_\beta]$ is active, where $\bar{v}=(\bar{v}_1,\cdots,\bar{v}_\beta)$ is the base base vector associated with $[n_1,\cdots,n_\beta]$, $v\prec\bar{v}$ means

that the base vector $v=(v_1,\cdots,v_\beta)$ precedes $\bar{v}$ in the vector V_r, and i_a is the index of the active run cell in $[n_1,\cdots,n_\beta]$. The second part of the sum in (15) is from (12).

Our ordering for both frequency and run states and the validity of the Equations (9), (11), (12), (14), and (15) to identify their positions in *FS* and *RS* are illustrated in Example 1 below.

Example 1: Let $\alpha=2$, $\beta=3$, $\bar{\alpha}=\bar{\beta}=2$, $(f_1,f_2)=(2,3)$, and $(r_1,r_2,r_3)=(2,3,2)$. The results discussed in Sections 3.1 and 3.2 can be summarized in the following table:

For frequency cells				For run cells			
V_f	*FS*	L	G	V_r	*RS*	L	G
(0,0)	[0,0]	1	1	(0,0,0)	[0,0,0]	1	1
	[0,1]	2	2		[0,0,1]	2	2
	[0,2]	3	3		[0,1,0]	3	3
	[1,0]	4	4		[0,2,0]	4	4
	[1,1]	5	5		[1,0,0]	5	5
	[1,2]	6	6	(0,0,1)	[0,0,2]	1	6
(0,1)	(0,3)	1	7		[0,1,2]	2	7
	[1,3]	2	8		[0,2,2]	3	8
(1,0)	(2,0)	1	9		[1,0,2]	4	9
	[2,1]	2	10	(0,1,0)	[0,3,0]	1	10
	[2,2]	3	11		[0,3,1]	2	11
(1,1)	[2,3]	1	12		[1,3,0]	3	12
				(1,0,0)	[2,0,0]	1	13
					[2,0,1]	2	14
					[2,1,0]	3	15
					[2,2,0]	4	16
				(1,1,1)	[2,3,2]	1	17

where columns "V_f" and "V_r" contain the necessary base vectors for the frequency states in column "*FS*" and the run states in column "*RS*" respectively, the values in the columns "L" and "G" are the local positions (within the set of frequency or run states associated with the same base vector) and the global positions in *FS* or *RS* of the corresponding frequency states or run states according to the formulas (9), (11), (12), (14), and (15).

For example, consider the run state $[2,1,0]$ in Example 1. Its local position in the group associated with the base vector $(1,0,0)$ is 3. The combined count from the base vector groups $(0,0,0)$, $(0,0,1)$, and $(0,1,0)$ that precede the base vector group $(1,0,0)$ is 12. Thus, the global position of the run state $[2,1,0]$ in vector *RS* is $G=3+12=15$. Note that the run state $[r_1,r_2,r_3]=[2,3,2]$ represents "at least $\bar{\beta}=2$ run quotas reached" and thus the run base vector $(1,1,1)$ also represents the base vectors $(0,1,1),(1,0,1)$, and $(1,1,0)$.

3.3. Generating All States for Model I

A given frequency state $FS_i=[m_1,\cdots,m_\alpha]$ in (10) and a given run state $RS_j=[n_1,\cdots,n_\beta]$ in (13) can be combined to form a group of states of Model I of the form

$$[k;FS_i;RS_j]=[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta] \qquad (16)$$

as in (1), where the current outcome occurs in the k-th cell for some k, $1\le k\le\alpha+\beta+1$. However, we will see that only a subset of values of k taken from $\{1,2,\cdots,\alpha+\beta+\gamma\}$ are possible in (16). For the fixed pair $[FS_i;RS_j]$, let $N[FS_i;RS_j]$ be the number of states in the group (16), *i.e.* the number of possible values of *k*. Let $Nnz[FS_i]$ be the number of nonzero entries in $[m_1,\cdots,m_\alpha]$ and $Nc[n_1,\cdots,n_\beta]$ be the number of complete cells in $[n_1,\cdots,n_\beta]$. If RS_j is an active run state, $N[FS_i;RS_j]=1$ since the current outcome must be in the active run cell of RS_j, allowing only one value for k in (16). If RS_j is inactive, the current outcome can occur in any nonzero frequency cell in FS_i, any complete run cell in RS_j, or any slack cell. Therefore, if RS_j is inactive, (16) represents a group of $N[FS_i;RS_j]=Nnz[FS_i]+Nc[RS_j]+\gamma$ different states of Model I. We arrange the states in this group in ascending order of the values of *k*.

In addition to the initial state $[\varnothing;0,\cdots,0;0,\cdots,0]$, we generate all other incomplete states of Model I by

```
for i = 1,···,N_f
   for j = 1,···,N_r
      generate states in (16) and
      arrange them in ascending order of the values of k
   end
end
```

but we exclude the combining of the frequency state $FS_{N_f}=[f_1,\cdots,f_\alpha]$ and the run state $RS_{N_r}=[r_1,\cdots,r_\beta]$ which corresponds to the complete state "at least $\bar{\alpha}$ frequency quotas reached and at least $\bar{\beta}$ run quotas reached". The complete state corresponds to the constant pgf in (4) and are not part of the vector $\Phi(t)$ in (5).

Let $S=(S_1,\cdots,S_N)$ be the vector of all incomplete states of Model I arranged in the order they are generated above but preceded by the initial state $[\varnothing;0,\cdots,0;0,\cdots,0]$ as its first entry. Note that the initial state is the only state with $k=\varnothing$ since it has no current outcome. The initial state is immediately followed in *S* by the group of states $[k;0,\cdots,0;0,\cdots,0]$ for

$k=\alpha+\beta+1,\cdots,\alpha+\beta+\gamma$. The total number of incomplete states for Model I is given by

$$N=1+\sum_{i=1}^{N_f-1}\sum_{j=1}^{N_r}N\left[FS_i;RS_j\right]+\sum_{j=1}^{N_r-1}N\left[FS_{N_f};RS_j\right], \quad (17)$$

where the leading 1 corresponds to the initial state.

For any given state $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]\in S$, let i_1 be the position of $\left[m_1,\cdots,m_\alpha\right]$ in the vector FS determined by (11) with the exception that $i_1=N_f$ if $\left[m_1,\cdots,m_\alpha\right]=\left[f_1,\cdots,f_\alpha\right]$; let j_1 be the position of $\left[n_1,\cdots,n_\beta\right]$ in the vector RS determined by (14) or (15); and let l_1 be the local position of $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$, by ascending order on k, in the group (16). The position of $\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$ in the vector S can be determined by

$$1+\sum_{i=1}^{i_1-1}\sum_{j=1}^{N_r}N\left[FS_i;RS_j\right]+\sum_{j=1}^{j_1-1}N\left[FS_{i_1};RS_j\right]+l_1. \quad (18)$$

This formula is extremely useful when we generate the matrix *A* and the vector *b*.

3.4. Generating *A* and *b*

The matrix *A* and the vector *b* in (5) are initialized to zero. For each non-initial state S_i $(i>1)$ in $S=\left(S_1,\cdots,S_N\right)$, say $S_i=\left[k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right]$, the i-th row of *A* and element $b(i)$ of *b* will be determined as follows: From the equation for $\phi\left(t\middle|k;m_1,\cdots,m_\alpha;n_1,\cdots,n_\beta\right)$ in (3), if $\phi\left(t\middle|1;M_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right)$ is a constant pgf according to the boundary conditions (4), then the value of $b(i)$ is increased by p_{k1}; otherwise $A(i,j)=p_{k1}$ where j is the position of the state $\left[1;M_1,\cdots,m_\alpha;\tilde{n}_1,\cdots,\tilde{n}_\beta\right]$ in *S* as determined by (18). All other pgf's on the right-hand side of (3) are then similarly processed to complete the i-th row of *A* and $b(i)$. If more than one constant pgf is present on the right-hand side of (3), $b(i)$ equals the sum of the probabilities of the cells corresponding to the constant pgf's. Similarly, the first row of *A* and $b(1)$ are determined from (2). The matrix *A* is stored in sparse matrix format for our computer program (e.g., Section 3.4 of [14]).

Remark 2: The states used in our algorithm are sufficient and necessary for solving the general Model I. For special cases (e.g., independent multinomial trials) certain groups of states in our algorithm can be reduced to a single state, further enhancing the efficiency of the algorithm.

4. Numerical Results

Our computer program for Model I is in C++ and is based on the methods discussed in Sections 2 and 3 for calculating the distribution, expectation, and standard deviation of the waiting time variable $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$. Similar computer program for Model II is also developed. The programs have been successfully implemented and tested with various combinations of the parameters α, β, γ, $f_i's$, $r_j's$, $\bar{\alpha}$, $\bar{\beta}$, τ and various probabilities. Monte Carlo simulation algorithms for both models were also developed for comparison to the pgf method.

Example 2: Consider tossing one fair die initially. In every subsequent trial, we toss one of six unfair dice labeled 1 through 6. The die which is selected for the next trial matches the count on the face of the die from the current trial. Suppose we are looking for frequency quotas of 20 and 21 for faces 1 and 2, run quotas of 3 and 4 for faces 2 and 3, and faces 5 and 6 are considered slack cells. The initial cell probabilities (tossing a fair die) are $(1/6,1/6,1/6,1/6,1/6,1/6)$ and the transition cell probabilities (tossing one of six unfair dice) are

$$(2/12,3/12,3/12,1/12,2/12,1/12)$$
$$(3/12,3/12,1/12,2/12,2/12,1/12)$$
$$(1/12,2/12,2/12,3/12,3/12,1/12)$$
$$(2/12,2/12,3/12,3/12,1/12,1/12)$$
$$(3/12,1/12,2/12,2/12,3/12,1/12)$$
$$(1/12,2/12,2/12,3/12,3/12,1/12)$$

In Example 2 $\alpha=\beta=\gamma=2$. For the waiting time variables $WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}$ and $WT_{(\alpha,\beta,\gamma)}^{\tau}$, **Table 1** lists the expectations (E), standard deviations (sd), sizes of the matrix *A* (*N*), and computation times (CPU, "m" stands for minutes and "s" for seconds) produced by the pgf method and those produced by the Monte Carlo method with 1,000,000 simulations for each possible combination of $(\bar{\alpha},\bar{\beta})$ and each possible value of τ. The numerical values are rounded to two digits after the decimal point. The algorithm for the pgf method was terminated when, say for Model I, the condition $1-\sum_{k=0}^{n}P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right)<10^{-8}$ was satisfied for some $n>1$ since $P\left(WT_{(\alpha,\beta,\gamma)}^{(\bar{\alpha},\bar{\beta})}=k\right)$ is much smaller than 10^{-8} for $k=n+1,n+2,\cdots$. Computation of the results was carried out on a 3.6 GHz Intel Xeon Pentium IV with 2 Gb memory running RedHat Enterprise Linux operating system.

The case $(\bar{\alpha},\bar{\beta})=(\alpha,\beta)$ for Model I and the case $\tau=\alpha+\beta$ for Model II are mathematically identical which is reflected in the results of the pgf method in

Table 1. Expectations and standard deviations for example 2.

	Model I: pgf Method				Monte Carlo Method		
$(\bar{\alpha}, \bar{\beta})$	E	sd	*N*	CPU	E	sd	CPU
(1,1)	555.74	537.61	6228	3.8 s	556.01	538.88	44.9 s
(2,1)	561.59	532.82	6839	4.3 s	561.33	531.93	45.4 s
(1,2)	1767.33	1382.43	12,461	21.8 s	1764.83	1383.14	2 m 25 s
(2,2)	1767.63	1382.08	13,683	22.8 s	1765.10	1379.56	2 m 26 s
	Model II: pgf Method				**Monte Carlo Method**		
τ	E	sd	*N*	CPU	E	sd	CPU
1	87.53	24.37	3740	0.05 s	87.56	24.31	7.4 s
2	120.86	20.96	10,325	0.2 s	120.82	21.03	10.2 s
3	561.28	533.07	13,424	8.5 s	561.23	533.88	47.2 s
4	1767.63	1382.08	13,683	22.6 s	1764.49	1380.89	2 m 31 s

Table 1. Note that the sizes *N* of the *A* matrices are quite large in the pgf method, but the matrices are extremely sparse. For example, the size of the matrix *A* is $N = 13,683$ for the parameter $\tau = 4$ of Model II in the table. A dense matrix with this size is already too large to be handled by the computer used for the calculation in this section. But, by utilizing the sparsity of the matrices, our algorithm can efficiently solve the problem within 23 seconds and our algorithm is more than six times faster than the Monte Carlo Method. Moreover, our algorithms obtain their results by direct solution rather than by estimation based on simulations. To our knowledge, such direct solution methods were not previously available for the general Model I or Model II. The results demonstrate that our algorithms are efficient compared to the Monte Carlo simulation method.

REFERENCES

[1] N. Balakrishnan and M. V. Koutras, "Runs and Scans with Applications," John Wiley & Sons, New York, 2002.

[2] J. C. Fu and W. Y. Lou, "Distribution Theory of Runs and Patterns and its Applications," World Scientific Publisher, Singapore City, 2003.

[3] A. P. Godbole and S. G. Papastavridis, "Runs and Patterns in Probability: Selected Papers," Kluwer, Dordrecht, 1994. doi:10.1007/978-1-4613-3635-8

[4] S. Aki and K. Hirano, "Sooner and Later Waiting Time Problems for Runs in Markov Dependent Bivariate Trials," *Annals of the Institute of Statistical Mathematics*, Vol. 51, No. 1, 1999, pp. 17-29. doi:10.1023/A:1003874900507

[5] K. Balasubramanian, R. Viveros and N. Balakrishnan, "Sooner and Later Waiting Time Problems for Markovian Bernoulli Trials," *Statistics & Probability Letters*, Vol. 18, No. 2, 1993, pp. 153-161. doi:10.1016/0167-7152(93)90184-K

[6] Q. Han and S. Aki, "Waiting Time Problems in a Two-State Markov Chain," *Annals of the Institute of Statistical Mathematics*, Vol. 52, No. 4, 2000, pp. 778-789. doi:10.1023/A:1017537629251

[7] N. Kolev and L. Minkova, "Run and Frequency Quotas in a Multi-State Markov Chain," *Communications in Statistics—Theory and Methods*, Vol. 28, No. 9, 1999, pp. 2223-2233. doi:10.1080/03610929908832417

[8] K. D. Ling and T. Y. Low, "On the Soonest and Latest Waiting Time Distributions: Succession Quotas," *Communications in Statistics—Theory and Methods*, Vol. 22, No. 8, 1993, pp. 2207-2221. doi:10.1080/03610929308831143

[9] M. Sobel and M. Ebneshahrashoob, "Quota Sampling for Multinomial via Dirichlet," *Journal of Statistical Planning and Inference*, Vol. 33, No. 2, 1992, pp. 157-164. doi:10.1016/0378-3758(92)90063-X

[10] M. Ebneshahrashoob, T. Gao and M. Sobel, "Double Window Acceptance Sampling," *Naval Research Logistics* (*NRL*), Vol. 51, No. 2, 2004, pp. 297-306. doi:10.1002/nav.10119

[11] M. Ebneshahrashoob, T. Gao and M. Sobel, "Sequential Window Problems," *Sequential Analysis*: *Design Methods and Applications*, Vol. 24, No. 2, 2005, pp. 159-175. doi:10.1081/SQA-200056194

[12] M. Ebneshahrashoob, T. Gao and M. Wu, "An Efficient Algorithm for Exact Distribution of Scan Statistics," *Methodology and Computing in Applied Probability*, Vol. 7, No. 4, 2005, pp. 459-471. doi:10.1007/s11009-005-5003-0

[13] M. J. Evans and J. S. Rosenthal, "Probability and Statistics, the Science of Uncertainty," W. H. Freeman and Company, New York, 2004.

[14] Y. Saad, "Iterative Methods for Sparse Linear Systems," SIAM: Society for Industrial and Applied Mathematics, Philadelphia, 2003. doi:10.1137/1.9780898718003

