COMMUNICATIONS IN INFORMATION AND SYSTEMS
Vol. 4, No. 2, pp. 135-164, March 2005

THROUGHPUT OF Q-ARY SPLITTING ALGORITHMS FOR CONTENTION RESOLUTION IN COMMUNICATION NETWORKS[*]

B. VAN HOUDT[†] AND C. BLONDIA[†]

Abstract. The throughput characteristics of contention-based random access channels which use Q-ary splitting algorithms (where Q is the number of groups into which colliding users are split) are analyzed. The algorithms considered are of the Capetanakis-Tsybakov-Mikhailov-Vvedenskaya (CTMV) type and are studied for infinite populations of identical users generating packets according to a discrete time batch Markovian arrival process (D-BMAP). D-BMAPs are a class of tractable Markovian arrival processes, which, in general, are non-renewal. Free channel-access is assumed in combination with Q-ary collision resolution algorithms that exploit either binary or ternary feedback. For the resulting schemes, tree structured Quasi-Birth-Death (QBD) Markov chains are constructed and their stability is determined. The maximum achievable throughput is determined for a variety of arrival processes and splitting factors Q. It is concluded that binary ($Q = 2$) and ternary ($Q = 3$) algorithms should be preferred above other splitting factors Q as the throughput for $Q > 3$ quickly degrades when subject to bursty arrival streams. If packets arrivals are correlated and bursty, higher throughput rates can be achieved by making use of biased coins.

Key words: Random access algorithms, contention resolution, tree algorithms, batch Markovian arrival process (D-BMAP), Matrix Analytic Methods.

1. Introduction.

The study of random access systems of the Capetanakis-Tsybakov-Mikhailov-Vvedenskaya (CTMV) type has a long history, e.g., [7, 28, 10, 18, 12, 13, 11, 8, 23, 14, 6]. Underlying most of the theoretical work done in this area are the following key assumptions [27, 19]:

1. New arrivals occur according to a Poisson process with rate λ.
2. The number of nodes or stations is assumed to be infinite. In practice, the number of nodes is always finite. Assuming an infinite number provides us with pessimistic estimates for finite populations [1, 19]. In particular, each finite set of nodes can regard itself as an infinite set of virtual stations, one for each arriving packet. This situation is equivalent to the infinite node assumption and allows a station with backlogged packets to compete with itself.
3. A single error free channel provides immediate—that is, at the end of the slot—binary (collision or not) or ternary (collision, success or empty) feedback.
4. If two or more stations transmit simultaneously, then there is a collision, meaning that the transmissions interfere destructively so that none succeeds.

[*]Received on July 24, 2004; accepted for publication on November 15, 2004.
[†]University of Antwerp, Department of Mathematics and Computer Science, Middelheimlaan, 1, B-2020 Antwerp - Belgium {benny.vanhoudt,chris.blondia}@ua.ac.be

5. Time is *slotted* and may be considered discrete. Users are synchronized with respect to the time slots. Each slot has a fixed duration equal to the time required to transmit a packet.

A number of algorithms belonging to the class of the CTMV type have been studied with some of the assumptions weakened. For instance, Polyzos and Molle [21] have considered finite population models for the grouped access strategy, which they refer to as window access. In case of a finite population, one generally assumes that the new arrivals occur according to a Bernoulli process instead of a Poisson process (in which case the number of arrivals in consecutive slots is still independent). Finite population models were also developed by Boxma, Denteneer and Resing [6], who focused on approximating the delay characteristics of contention trees. Kessler, Seri and Sidi [24, 15] have relaxed the third and fourth assumption and studied the performance of splitting algorithms in noisy channels with memory and Markovian capture. Many researchers have also considered different types of feedback, e.g., "success—failure" and "something—nothing", and early/delayed feedback. A comprehensive overview of most of the extensions made to a non-standard environment can be found in [19, Section 6].

What is apparent from this overview is that almost all researchers assume Poisson arrivals, except for some of the results on blocked access algorithms and a limited number of finite population studies that consider Bernoulli arrivals. This might seem like an obvious choice, especially in case of an infinite population, because the traffic generated by a very large population with independent users approaches a Poisson process. Nevertheless, studying the performance of an algorithm with an infinite population under a broad set of arrival processes might be very useful because such an infinite population model is a pessimistic estimate for a finite population. Thus, we can further extend the theoretical foundation of algorithms of the CTMV type by proving that these algorithms have good stability characteristics in such an environment. In 1998, during the 50-th birthday of the IEEE Transactions on Information Theory Society, the ignorance of the bursty nature of real sources was identified by Ephremides and Hajek [9] as one of the key reasons why the union between information theory and communication networks has been only partially successful. It is the bursty nature of the arrivals that separates Markovian arrival processes from Poisson arrivals.

In this paper we leave the last four above-mentioned assumptions unchanged, instead we greatly relax the assumption made on the arrival process. That is, instead of assuming Poisson arrivals with a mean rate λ, we consider a rich class of arrival processes commonly known as discrete-time batch Markovian arrival processes (D-BMAPs). This class of arrival processes is known to lend itself very well to modeling bursty and correlated arrival processes commonly arising in computer and communication applications [4, 16, 20, 31, 25]. Recently, Van Houdt and Blondia [30] studied the impact of introducing D-BMAPs arrivals on the stability of CTMV type random access systems with blocked and grouped access. In [29] Van Houdt and Blondia demonstrated that the basic binary CTMV algorithm with free access (see Section 2 for a description) can be studied, using matrix analytical methods, by constructing a tree structured Markov chain of the Quasi-Birth-Death (QBD) type (see Section 4). In this work we extend the model presented in [29] in a number of ways: (1) we consider both the modified and basic version of the CTMV type algorithms, (2) allow the splitting factor Q to be larger than 2 and (3) no longer restrict ourselves to fair coins only. Using various numerical examples we demonstrate that the maximum stable throughput degrades as the arrival process becomes more bursty. However, for small splitting factors Q, the degradation is limited and the good efficiency characteristics of random access systems of the CTMV type with free access remain valid. We also demonstrate that the optimality of using fair coins for the basic version of the CTMV type algorithms is a property of the Poisson arrival process only. For correlated arrival processes, the use of biased coins increases the maximum stable throughput. Finally, as the Poisson arrival process is a member of the set of all D-BMAP processes, this paper presents a novel approach to obtain the well known stability results by Mathys and Flajolet [18].

The paper is organized as follows. Section 2 provides a short description of the CTMV type algorithms considered. In Section 3 we briefly recall the definition of a D-BMAP, whereas Section 4 reviews a Quasi-Birth-Death Markov chain with a tree structure. Next, the analytical models are presented in Section 5. An algorithm to determine the stability of such a Markov chain is given in Section 6. Finally, some numerical examples are presented in Section 7, whereas conclusions are drawn and model extensions are discussed in Section 8.

2. Specification of the Algorithms. In this section, we specify the algorithms to be analyzed [10, 18]. In a first subsection we describe a set of CTMV type algorithms that require binary feedback, called the basic CTMV algorithms, whereas in a second subsection the modified CTMV algorithms, exploiting ternary feedback, are considered. We start by summarizing the common features of both sets of algorithms.

A single channel (bus, cable, broadcast medium) is shared among many users (sources, nodes, stations) that transmit packetized messages. Time is slotted and transmissions can only occur at the beginning of a time slot. Each time slot has a fixed duration equal to the length of a packet. Each transmission is within the reception range of every user (in a wireless centralized LAN environment the Base Station could broadcast the result of each uplink transmission).

CTMV type algorithms are collision resolution algorithms for which each user strives to retransmit its colliding packet till it is correctly received. The users have to resolve this contention without the benefit of any other source of information on other users' activity. Colliding users are recursively separated, according to some randomization procedure, into distinct groups. The users of the first group retransmit in the next slot, while the users of the i-th group, $i > 1$, wait until the first $i-1$ groups are resolved.

Users that hold a packet (at time t) are referred to as active users (at time t). CTMV algorithms are conveniently implemented by letting each active user maintain an integer value, referred to as the current stack level. The current stack level held by a station can be seen as a representation of the number of "groups" that need to be resolved before a station is allowed to (re)transmit. A user is allowed to transmit its packet whenever its current stack level equals zero. At the end of each time slot the current stack level of all active stations is updated. The rules used to update the current stack level are different for both schemes.

2.1. The Basic Q-ary CTMV Algorithm with Free Access. The basic Q-ary CTMV algorithms are those corresponding with the original proposals made by Capetanakis [7]. The current stack level, maintained by each active user, is updated as follows:

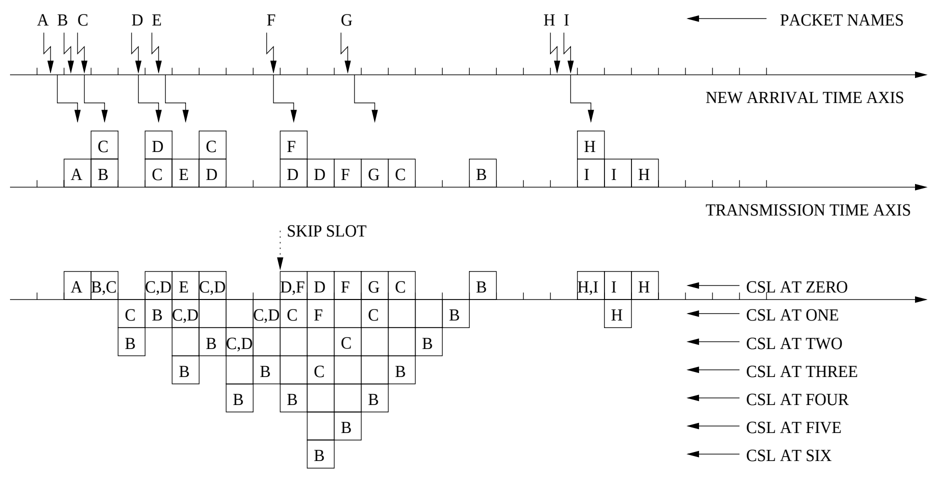
- An active user transmits in a time slot t whenever its current stack level for slot t is equal to zero. A user that became active during time slot $t-1$ initializes the current stack level for slot t at zero.
- At the end of a time slot t in which no collision occurs, users with a stack level $i, i > 0$, for slot t set their current stack level for slot $t+1$ at $i-1$ (while a possible successful user becomes inactive).
- At the end of a time slot t in which a collision occurs, all users with a current stack level $i, i > 0$, for slot t set their current stack level for slot $t+1$ at $i+Q-1$. Users with a current stack level for slot t equal to zero split into Q distinct groups: a user joins the i-th group with a probability p_{i-1}. Users that join the i-th group set their current stack level for slot $t+1$ equal to $i-1$.

Notice, new packet arrivals are allowed to take part in the scheme without any further delay. This channel-access technique is commonly known as free access, as opposed to blocked access schemes were new arrivals have to wait until all prior collisions have been resolved. Selecting one of the Q distinct groups (after a collision) can be seen as flipping a Q-sided coin. A distinction is also made between fair coins, i.e., $p_0 = \ldots = p_{Q-1} = 1/Q$, and biased coins. We will consider both fair and biased coins (we do assume that all the stations use the same coins, either fair or biased).

2.2. The Modified Q-ary CTMV Algorithm with Free Access. The modified CTMV algorithms are a well-known improvement of the basic CTMV algorithm that skips so-called *doomed* slots [1, 28]. *Doomed* slots are slots for which all active stations know that the above-mentioned operation of the basic Q-ary CTMV algorithm would result in a collision. In order to implement this optimization, one does require ternary feedback (empty, successful or collision slot). While the basic CTMV algorithm only requires binary feedback (collision or not). The idea is the following.

Suppose that a collision is followed by $Q-1$ empty slots. This implies that all packets involved in the collision selected the Q-th group. Using the basic Q-ary CTMV algorithm, these stations would transmit in the next slot (together with possible newcomers), generating a guaranteed collision. The modified scheme improves the basic scheme by omitting these slots and by splitting the set of stations that would otherwise result in a guaranteed collision into Q subsets. If the next $Q-1$ slots are again empty, we would get another guaranteed collision and therefore the next slot is again skipped. Thus, whenever, for some $i \geq 1$, the last $1+i(Q-1)$ slots contain a collision followed by $i(Q-1)$ empty slots, this otherwise-wasted slot can be skipped by having all stations immediately act as if it had occurred. This modified scheme is conveniently implemented using a current stack level and a simple count down counter.

Figure 1 presents an example of the transmission process for $Q=3$, it also includes a list of group numbers (1, 2 or 3) for each packet to indicate which group the packet joins after each collision (in which it is involved). Thus, the list $2,3,1,1,\ldots$ for packet D indicates that packet D joins the second group as a result of its first collision, the third as a result of its second collision, the first as a result of its third collision (the skipped collision) and again the first as a result of its fourth collision.



THE SEQUENCE OF GROUPS SELECTED BY THE ASSOCIATED TRANSMITTER:

A: ...	D: 2,3,1,1,...	G: ...
B: 3,...	E: ...	H: 2,...
C: 2,2,3,2,...	F: 2,...	I: 1,...

FIG. 1. *Example of the Transmission Process: CSL = Current Stack Level*

3. Discrete Time Batch Markovian Arrival Processes (D-BMAP). The D-BMAP is the discrete time counterpart of the BMAP [17] and was first introduced in [3]. Formally, a D-BMAP is defined by an infinite set of positive $l \times l$ matrices $(B_n)_{0 \leq n < \infty}$, with the property that

$$B = \sum_{n=0}^{\infty} B_n \tag{1}$$

is a transition matrix. By definition the Markov chain associated with B and having $\{i \mid 1 \leq i \leq l\}$ as its state space, is controlling the actual arrival process as follows. Suppose it is in state i at time t. By going to the next time instance $t+1$, there occurs a transition to another or possibly the same state, and a batch arrival may or may not occur. The entries $(B_n)_{i,j}$ represent the probability of having a transition from state i to j and a batch arrival of size n. So, a transition from state i to j without an arrival will occur with probability $(B_0)_{i,j}$.

For B aperiodic and irreducible the Markov chain has a unique stationary distribution. Let β be the stationary probability vector of the Markov chain characterized by B, i.e., $\beta B = \beta$ and $\beta e = 1$ with e a column vector of 1's. The mean arrival rate λ of the D-BMAP $(B_n)_n$ is given by

$$\lambda = \beta \left(\sum_{n=1}^{\infty} n B_n \right) e. \tag{2}$$

Many properties like the autocorrelation function or the index of dispersion for count (IDC) can be found in [3, 4, 5]. Another important characteristic of D-BMAPs is that any finite superposition of D-BMAPs is again a D-BMAP. Recently, open-source software became available to match IP traffic by means of a BMAP arrival process (see [16] and the references therein).

4. Markov Chain of Quasi-Birth-Death Type with a Tree Structure. In this section, we briefly describe the main characteristics of a tree structured Quasi-Birth-Death (QBD) Markov chain (MC). This type of MCs was first introduced by Takine, et al. [26] and Yeung, et al. [32, 33]. Consider a discrete time bivariate MC $\{(X_t, N_t), t \geq 0\}$ in which the values of X_t are represented by nodes of a Q-ary tree, and where N_t takes integer values between 1 and m. X_t is referred to as the node and N_t as the auxiliary variable of the MC at time t. A description of the transitions of the MC is given below.

A Q-ary tree is a tree for which each node has Q children. The root node is denoted as $\emptyset$. The remaining nodes are denoted as strings of integers, with each integer between 1 and Q. For instance, the k-th child of the root node is represented by k, the l-th child of the node k is represented by kl, and so on. Throughout this paper we use lower case letters to represent integers and upper case letters to represent strings of integers when referring to nodes of the tree. We use '+' to denote concatenation on the right. For example, if $J = j_1 j_2 \dots j_v, k = j_{v+1}$ then $J + k = j_1 j_2 \dots j_v j_{v+1}$.

The MC (X_t, N_t) is called an MC of the QBD-type with a tree structure if at each step the chain can only make transitions to its parent, children of its parent, or to its children. Moreover, if the chain is in state $(J + k, i)$ at time t then the state at time $t + 1$ is determined as follows:

1. (J, j) with probability $d_k^{i,j}, k = 1, \dots, Q$,
2. $(J + s, j)$ with probability $a_{k,s}^{i,j}, k, s = 1, \dots, Q$,
3. $(J + ks, j)$ with probability $u_s^{i,j}, k, s = 1, \dots, Q$.

Define $m \times m$ matrices $D_k, A_{k,s}$ and U_s with respective $(i, j)^{th}$ elements given by $d_k^{i,j}, a_{k,s}^{i,j}$ and $u_s^{i,j}$. Notice that transitions from state $(J + k, i)$ do not dependent upon J, moreover, transitions to state $(J + ks, j)$ are also independent of k. When the MC is in the root state $(X_t = \emptyset)$ at time t then the state at time $t + 1$ is determined as follows:

1. $(\emptyset, j)$ with probability $f^{i,j}$,
2. (s, j) with probability $u_s^{i,j}, s = 1, \dots, Q$.

Define the $m \times m$ matrix F with corresponding $(i, j)^{th}$ element given by $f^{i,j}$. We state that a tree structured QBD MC is stable if and only if for all states $(J + k, i)$, the probability of eventually reaching a state of the form (J, j) equals one. For a more detailed description of the notation and algebra see Yeung, et al. [32].

5. Analysis of the Random Access Schemes. The analysis of the random access schemes is divided into five different parts, each presented in a different subsection. Each part describes a tree structured QBD MC that is stable, resp. unstable, whenever either the basic or the modified CTMV algorithm, for specific values of Q, is stable, resp. unstable. The five subsections are listed below:

1. the basic CTMV algorithm with $Q = 2$,
2. the basic CTMV algorithm with $Q > 2$,
3. the modified CTMV algorithm with $Q = 2$,
4. the modified CTMV algorithm with $Q = 3$,
5. the modified CTMV algorithm with $Q > 3$.

With each new subsection some additional complexity is introduced. The MC developed in the first subsection was first presented in [29] and a short description is included here for clarity reasons.

5.1. The Basic CTMV algorithm with $Q = 2$. Consider the following stochastic process $(X_t, N_t = (Y_t, Z_t))$. Let X_t be the *backlogged string* consisting of the status of all backlogged stations at time slot t. A station is called a backlogged station whenever its current stack level is larger than 0. For instance, when $X_t = n_k \dots n_2 n_1$ there are $\sum_{i=1}^{k} n_i$ backlogged stations, for $n_i \geq 0$ backlogged stations the current stack level equals i. In this example there are no stations with a stack level larger than k. The sample space of the random variable X_t is $\Omega_1 = \{\emptyset\} \cup \{J \mid J = n_k \ldots n_1, n_j \geq 0, 1 \leq j \leq k, k \geq 1\}$. Notice that the string J is allowed to have a number of leading zeros. The random variable X_t has a tree structure. For instance, the children of $j_1 j_2 \ldots j_n$ are $j_1 j_2 \ldots j_n k, k \geq 0$. Thus, each node in the tree has an infinite number of children. Let Y_t be the number of stations that transmit at time slot t. The sample space of the random variable Y_t is $\Omega_2 = \{n \mid n \geq 0\}$. Finally, let Z_t be the state of the D-BMAP arrival process at the end of time slot t, hence the sample space Ω_3 of Z_t is $\{n \mid 1 \leq n \leq l\}$.

It is easy to see that $(X_t, N_t = (Y_t, Z_t))$ is an MC. The state space of the MC is $\Omega_1 \times (\Omega_2 \times \Omega_3)$. In order to study the stability (and to calculate the stationary distribution) of this MC numerically the nodes of X_t should have a finite number of children and the auxiliary variable N_t should have a finite range. Therefore, the MC (X_t, N_t) is approximated by another bivariate MC (X_t^d, N_t^d). (X_t^d, N_t^d) is obtained by setting a maximum d on the number of stations that can have the same current stack level (including level 0, i.e., the number of stations that transmit in slot t). If a situation occurs in which $d + k, k > 0$, stations have the same current stack level, k stations are assumed to drop their packet. We state that d is chosen sufficiently large when the ratio of dropped packets due to the introduction of d is smaller than 10^{-9}. This ratio can be obtained by comparing the load of the input D-BMAP and the probability that a successful transmission takes place. Provided that d is chosen sufficiently large we can study the stability of (X_t, N_t) by studying the MC (X_t^d, N_t^d).

Indeed, the chain (X_t, N_t) is unstable whenever the chain (X_t^d, N_t^d) is unstable. The stability of the chain (X_t^d, N_t^d) is not sufficient to formally prove that the chain (X_t, N_t) is stable. For instance, for every D-BMAP, (X_t^1, N_t^1) is stable. Even when d is chosen sufficiently large, it is still possible that the dropping of these rare packets (even when we lose less than one in a billion) causes the chain (X_t^d, N_t^d) to become stable while (X_t, N_t) is not. Hence, it is possible that we slightly overestimate the stability point of a particular arrival process. The Poisson results by Mathys and Flajolet [18] are the only existing point of comparision to get an idea of the margin of overestimation. Numerical results (not included in Section 7) have indicated that for $d = 10$ the overestimation is less than 0.000003 (the chain was unstable for $\lambda = 0.36018$ while the exact result by Flajolet states 0.360177). Further increasing d would result in even smaller overestimation errors. For each of the five models presented in this paper, the introduction of the parameter d is the only required approximation.

Let us now consider the MC $(X_t^d, N_t^d = (Y_t^d, Z_t))$ in more detail. X_t^d is the *backlogged string* that holds the status of all backlogged stations. As before, when $X_t^d = n_k \ldots n_2 n_1$ then $0 \leq n_i \leq d$ backlogged stations have a the current stack level equal to i. The sample space of the random variable X_t^d is obviously $\Omega_1^d = \{\emptyset\} \cup \{J \mid J = n_k \ldots n_1, 0 \leq n_j \leq d, 1 \leq j \leq k, k \geq 1\}$. Each node in Ω_1^d has $d + 1$ children. As opposed to the general description of the tree structured QBD MC in Section 4 we represent the children of a node by 0 to d instead of 1 to $d+1$. Y_t^d represents the number of stations that transmit in slot t (i.e., the current stack level of these stations is 0 at time t). The sample space of Y_t^d is $\Omega_2^d = \{n \mid 0 \leq n \leq d\}$. Finally, Z_t is the same random variable as before.

Assume that the MC (X_t^d, N_t^d) is in node $J+k$ at time t, i.e., $X_t^d = J+k$. Either slot t contains a collision, in which case the chain will be in a state of the form $J+ks, 0 \leq s \leq d$, at time $t+1$, or slot t does not hold a collision, meaning that the chain will be in state J at time $t+1$. Therefore, the chain can only make transitions to its parent or to its children. In order for the MC (X_t^d, N_t^d) to be a tree structured QBD MC the following two additional conditions have to be satisfied:

1. The probability of making a transition from state $(J+k,(i,j))$ to state $(J,(i',j'))$ may not dependent upon J. Clearly, j', the new state of the D-BMAP, is solely determined by j, the old state of the D-BMAP, and thus independent of J. The number of stations that transmit in slot $t+1$, that is, i', is determined by k, the number of stations that decrease their current stack level from 1 to 0, and j, the old state of the D-BMAP (because this state j determines the number of new arrivals in slot $t+1$).
2. The probability of making a transition from state $(J+k,(i,j))$ to state $(J+ks,(i',j'))$ may not dependent upon J or k. Again, j', the new state of the D-BMAP, is determined by j, the old state of the D-BMAP. While, s, the number of stations that increase their current stack level to 1 (as a result of the coin flipping), is determined by i and the probabilities p_0 and $p_1 = 1-p_0$. Finally, i', the number of stations that transmit in slot $t+1$, is determined by i, p_0 and j, the old state of the D-BMAP (because this state j determines the number of new arrivals).

In conclusion, the MC (X_t^d, N_t^d) is a tree structured QBD MC. A tree structured QBD MC is fully characterized by the matrices D_k, U_s, $A_{k,s}$ and F (see Section 4). In the remainder of this section we indicate how to calculate these matrices. Once that we obtained these matrices, they are the input variables of the iterative algorithm described in Section 6. This iterative algorithm determines whether the MC is stable or not.

The matrix F is of no importance for the stability of the MC, therefore, there is no need to discuss it in any of the five models considered. The matrices $A_{k,s}$ hold the transition probabilities that the chain (X_t^d, N_t^d) goes from state $(J+k,(i,j))$ to the state $(J+s,(i',j'))$. These transitions are transitions between sibling nodes. Remember that the chain (X_t^d, N_t^d) can only make transitions to its parent or to its children, therefore, the entries of the matrices $A_{k,s}$ are zero.

The matrices D_k hold the transition probabilities that the chain (X_t^d, N_t^d) goes from state $(J+k,(i,j))$ to the state $(J,(i',j'))$. This happens when a collision does not occur in slot t. Therefore, the state i, the number of stations that transmit in slot t, must be equal to 0 or 1. Moreover, the state i', the number of stations that transmit in slot $t+1$, equals k, the number of stations that decrease their current stack level from 1 to 0, plus some possible new arrivals. Hence,

$$D_k((i,j),(i',j')) = \begin{cases} (B_{i'-k})_{j,j'} & i \leq 1, i' \geq k, i' < d, \\ \sum_{l \geq d-k} (B_l)_{j,j'} & i \leq 1, i' \geq k, i' = d, \\ 0 & otherwise, \end{cases} \tag{3}$$

where $(B_n)_{j,j'}$ holds the probability that n new arrivals occur and that the input D-BMAP changes its state from j to j' (see Section 3).

The matrices U_s hold the transition probabilities that the chain (X_t^d, N_t^d) goes from state $(J+k,(i,j))$ to the state $(J+ks,(i',j'))$. This happens when slot t holds a collision. Therefore, the state i, the number of stations that transmit in slot t, must be larger than or equal to 2. Moreover, the state i', the number of stations that transmit in slot $t+1$, equals i, the number of stations that transmitted in slot t, minus s, the number of stations that increase their current stack level to 1 (as a result of the coin flipping), plus some possible new arrivals. Clearly, s can never be larger than i. Hence,

$$U_s((i,j),(i',j')) = \begin{cases} C_s^i p_0^{i-s} p_1^s (B_{i'-(i-s)})_{j,j'} & i > 1, i \geq s, i' \geq i-s, i' < d, \\ C_s^i p_0^{i-s} p_1^s \sum_{l \geq d-(i-s)} (B_l)_{j,j'} & i > 1, i \geq s, i' \geq i-s, i' = d, \\ 0 & otherwise, \end{cases} \tag{4}$$

where C_s^i denotes the number of different possible combinations of s from i different items.

5.2. The Basic CTMV algorithm with $Q > 2$. As in the previous subsection, we will construct a tree structured QBD MC that allows us to study the stability of the basic CTMV algorithm (but now for $Q > 2$). In the remainder of this section we indicate how to construct this MC and how to calculate the matrices that characterize the MC. These matrices are the input variables of the iterative algorithm described in Section 6.

Let $q_i, 0 \leq i \leq Q-1$, be the probability that a station increases its current stack level to i, as a result of the coin flipping procedure, provided that it does not increase its current stack level to a value above i. Hence,

$$q_i = \frac{p_i}{1 - \sum_{j>i} p_j}, \tag{5}$$

where $p_i, 0 \leq i \leq Q-1$, is the probability that a station increases its current stack level to i as a result of the coin flip.

Consider the stochastic process $(X_t, N_t = (Y_t, Z_t))$, where X_t denotes the *backlogged string* consisting of the status of all backlogged stations at time slot t, Y_t denotes the number of stations that transmit in time slot t and Z_t denotes the state of the input D-BMAP at the end of time slot t. In the previous subsection we showed that this process (to be correct its approximation (X_t^d, N_t^d)) is a tree structured QBD MC if $Q = 2$. For $Q > 2$, this process is still a tree structured MC, but it is no longer of the QBD type. For instance, after each slot in which a collision occurs, $Q - 1$ integers are added to the backlogged string. These $Q - 1$ integers represent the number of stations that increase their current stack level to 1, 2, ..., $Q - 1$ as a result of their coin flipping procedure.

We shall reduce the MC (X_k, N_k) to a tree structured QBD MC by constructing an expanded MC $(\mathcal{X}_t, \mathcal{N}_t = (\mathcal{Y}_t, \mathcal{Z}_t, \mathcal{Q}_t))$. The technique used to construct this expanded MC is similar to Ramaswami's [22] in order to reduce an M/G/1-type MC to a QBD MC. The key idea behind this expanded MC is that whenever a transition occurs that adds $Q - 1$ integers to the node variable X_k, we split this transition into $Q - 1$ transitions that each add one integer to the node variable $\mathcal{X}_t$.

Assume a given realization $(X_k(w), N_k(w))$ of the MC (X_k, N_k). The expanded chain $(\mathcal{X}_t, \mathcal{N}_t = (\mathcal{Y}_t, \mathcal{Z}_t, \mathcal{Q}_t))$ is constructed as follows (the range of $\mathcal{Q}_t$ is 0 to $Q - 2$).

Initial state: If $(X_0(w), N_0(w)) = (J, (i, j))$, then set $(\mathcal{X}_0(w), \mathcal{N}_0(w)) = (J, (i, j, 0))$. Also, set $k = 0$ and $t = 0$, k represents the steps of the original chain and t represents the steps of the expanded chain. We will establish a one-to-one correspondence between the state $(J, (i, j))$ of the original chain and the state $(J, (i, j, 0))$ of the expanded chain.

Transition Rules: We distinguish between three possible cases: $\mathcal{Q}_t(w) = 0$, $\mathcal{Q}_t(w) > 1$ and $\mathcal{Q}_t(w) = 1$.

1. $\mathcal{Q}_t(w) = 0$: Consider $(X_k(w), (Y_k(w), Z_k(w)))$, and do one of the following:
   - Assume that the k-th time slot does not hold a collision, i.e., $Y_k(w) \leq 1$. We set $\mathcal{X}_{t+1}(w) = X_{k+1}(w)$ and $\mathcal{N}_{t+1}(w) = (Y_{k+1}(w), Z_{k+1}(w), 0)$. Thus, the transitions that do not correspond to a collision remain identical. Next, both k and t are increased by one.
   - Assume that the k-th time slot does hold a collision, i.e., $Y_k(w) > 1$. Therefore, $X_{k+1}(w)$ can be written as $X_k(w) + s_{Q-1}s_{Q-2} \ldots s_2 s_1$. Then, $(\mathcal{X}_{t+1}(w), \mathcal{N}_{t+1}(w)) = (X_k(w) + s_{Q-1}, (Y_k(w) - s_{Q-1}, Z_k(w), Q - 2))$. Next, increment both t and k by one.
2. $\mathcal{Q}_t(w) > 1$: $X_k(w)$ can be written as $J + s_{Q-1}s_{Q-2} \ldots s_2 s_1$. Set $\mathcal{X}_{t+1}(w) = \mathcal{X}_t(w) + s_{\mathcal{Q}_t(w)}$ and $\mathcal{N}_{t+1}(w) = (\mathcal{Y}_t(w) - s_{\mathcal{Q}_t(w)}, \mathcal{Z}_t(w), \mathcal{Q}_t(w) - 1))$. Next, increase t by one and do not alter the value of k.
3. $\mathcal{Q}_t(w) = 1$: As before $X_k(w)$ can be written as $J + s_{Q-1}s_{Q-2} \ldots s_2 s_1$, set $\mathcal{X}_{t+1}(w) = \mathcal{X}_t(w) + s_1$ and $\mathcal{N}_{t+1}(w) = (Y_k(w), Z_k(w), 0)$. Increase t by one and do not alter the value of k.

The expanded MC $(\mathcal{X}_t, \mathcal{N}_t)$ is a tree structured QBD MC. The only problem is that every node in $(\mathcal{X}_t, \mathcal{N}_t)$ has an infinite number of children and the auxiliary variable has an infinite number of states. As in the previous subsection, we can resolve this problem by approximating the expanded chain by the chain $(\mathcal{X}_t^d, \mathcal{N}_t^d = (\mathcal{Y}_t^d, \mathcal{Z}_t, \mathcal{Q}_t))$ that is obtained by putting a maximum d on the number of stations that are allowed to have an identical current stack level.

The expanded MC $(\mathcal{X}_t^d, \mathcal{N}_t^d)$ does not allow transitions between sibling nodes. Therefore, the entries of the matrices $A_{k,s}$ are zero. Looking at the transition rules described above, the transition blocks of the MC $(\mathcal{X}_t^d, \mathcal{N}_t^d)$ are the following.

The matrices D_k hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{N}_t^d)$ goes from state $(J+k, (i, j, m))$ to the state $(J, (i', j', m'))$. This can only happen if $m = 0$, $m' = 0$ and $i \le 1$. Hence,

$$D_k((i,j,m),(i',j',m')) = \begin{cases} (B_{i'-k})_{j,j'} & m = m' = 0, i \le 1, i' \ge k, i' < d, \\ \sum_{l \ge d-k} (B_l)_{j,j'} & m = m' = 0, i \le 1, i' \ge k, i' = d, \\ 0 & otherwise, \end{cases} \tag{6}$$

where $(B_n)_{j,j'}$ holds the probability that n new arrivals occur and that the input D-BMAP changes its state from j to j' (see Section 3).

The matrices U_s hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{N}_t^d)$ goes from state $(J+k, (i, j, m))$ to the state $(J+ks, (i', j', m'))$. We separate three different cases. First, assume that $m = 0$. Hence,

$$\begin{aligned} &U_s((i,j,0),(i',j',m')) \\ &= \begin{cases} C_s^i q_{Q-1}^s (1-q_{Q-1})^{i-s} (I_l)_{j,j'} & m' = Q-2, i > 1, i' = i-s, \\ 0 & otherwise, \end{cases} \end{aligned} \tag{7}$$

where I_l is an $l \times l$ unity matrix. We simply add the number of colliding stations that increase their current stack level to $Q-1$ to the *backlogged string* $\mathcal{X}_t^d$. Second, for $m = 1$, we get

$$U_s((i,j,1),(i',j',m')) = \begin{cases} C_s^i q_1^s (1-q_1)^{i-s} (B_{i'-(i-s)})_{j,j'} & m' = 0, \\ & i \ge s, d > i' \ge i-s, \\ C_s^i q_1^s (1-q_1)^{i-s} \sum_{l \ge d-(i-s)} (B_l)_{j,j'} & m' = 0, i \ge s, i' = d, \\ 0 & otherwise. \end{cases} \tag{8}$$

Here the number of colliding stations that increase their current stack level to 1 is added to the *backlogged string* and we take the new arrivals into account. Finally, for $Q-1 > m > 1$, we have

$$U_s((i,j,m),(i',j',m')) = \begin{cases} C_s^i q_m^s (1-q_m)^{i-s} (I_l)_{j,j'} & m' = m-1, i' = i-s, \\ 0 & otherwise. \end{cases} \tag{9}$$

Once more, the number of colliding stations that increase their current stack level to m is concatenated to the *backlogged string*. This completes the discussion of the basic Q-ary CTMV algorithms. In the next sections we indicate how to adapt the MC above to model the modified CTMV schemes.

5.3. The modified CTMV algorithm with $Q = 2$. Consider the stochastic process $(X_t, N_t = (Y_t, Z_t))$, where X_t denotes the *backlogged string* consisting of the status of all backlogged stations at time slot t, Y_t denotes the number of stations that transmit in time slot t and Z_t denotes the state of the input D-BMAP at the end of time slot t. For the modified CTMV algorithm, the stochastic process $(X_t, N_t = (Y_t, Z_t))$ is not Markovian. We illustrate this by means of an example. Let $X_t = J + k, k > 1$ and $Y_t = 0$. This implies that the t-th time slot is empty and k stations have a current stack level equal to one. Consider the following two possibilities for X_{t-1}.

- $X_{t-1} = J$ and $Y_{t-1} = k$, in this case slot $t - 1$ holds a collision of exactly k stations. The state $X_t = J + k$ and $Y_t = 0$ is reached if each of the k colliding stations increments its current stack level to 1 (and no new arrivals occur). Moreover, at the end of slot t all stations know that slot $t + 1$ would result in a collision, i.e., is a *doomed* slot. As a results, all stations immediately act as if the collision did occur. Therefore, it is possible that $X_{t+1} = J + s$ (if s of the k stations decide to set their current stack level to 1 as a result of the coin flip).
- $X_{t-1} = J + k + 0$ and $Y_{t-1} = 1$, in which case slot $t - 1$ holds a successful transmission. As opposed to the first case, the stations do not consider slot $t + 1$ as a *doomed* slot, and the collision in slot $t + 1$ will take place. This implies that X_{t+1} is equal to J.

In conclusion, the state of the stochastic process $(X_t, N_t = (Y_t, Z_t))$ at time $t + 1$ is not solely determined by the state a time t, which implies that $(X_t, N_t = (Y_t, Z_t))$ is not Markovian.

Nevertheless, from the stochastic process $(X_t, N_t = (Y_t, Z_t))$, we can construct a tree structured QBD MC by adding a value, say -1, to the range of Y_t. $Y_t = -1$ then implies that slot t is empty and the next slot would have been a *doomed* slot (if we were using the basic scheme). While $Y_t = 0$ implies that slot t is empty and the next slot is not considered to be a *doomed* slot. Denote the stochastic process that is obtain by adding -1 to the range of Y_t as $(X_t, M_t = (Y_t, Z_t))$. The transitions to and from a state with $Y_t = -1$ are as follows. We enter a state with $Y_t = -1$ whenever a transition occurs from a collision slot to an empty slot. We stay in a state with $Y_t = -1$ if the next slot is an empty slot, otherwise we enter a state with $Y_t \neq -1$.

The stochastic process (X_t, M_t) can be shown to be a tree structured QBD MC (with similar arguments as in Section 5.1). (X_t, M_t) does however allow transitions between sibling nodes. This happens whenever an otherwise *doomed* slot is skipped. It is possible to use a more complex (and time consuming) iterative formula (compared to the one in Section 6), that determines whether a tree structured MC, that does allow transitions between sibling nodes, is stable. Instead, we construct a new tree structured QBD MC $(\mathcal{X}_t, \mathcal{M}_t = (\mathcal{Y}_t, \mathcal{Z}_t))$ that only uses transitions to parent and child nodes[1].

The range of the random variable $\mathcal{Y}_t$ equals $\{(0,n) \mid -1 \leq n\} \cup \{(1,n) \mid 2 \leq n\}$. We will establish a one-to-one correspondence between the states $(J,(i,j))$ of the MC (X_t, M_t) and the states $(J,((0,i),j))$ of $(\mathcal{X}_t, \mathcal{M}_t)$. The idea behind this expanded chain $(\mathcal{X}_t, \mathcal{M}_t)$ is that a transition from a node $J+k$ to a node $J+s$ is split into two transitions, a first one from node $J+k$ to J, followed by a second one from node J to $J+s$. When the transition from node $J+k$ to J takes place we store the value of k in $\mathcal{Y}_t$ by setting $\mathcal{Y}_t = (1,k)$. The fact that the first component of $\mathcal{Y}_t$ is equal to 1 indicates that the next transition has to be the second step of a split transition.

Assume a given realization $(X_k(w), M_k(w))$ of the MC (X_k, M_k). The expanded chain $(\mathcal{X}_t, \mathcal{M}_t)$ is constructed as follows.

Initial state: If $(X_0(w), M_0(w)) = (J,(i,j))$, then set $(\mathcal{X}_0(w), \mathcal{M}_0(w)) = (J,((0,i),j))$. Also, set $k = 0$ and $t = 0$, k represents the steps of the original chain and t represents the steps of the expanded chain.

Transition Rules: We consider two possibilities: $\mathcal{Y}_t(w) = (0,i)$ and $\mathcal{Y}_t(w) = (1,i)$.

1. $\mathcal{Y}_t(w) = (0,i)$: Consider $(X_k(w), M_k(w) = (Y_k(w), Z_k(w)))$, and assume that the k-th time slot holds a collision. We set $\mathcal{X}_{t+1}(w) = X_{k+1}(w)$ and $\mathcal{M}_{t+1}(w) = ((0, Y_{k+1}(w)), Z_{k+1}(w))$. Thus, the transitions remain identical in case of a collision. On the other hand, if the k-th time slot does not hold a collision, $Y_k(w) = 0, 1$ or -1. For $Y_k(w) = -1$, we can write $X_k(w)$ as $J+s$ with $s > 1$ and we get $(\mathcal{X}_{t+1}(w), \mathcal{M}_{t+1}(w)) = (J,((1,s), Z_k(w)))$. Second, for $Y_k(w) \neq -1$, we get $(\mathcal{X}_{t+1}(w), \mathcal{M}_{t+1}(w)) = (X_{k+1}(w),((0,Y_{k+1}(w)),Z_{k+1}(w)))$. Both k and t are increased by one in each of the cases mentioned above.
2. $\mathcal{Y}_t(w) = (1,i)$: $X_k(w)$ can be written as $J+s'$, set $\mathcal{X}_{t+1}(w) = \mathcal{X}_t(w) + s'$ and $\mathcal{M}_{t+1}(w) = ((0, Y_k(w)), Z_k(w))$. Next, increase t by one and do not alter the value of k.

As in the previous subsections, we make the number of children in each node and the number of states of the auxiliary variable $\mathcal{M}_t$ finite by putting a maximum d on the number of stations that are allowed to have the same current stack level.

[1] Actually, the MC that is created is called a tree-like process and various iterative algorithms apart from the one presented in Section 6 can be found in [2].

Looking at the transitions rules, the transition blocks $D_k, 0 \leq k \leq d$, and $U_s, 0 \leq s \leq d$, are the following. The matrices D_k hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J+k, ((m,i),j))$ to the state $(J, ((m',i'),j'))$. For $m = 0$ and $i \neq -1$, we get

$$D_k(((0,i),j),((m',i'),j')) = \begin{cases} (B_{i'-k})_{j,j'} & m' = 0, i \leq 1, i' \geq k, i' < d, \\ \sum_{l \geq d-k} (B_l)_{j,j'} & m' = 0, i \leq 1, i' \geq k, i' = d, \\ 0 & \textit{otherwise}. \end{cases} \tag{10}$$

For $m = 0$ and $i = -1$, we set

$$D_k(((0,-1),j),((m',i'),j')) = \begin{cases} (B_{i'-k})_{j,j'} & k = 0 \text{ or } 1, m' = 0, i' \geq k, i' < d, \\ \sum_{l \geq d-k} (B_l)_{j,j'} & k = 0 \text{ or } 1, m' = 0, i' \geq k, i' = d, \\ (I_l)_{j,j'} & k > 1, m' = 1, i' = k, \\ 0 & \textit{otherwise}, \end{cases} \tag{11}$$

where I_l is an $l \times l$ identity matrix. A visit to one of the states $(J+k, ((0,-1),j))$, with $k = 0$ or 1, can never occur (the states are transient with an expected return probability equal to 0). Nevertheless, we can still make use of the iterative scheme in Section 6 by making sure that the probability of eventually returning to a state of the form $(J, ((m,i),j))$ equals one. We realize this by making sure that the corresponding rows of the matrices D_0 and D_1 are stochastic. This explains the somewhat unexpected first two lines in the equation above (we act as if $i = 0$, but any stochastic row will do). For $m = 1$, all entries of $D_k, 0 \leq k \leq d$, are zero.

The matrices U_s hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J+k, ((m,i),j))$ to the state $(J+ks, ((m',i'),j'))$. For $i \neq s$, meaning that the first group is not empty, we get

$$\begin{aligned} & U_s(((m,i),j),((m',i'),j')) \\ & = \begin{cases} C_s^i p_1^s p_0^{i-s} (B_{i'-(i-s)})_{j,j'} & m' = 0, i > 1, \\ & i > s, d > i' \geq i-s, \\ C_s^i p_1^s p_0^{i-s} \sum_{l \geq d-(i-s)} (B_l)_{j,j'} & m' = 0, i > 1, i > s, i' = d, \\ 0 & \textit{otherwise}. \end{cases} \end{aligned} \tag{12}$$

For $i = s$, which implies that the first group is empty, we have

$$U_s(((m,i),j),((m',i'),j')) = \begin{cases} p_1^s (B_0)_{j,j'} & m' = 0, i > 1, i' = -1, \\ p_1^s (B_{i'})_{j,j'} & m' = 0, i > 1, 0 < i' < d, \\ p_1^s \sum_{l \geq d} (B_l)_{j,j'} & m' = 0, i > 1, i' = d, \\ 0 & \textit{otherwise}, \end{cases} \tag{13}$$

because the next slot would be doomed if there are no new arrivals when using the basic CTMV algorithm. Notice that both these sets of equations are valid for $m = 0, 1$ and also for $i = -1$.

5.4. The modified CTMV algorithm with $Q = 3$. For the basic CTMV algorithm we made use of two different models, one for $Q = 2$ and another for $Q > 2$. For the modified CTMV algorithm we make use of three different models. Each model description is only valid for the specified range of Q. Rather than going through the entire process that is used to construct the remaining two models, i.e., tree structured QBD MCs, we restrict ourselves to a description of the state space of the MCs and their corresponding transition probabilities. The techniques used to construct both models are a combination of the methods used to construct the previous two models.

The MC $(\mathcal{X}_t^d, \mathcal{M}_t^d = (\mathcal{Y}_t^d, \mathcal{Z}_t))$, used to study the modified ternary CTMV algorithm, is defined on the state space $\Omega_1^d \times (\Omega_2^d \times \Omega_3)$, where $\Omega_1^d = \{\emptyset\} \cup \{J \mid J = n_k \dots n_1, 0 \leq n_j \leq d, 1 \leq j \leq k, k \geq 1\}$, $\Omega_2^d = \{(0,n) \mid -1 \leq n \leq d\} \cup \{(1,n) \mid 0 \leq n \leq d\} \cup \{(2,n) \mid 2 \leq n \leq d\}$ and $\Omega_3 = \{n \mid 1 \leq n \leq l\}$. The transition matrices D_k, U_s and $A_{k,s}$ are the following. The entries of the matrices $A_{k,s}$ are all zero. Thus, the chain does not allow transitions between sibling nodes.

The matrices D_k hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J + k, ((m, i), j))$ to the state $(J, ((m', i'), j'))$. For $m = 0$ and $i \neq -1$, we get

$$D_k(((0,i),j),((m',i'),j')) = \begin{cases} (B_{i'-k})_{j,j'} & m' = 0, i \leq 1, i' \geq k, i' < d, \\ \sum_{l \geq d-k} (B_l)_{j,j'} & m' = 0, i \leq 1, i' \geq k, i' = d, \\ 0 & \textit{otherwise}. \end{cases} \tag{14}$$

For $m = 0$ and $i = -1$, we set

$$D_k(((0,-1),j),((m',i'),j')) = \begin{cases} (B_0)_{j,j'} & k = 0 \text{ or } 1, m' = 0, i' = -1, \\ (B_{i'})_{j,j'} & k = 0 \text{ or } 1, m' = 0, d > i' > 0, \\ \sum_{l \geq d} (B_l)_{j,j'} & k = 0 \text{ or } 1, m' = 0, i' = d, \\ (I_l)_{j,j'} & k > 1, m' = 2, i' = k, \\ 0 & \textit{otherwise}, \end{cases} \tag{15}$$

where I_l is a $l \times l$ identity matrix. For $m = 1$ and 2, all entries of $D_k, 0 \leq k \leq d$, are zero.

The matrices U_s hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J + k, ((m, i), j))$ to the state $(J + ks, ((m', i'), j'))$. For $m = 0$ or 2, we get

$$U_s(((m,i),j),((m',i'),j')) \\ = \begin{cases} C_s^i q_2^s (1-q_2)^{i-s} (I_l)_{j,j'} & m'=1, i>1, i\geq s, i'=i-s, \\ 0 & \textit{otherwise}. \end{cases} \tag{16}$$

For $m=1$ and $i>0$, we get

$$U_s(((1,i),j),((m',i'),j')) \\ = \begin{cases} C_s^i q_1^s (1-q_1)^{i-s} (B_{i'-(i-s)})_{j,j'} & m'=0, i\geq s, \\ & d>i'\geq i-s, \\ C_s^i q_1^s (1-q_1)^{i-s} \sum_{l\geq d-(i-s)} (B_l)_{j,j'} & m'=0, i\geq s, i'=d, \\ 0 & \textit{otherwise}. \end{cases} \tag{17}$$

While for $m=1$ and $i=0$, we have

$$U_s(((1,0),j),((m',i'),j')) = \begin{cases} (B_0)_{j,j'} & m'=s=0, i'=-1, \\ (B_{i'})_{j,j'} & m'=s=0, 0<i'<d, \\ \sum_{l\geq d}(B_l)_{j,j'} & m'=s=0, i'=d, \\ 0 & \textit{otherwise}. \end{cases} \tag{18}$$

5.5. The modified CTMV algorithm with $Q>3$. The MC $(\mathcal{X}_t^d, \mathcal{M}_t^d = (\mathcal{Y}_t^d, \mathcal{Z}_t))$, used to study the modified CTMV algorithm with $Q>3$, is defined on the state space $\Omega_1^d \times (\Omega_2^d \times \Omega_3)$, where $\Omega_1^d = \{\emptyset\} \cup \{J \mid J = n_k \ldots n_1, 0 \leq n_j \leq d, 1 \leq j \leq k, k \geq 1\}$, $\Omega_2^d = \{(i,n) \mid 0 \leq i \leq Q-3, -1 \leq n \leq d\} \cup \{(Q-2,n) \mid 0 \leq n \leq d\} \cup \{(Q-1,n) \mid 2 \leq n \leq d\}$ and $\Omega_3 = \{n \mid 1 \leq n \leq l\}$. The transition matrices D_k, U_s and $A_{k,s}$ are the following. As before the entries of the matrices $A_{k,s}$ are all zero.

The matrices D_k hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J+k,((m,i),j))$ to the state $(J,((m',i'),j'))$. For $m=0$ and $i\neq -1$, we get

$$D_k(((0,i),j),((m',i'),j')) = \begin{cases} (B_{i'-k})_{j,j'} & m'=0, i\leq 1, i'\geq k, i'<d, \\ \sum_{l\geq d-k}(B_l)_{j,j'} & m'=0, i\leq 1, i'\geq k, i'=d, \\ 0 & \textit{otherwise}. \end{cases} \tag{19}$$

For $m=0$ and $i=-1$, we set

$$D_k(((0,-1),j),((m',i'),j')) = \begin{cases} (B_0)_{j,j'} & k=0 \text{ or } 1, m'=0, i'=-1, \\ (B_{i'})_{j,j'} & k=0 \text{ or } 1, m'=0, d>i'>0, \\ \sum_{l\geq d}(B_l)_{j,j'} & k=0 \text{ or } 1, m'=0, i'=d, \\ (I_l)_{j,j'} & k>1, m'=Q-1, i'=k, \\ 0 & \textit{otherwise}, \end{cases} \tag{20}$$

where I_l is a $l \times l$ identity matrix. For $m \neq 0$, all entries of $D_k, 0 \leq k \leq d$, are zero.

The matrices U_s hold the transition probabilities that the chain $(\mathcal{X}_t^d, \mathcal{M}_t^d)$ goes from state $(J+k, ((m,i),j))$ to the state $(J+ks, ((m',i'),j'))$. For $m = 0$ or $Q-1$, we get

(21)
$$U_s(((m,i),j),((m',i'),j')) = \begin{cases} C_s^i q_{Q-1}^s (1-q_{Q-1})^{i-s} (I_l)_{j,j'} & m' = Q-2, i > 1, \\ & i \geq s, i' = i-s, \\ 0 & \textit{otherwise.} \end{cases}$$

For $m = 1$ and $i \geq 0$,

$$U_s(((1,i),j),((m',i'),j'))$$
$$= \begin{cases} C_s^i q_1^s (1-q_1)^{i-s} (B_{i'-(i-s)})_{j,j'} & m' = 0, i \geq s, \\ & d > i' \geq i-s, \\ C_s^i q_1^s (1-q_1)^{i-s} \sum_{l \geq d-(i-s)} (B_l)_{j,j'} & m' = 0, i \geq s, i' = d, \\ 0 & \textit{otherwise.} \end{cases} \tag{22}$$

For $m = 1$ and $i = -1$,

$$U_s(((1,-1),j),((m',i'),j')) = \begin{cases} (B_0)_{j,j'} & m' = s = 0, i' = -1, \\ (B_{i'})_{j,j'} & m' = s = 0, 0 < i' < d, \\ \sum_{l \geq d} (B_l)_{j,j'} & m' = s = 0, i' = d, \\ 0 & \textit{otherwise.} \end{cases} \tag{23}$$

While, for $m = Q-2$,

(24)
$$U_s(((Q-2,i),j),((m',i'),j')) = \begin{cases} C_s^i q_{Q-2}^s (1-q_{Q-2})^{i-s} (I_l)_{j,j'} & m' = Q-3, i > 0, \\ & i \geq s, i' = i-s, \\ (I_l)_{j,j'} & m' = Q-3, \\ & i = s = 0, i' = -1, \\ 0 & \textit{otherwise.} \end{cases}$$

Finally, for $1 < m < Q-2$, we have

$$U_s(((m,i),j),((m',i'),j'))$$
$$= \begin{cases} C_s^i q_m^s (1-q_m)^{i-s} (I_l)_{j,j'} & m' = m-1, i > -1, \\ & i \geq s, i' = i-s, \\ (I_l)_{j,j'} & m' = m-1, s = 0, i = i' = -1, \\ 0 & \textit{otherwise.} \end{cases} \tag{25}$$

6. Stability of Tree Structured QBD MCs. In this section, we present an iterative algorithm that allows us to investigate the stability of a tree structured QBD MC that only allows transitions to parent and child nodes. The input parameters of the iterative algorithm are the matrices $D_k, 0 \leq k \leq d$, and $U_s, 0 \leq s \leq d$. The following three sets of matrices play an important role [32].

Let $G_k, 0 \leq k \leq d$, denote the matrix whose $(i,v)^{th}$ element is the probability that the Markov chain (X_t^d, N_t^d) is in state (J,v) at the end of the fundamental period given that this period starts from state $(J+k,i)$. These matrices are stochastic for recurrent QBD Markov chains with a tree structure (Takine, *et al* [26]). Define $R_k, 0 \leq k \leq d$, as the matrix whose $(i,v)^{th}$ element is the expected number of visits to $(J+k,v)$ given that $(X_0^d, N_0^d) = (J,i)$ before visiting node J again. Finally, let $V_k, 0 \leq k \leq d$, denote the matrix whose $(i,v)^{th}$ element is the taboo probability that starting from $(J+k,i)$, the chain eventually returns to a node with the same length as $J+k$ by visiting $(J+k,v)$, under the taboo of the node J and the sibling nodes of $J+k$, i.e., the nodes $J+s, s \neq k$.

Yeung and Alfa [32] have shown that the matrices G_k and R_k can be expressed in terms of V_k. Moreover, because the MC does not allow transitions between sibling nodes, they were able to shown that the following simple expressions hold

$$G_k = (I - V_k)^{-1} D_k, \tag{26}$$

$$R_k = U_k (I - V_k)^{-1}, \tag{27}$$

$$V_k = A_{k,k} + \sum_{s=0}^{d} U_s G_s. \tag{28}$$

Notice that the matrices $V_k, 0 \leq k \leq d$, are identical if the matrices $A_{k,k}, 0 \leq k \leq d$, are identical. Clearly, if the MC only allows transitions to parent or child nodes, the entries of the matrices $A_{k,k}, 0 \leq k \leq d$, are equal to zero, resulting in identical $V_k, 0 \leq k \leq d$ matrices. In the remaining part of this section we drop the subscript k if we refer to V_k. Using equations (26) and (28), we obtain

$$V = \sum_{s=0}^{d} U_s (I - V)^{-1} D_s. \tag{29}$$

As a special case of Theorem 2 in Yeung and Alfa [32], the matrix V can be obtained as $\lim_{N \to \infty} V[N]$ from the recursion

$$V[N+1] = \sum_{s=0}^{d} U_s (I - V[N])^{-1} D_s, \tag{30}$$

where $V[0] = 0$. Also, the matrices $G_s[N] = (I - V[N])^{-1} D_s$ converge to the substochastic matrices G_s. We repeat the recursion until all matrices $G_s[N], 0 \leq s \leq d$, have stabilized.

The iterative formula (30) can be further optimized by making use of the structural properties of the matrices D_s, U_s and $V[N]$. For the basic and the modified binary CTMV algorithm, this optimization was limited to an acceleration of the product of $(I - V[N])^{-1}$ with the matrices D_s, where we made use of the fact that about 80 percent of the rows of D_s contain nothing but zeros. For higher splitting factors Q, this percentage is even higher (90 to 95 percent). The inversion of the matrix $I - V[N]$ was also optimized for $Q > 2$. We will demonstrate this for the basic CTMV algorithm with $Q > 2$, the technique is similar (slightly more complex) for the modified scheme with $Q = 3$ and $Q > 3$.

Consider the $l(d+1)(Q-1) \times l(d+1)(Q-1)$ matrix V, that corresponds to the tree structured QBD MC presented in Section 5.2, whose $(i,v)^{th}$ element is the taboo probability that starting from $(J+k, i)$, the chain eventually returns to a node with the same length as $J+k$ by visiting $(J+k, v)$, under the taboo of the node J and the sibling nodes of $J+k$. Next, subdivide the matrix V in blocks of size $l(d+1) \times l(d+1)$.

$$V = \begin{pmatrix} V_{0,0} & V_{0,1} & \dots & V_{0,Q-2} \\ \vdots & \vdots & \ddots & \vdots \\ V_{Q-2,0} & V_{Q-2,1} & \dots & V_{Q-2,Q-2} \end{pmatrix}, \tag{31}$$

where the elements of V_{q_1,q_2} are the taboo probabilities that starting from $(J + k, (i, j, q_1))$, the chain $(\mathcal{X}_t^d, \mathcal{N}_t^d = (\mathcal{Y}_t^d, \mathcal{Z}_t, \mathcal{Q}_t))$ eventually returns to a node with the same length as $J+k$ by visiting $(J+k, (v, u, q_2))$, under the taboo of the node J and the sibling nodes of $J+k$. Looking at the transition probabilities of $(\mathcal{X}_t^d, \mathcal{N}_t^d)$, these taboo probabilities are equal to zero if $q_2 \neq 0$. Thus,

$$V = \begin{pmatrix} V_{0,0} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ V_{Q-2,0} & 0 & \dots & 0 \end{pmatrix}. \tag{32}$$

The inverse $(I-V)^{-1}$ of a matrix V with such a structure is found as

$$(I-V)^{-1} = \begin{pmatrix} (I - V_{0,0})^{-1} & 0 & 0 & \dots & 0 \\ V_{1,0}(I - V_{0,0})^{-1} & I & 0 & \dots & 0 \\ V_{2,0}(I - V_{0,0})^{-1} & 0 & I & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ V_{Q-2,0}(I - V_{0,0})^{-1} & 0 & 0 & \dots & I \end{pmatrix}. \tag{33}$$

Clearly, the matrices $0 \leq V[N] \leq V, N \geq 0$, have the same structure as V and therefore, we can reduce the complexity of the matrix inversion in (30) from $l^3d^3Q^3$ to l^3d^3Q. Moreover, the structure of $V[N]$ also implies that only the first $l(d+1)$ columns of the matrix products between the matrices U_s and $(I - V[N])^{-1}D_s$ differ from zero. Allowing us to reduce the complexity of these products from $l^3d^3Q^3$ to $l^3d^3Q^2$.

7. Numerical Results. For all the numerical examples presented in this section, d is chosen sufficiently large. We state that d is sufficiently large if the ratio of dropped packet due to the introduction of d is smaller than 10^{-9}. This ratio is found by comparing the load of the input D-BMAP with the probability p_s that a successful transmission takes place. The value of p_s can be calculated by means of the matrices $R_k, 0 \leq k \leq d$, presented in Section 6. For most examples $d = 10$ was more than sufficient.

To test whether a tree structured MC is stable we calculate the matrices $G_k, 0 \leq k \leq d$, (as indicated in Section 6) and check whether they are stochastic. If all the row sums of G_k are between $1 - 10^{-9}$ and 1, we conclude that G_k is stochastic. If there is a row in G_k for which the row sum is below $1 - 10^{-4}$ we conclude that the matrix G_k is not stochastic. If the smallest row sum of G_k is between $1 - 10^{-4}$ and $1 - 10^{-9}$ we conclude that the stochastic nature of G_k is undetermined (i.e., the stability of the chain is unclear).

As with most of the iterative formulas used in the matrix analytical methods field, the number of iterations required by formula (30) increases significantly when the MC is close to unstable (whereas 10 to 100 iterations suffice for many stable and unstable MCs, the number of iterations can become as large as a few thousands when the chain is (very) close to the instability point). This limits the precision by which instability points are determined.

We determine the instability point, i.e., maximum achievable throughput, of the basic and the modified CTMV algorithm for different arrival processes that belong to the class of the D-BMAP processes. We consider both fair, i.e., $p_0 = p_1 = \ldots = p_{Q-1} = 1/Q$, and biased coins. We start by describing the D-BMAP subclasses that we considered. Next, we present the results for the basic CTMV algorithm with fair coins (for different values of Q), followed by the results of the modified CTMV algorithm with fair coins. Finally, we briefly consider the use of biased coins.

In the remainder of the paper, the instability point is also referred to as the stability point as this is the point where the CTMV protocol switches between being stable and unstable.

7.1. The D-BMAP subclasses.

7.1.1. The Poisson Process.

Mathys, et all. [18] have studied the stability, i.e., maximum achievable throughput, of the basic and modified Q-ary CTMV protocol (with open access) under Poisson traffic by means of functional equations. We will always start our numerical investigations by confirming their Poisson result. We can model the Poisson process as a D-BMAP by letting $B_n = e^{-\lambda}\lambda^n/n!$, for $n \geq 0$. For later reference, we abbreviate the Poisson process as $PP(\lambda)$.

7.1.2. The Erlang Process.

We define the Erlang process as follows. The Erlang process has independent and identically distributed interarrival times that obey an Erlang distribution with parameters k and λ. Clearly, for $k = 1$ the Erlang process is reduced to the Poisson process. The Erlang process can be modeled as a D-BMAP in the following way. Let $\beta_n = e^{-\lambda}\lambda^n/n!, n \geq 0$ and let $B_n, n \geq 0$, be $k \times k$ matrices defined as

$$(B_n)_{i,j} = \beta_{nk+j-i} \qquad nk \geq j-i, \tag{34}$$

$$(B_n)_{i,j} = 0 \qquad nk < j-i. \tag{35}$$

For later reference, we abbreviate the Erlang k process as $ER(\lambda, k)$.

7.1.3. The Markov Modulated Poisson Process. We restrict ourselves to the Markov modulated Poisson processes with two states. These processes are characterized by two parameters λ_1, λ_2 and a 2×2 matrix T. The process will generate arrivals according to a rate λ_i when the current state is i. Transitions from one state to another can occur at the end of each time slot according to a 2×2 transition matrix T

$$T = \begin{pmatrix} 1-1/e & 1/e \\ 1/f & 1-1/f \end{pmatrix}. \tag{36}$$

The expected sojourn time in state 1, resp. state 2, is e, resp. f time slots. For later reference, we abbreviate the Markov Modulated Poisson process with parameters λ_1, λ_2, e and f as $M(\lambda_1, \lambda_2, e, f)$.

7.1.4. The Bulk Arrival Process. The Bulk arrival process is defined as a discrete time arrival process characterized by a $1 \times n$ vector v and a length L. The arrival pattern of this process consists of a repetition of identical cycles. The first part of each cycle consists of a set of batches, characterized by v. For instance $v = [2, 3, 2]$ means that we first have a batch of size 2, in the next time slot we have a batch of size 3, followed by a batch of size 2. The second part of the cycle is a silent period with a geometrically distributed length with average L. The Bulk arrival process can be described by the following D-BMAP. Let $v = [v_1, \ldots, v_n]$ and let $B_m, m \geq 0$, be a set of $n+1 \times n+1$ matrices with

$$(B_{v_m})_{m,m+1} = 1, \tag{37}$$

$$(B_0)_{n+1,1} = 1/L, \tag{38}$$

$$(B_0)_{n+1,n+1} = 1-1/L. \tag{39}$$

The other components of the matrices B_m are equal to zero. The load of a Bulk arrival process equals $\sum_m v_m/(L+n)$. For later reference, we abbreviate the Bulk arrival process with parameters v and L as $B(v, L)$.

7.2. The basic CTMV algorithm with fair coins. Table 1 presents the stability points, i.e., maximum achievable throughput, of nine different arrival processes:

the Poisson process, three Markov modulated Poisson processes, three Bulk arrival processes and two Erlang processes and this for $Q = 2, 3, 4$ and 5. For the Poisson process and the Erlang processes we start with $\lambda = 0$ and increase λ until instability is reached. For the bulk arrival processes we fix v and decrease L until instability is reached (we started with a large value of L). Finally, for the Markov modulated Poisson processes we fix e, f and λ_2 (the last one possibly as a function of λ_1) and increase λ_1 until instability is reached.

For each couple (a, Q), where a is an arrival process and Q the splitting factor, Table 1 presents two values x and y. The first x is the lower bound α of the interval $]\alpha, \alpha+0.001[$ that holds the instability point of the arrival process a, i.e., the maximum load of the D-BMAP for which it is stable. The second y indicates the difference between α and α^* in multiples of 0.001, where $]\alpha^*, \alpha^* + 0.001[$ holds the instability point of the Poisson process.

Let us study these results in detail. The Poisson results presented in Table 1 are in complete correspondence with the results obtained in [18]. This means that the results obtained by Mathys [18] lie within the intervals presented in Table 1. Replacing the input Poisson process by a Markov modulated Poisson process results in an inferior stability. This implies that more bursty and more correlated (compare the second MMPP with the third) input traffic results in a worse stability, i.e., a lower maximum achievable throughput. Moreover, the higher the splitting factor Q the larger the throughput degradation, e.g., replacing the Poisson input by $M(\boldsymbol{.}, 0, 30, 30)$ input results in a loss of 1.2% for $Q = 2$, 2.6% for $Q = 3$, 3.5% for $Q = 4$ and 4.1% for $Q = 5$. Therefore, lower splitting factors Q are better equipped to cope with bursty and correlated input traffic. Intuitively, one can understand this as follows. More bursty and correlated traffic generally results in more collisions. A collision results in an increment of the current stack level of all backlogged stations. The higher Q the higher the increment. Thus, for every collision one needs at least $Q - 1$ empty or successful slots in order to return to the same current stack level. Therefore, higher splitting factors suffer more under increased burstiness (the scheme is unstable if Q times the probability that a slot holds a collision is larger than 1).

Also, notice that a factor $Q = 2$ performs better, 0.4%, than a factor $Q = 5$ for the $M(\boldsymbol{.}, 0, 300, 300)$ process (for Poisson input it was the complete opposite). As a matter of fact, for any two factors Q_1 and $Q_2 \neq Q_1$, within the range $[2, 5]$, we can find some input process in Table 1 for which the factor Q_1 outperforms the factor Q_2, except for $Q_1 = 2$ and $Q_2 = 3$.

Let us now consider the Erlang results. Replacing the input Poisson process by an Erlang process results in a superior stability. This result corresponds with the previous result, i.e., less bursty traffic results in a higher maximum achievable throughput. Moreover, the higher the splitting factor Q the larger the increment, e.g., replacing the Poisson input by $ER(\boldsymbol{.}, 3)$ input results in a gain of 0.7% for $Q = 2$,

TABLE 1
The basic Q-ary CTMV algorithm.

Process	$Q = 2$		$Q = 3$		$Q = 4$		$Q = 5$	
$PP(\centerdot)$	0.360	+0	0.401	+0	0.399	+0	0.387	+0
$M(\centerdot, 2\lambda_1, 30, 30)$	0.358	-2	0.397	-4	0.393	-6	0.380	-8
$M(\centerdot, 0, 30, 30)$	0.348	-12	0.375	-26	0.364	-35	0.346	-41
$M(\centerdot, 0, 300, 300)$	0.347	-13	0.373	-28	0.361	-38	0.343	-44
$ER(\centerdot, 2)$	0.365	+5	0.419	+18	0.427	+28	0.425	+38
$ER(\centerdot, 3)$	0.367	+7	0.427	+26	0.441	+42	0.444	+57
$B([2], \centerdot)$	0.348	-12	0.359	-42	0.327	-72	0.291	-96
$B([3], \centerdot)$	0.349	-11	0.372	-29	0.352	-47	0.325	-62
$B([4], \centerdot)$	0.348	-12	0.371	-30	0.355	-44	0.332	-55

2.6% for $Q = 3$, 4.2% for $Q = 4$ and 5.7% for $Q = 5$. Therefore, higher splitting factors Q are better equipped to take advantage of less bursty input traffic (the explanation is the same as before).

The Bulk arrival processes, the most artificial of the processes considered, are mainly introduced to indicate that exotic arrival patterns can seriously deteriorate the stability of the basic CTMV algorithm, especially for higher splitting factors Q. For the binary scheme the loss is only about 1.2% percent, for $Q = 5$ it varies between 5.5% and 9.6%. If we were to increase Q even more, things only become worse, e.g., for $Q = 10$ the basic CTMV algorithm is unstable for a load of 0.18 under $B([2], \centerdot)$ input traffic.

In conclusion, when implementing the basic CTMV algorithm, one should always select a splitting factor $Q = 2$ or 3 because the throughput degradation due to the introduction of correlation and burstiness is less severe for a low splitting factor Q, e.g., the difference between the worst possible and the best input traffic is 2.0% for $Q = 2$ (see Table 1). Although, the basic ternary CTMV algorithm is more sensitive to the specific nature of the input process, i.e., the variation of the maximum achievable throughput is higher compared to the binary scheme, it still remains a practical optimum because, for each of the nine processes considered, it outperforms the binary scheme.

7.3. The modified CTMV algorithm with fair coins. Mathys and Flajolet [18] have shown that the modified CTMV algorithm improves the stability by 2.7% for $Q = 2$, by 0.54% for $Q = 3$, by 0.15% for $Q = 4$ and 0.05% for $Q = 5$. In this subsection we investigate whether the improvements that we get for other input processes are similar.

Table 2 represents the stability results for the same nine arrival processes studied in the previous subsection. For each couple (a, Q), where a is an arrival process and

Table 2
The modified Q-ary CTMV algorithm.

Process	$Q=2$		$Q=3$		$Q=4$		$Q=5$	
$PP(\centerdot)$	0.388	+27	0.406	+5	0.400	+1	0.387	+0
$M(\centerdot, 2\lambda_1, 30, 30)$	0.384	+26	0.402	+5	0.395	+2	0.381	+1
$M(\centerdot, 0, 30, 30)$	0.371	+23	0.380	+5	0.365	+1	0.346	+0
$M(\centerdot, 0, 300, 300)$	0.370	+23	0.377	+4	0.362	+1	0.343	+0
$ER(\centerdot, 2)$	0.394	+29	0.424	+5	0.429	+2	0.425	+0
$ER(\centerdot, 3)$	0.396	+29	0.432	+5	0.443	+2	0.444	+0
$B([2], \centerdot)$	0.377	+29	0.365	+6	0.328	+1	0.291	+0
$B([3], \centerdot)$	0.378	+29	0.378	+6	0.353	+1	0.325	+0
$B([4], \centerdot)$	0.377	+29	0.378	+7	0.357	+2	0.333	+1

Q the splitting factor, Table 2 presents two values x and y. The first x is the lower bound α of the interval $]\alpha, \alpha + 0.001[$ that holds the instability point of the arrival process a. The second y denotes the difference between the lower bounds α of the basic and the modified CTMV algorithm (in multiples of 0.001, i.e., 0.1%).

The results for the Poisson process are in complete correspondence with the results obtained by Mathys and Flajolet [18]. When we focus on the result for $Q = 3$, we see that the MC was unstable for a load of 0.407. Mathys and Flajolet [18] showed that the actual stability point is 0.40697. This is another indication that the impact of the parameter d is indeed very small. Recall, the instability of the approximated MC always implies the instability of the exact MC. The only possible error exists in the fact that the approximated chain might become stable when the exact chain is not. This might happen when we choose a load that is fractionally larger than the actual stability point. The result for $Q = 3$ shows that this is not the case even if the difference between both values, i.e., the load and the stability point, is only 0.00003 or 0.003%.

Table 2 indicates that the impact of implementing the modified CTMV algorithm is more or less the same for each of the arrival processes, e.g., for $Q = 2$ the increment varies between 2.3% and 2.7%. Table 2 also confirms that it is hardly worthwhile to implement the modified CTMV algorithm for $Q > 3$. The fact that *doomed* slots occur less frequent, for large Q, is twofold. First, the probability that all colliding stations select the last group is smaller (we use fair coins). Second, even if all colliding stations select the last group, a *doomed* slot only occurs if the next $Q - 1$ slots are unused by new arrivals. Table 2 also indicates that there are arrival processes for which the modified binary CTMV algorithm outperforms the ternary one, e.g., $B([2], \centerdot)$.

7.4. Using Biased Coins. Mathys and Flajolet [18] have shown that the optimal biased coins, for the basic Q-ary CTMV algorithm under Poisson traffic, are fair

TABLE 3
The basic Q-ary CTMV algorithm.

$PP(\centerdot)$		$M(\centerdot, 0, 30, 30)$		$ER(\centerdot, 2)$	
p_0	$\alpha(\delta)$	p_0	$\alpha(\delta)$	p_0	$\alpha(\delta)$
0.6000	0.351	0.5500	0.343	0.6000	0.359
0.5500	0.358	0.5000	0.348	0.5500	0.364
0.5200	0.359	0.4800	0.349	0.5300	0.365 (21.1)
0.5100	0.360 (1.20)	0.4700	0.350 (0.47)	0.5200	0.365 (26.4)
0.5000	0.360 (2.28)	0.4650	0.350 (0.62)	0.5150	0.365 (27.0)
0.4900	0.360 (1.20)	0.4600	0.350 (0.60)	0.5100	0.365 (26.1)
0.4800	0.359	0.4500	0.350 (0.11)	0.5000	0.365 (19.9)
0.4500	0.358	0.4400	0.349	0.4800	0.364
0.4000	0.351	0.4200	0.348	0.4500	0.362

coins. For the modified scheme they demonstrated that the stability, under Poisson traffic, can be optimized by slightly increasing the probability of selecting the last group (the probability of selecting either one of the first $Q-1$ groups is identical). In this subsection, we investigate whether these results are also valid for other arrival processes. We restrict ourselves to the basic and the modified binary CTMV algorithm.

For each arrival process a considered, we vary the probabilities p_0 and $p_1 = 1 - p_0$ and determine the stability point that corresponds to the couple (a, p_0). When the stability point of different couples (a, p_0) lies within the same interval $]\alpha, \alpha + 0.001[$, we also add the drift δ to determine which value for p_0 performs best. This drift δ corresponds to the difference between the probability that the MC makes a transition to a parent node and the probability that the MC makes a transition to a child node, δ can be calculated from the matrices R_k presented in Section 6. Intuitively, a larger drift δ implies a more stable MC.

Table 3, resp. 4, represent the stability points, i.e., maximum throughput, as a function of p_0 for the basic, resp. modified, binary CTMV algorithm under Poisson input traffic, Markov modulated Poisson input traffic and Erlang input traffic. Both tables confirm the Poisson results obtained by Mathys and Flajolet [18].

Table 3 indicates that the optimal value for p_0 for the $ER(\centerdot, 2)$ lies somewhere in the interval $]0.51, 0.52[$, whereas the optimal value for the $M(\centerdot, 0, 30, 30)$ input traffic is found in the range $]0.46, 0.47[$. We already mentioned that the optimum for Poisson input is $p_0 = 0.5$. Thus, the burstier the input traffic the lower the optimal value of p_0 becomes. Intuitively, this can be understood as follows, the more bursty the input traffic becomes the better it is to postpone the retransmission of some of the colliding packets. Indeed, if a collision occurs, in slot t, under Erlang traffic it is more likely

TABLE 4
The modified Q-ary CTMV algorithm.

$PP(.)$		$M(.,0,30,30)$		$ER(.,2)$	
p_0	$\alpha(\delta)$	p_0	$\alpha(\delta)$	p_0	$\alpha(\delta)$
0.5000	0.387	0.5000	0.371	0.5000	0.394
0.4500	0.391	0.4500	0.379	0.4500	0.397
0.4300	0.392	0.4200	0.382	0.4400	0.397
0.4100	0.393 (2.27)	0.3800	0.384 (1.83)	0.4200	0.398 (8.27)
0.4068	0.393 (2.35)	0.3750	0.384 (1.89)	0.4175	0.398 (8.36)
0.4050	0.393 (2.32)	0.3700	0.384 (1.81)	0.4150	0.398 (8.23)
0.3800	0.392	0.3600	0.384 (1.25)	0.4100	0.398 (7.28)
0.3500	0.390	0.3400	0.383	0.3900	0.397

that no new arrivals will occur in the next slot, slot $t+1$, as opposed to the slots $t+i, i > 1$. Therefore, it is better to choose p_0 slightly larger than 0.5. Whereas for the Markov modulated traffic it is more likely that the D-BMAP is transmitting at a higher rate whenever a collision occurs and therefore it might be interesting to postpone more than half of the arrivals that occur during this high rate period to a period where a lower input rate is being used (i.e., the probability that new arrivals occur in slot $t+1$ is larger than in slot $t+i, i > 1$) . This line of reasoning also corresponds with the Poisson result: if a collision occurs in slot t, the probability of having a new arrival in slot $t+i$ is identical for all $i > 0$ ($= 1 - e^{-\lambda}$). Therefore, there is no reason to prefer the next slot above any of the other slots, i.e., $p_0 = 0.5$.

Table 4 confirms that burstier input traffic also results in a lower the optimal value of p_0 when the modified binary CTMV algorithm is used. However, for the modified algorithm the maximum throughput that can be achieved with biased coins differs more from the maximum throughput achieved with fair coins in comparison with the basic version of the CTMV algorithm. Moreover, the ranges of the optimal p_0 are very different from the ones that we found for the basic scheme (about 0.9 lower). This can be understood as follows. Selecting a smaller value for p_0 becomes more attractive because a lower penalty is paid when all the colliding stations select the last (second) group (because all the doomed slots are saved).

In conclusion, for bursty and correlated arrival patterns higher throughput results can be achieved by adapting p_0, especially if the modified scheme is used. It is however hard to predict the exact optimal value for p_0 (as it depends upon the specific nature of the arrival process).

8. Conclusions and Related Work. We have analyzed the maximum stable throughput of the basic and modified Q-ary CTMV algorithm with free access for both fair and biased coins by constructing several tree structured QBD MCs and by determining their stability. As opposed to any prior work, we did not restrict our study to Poisson arrival patterns, but considered a much more general class of input processes (D-BMAPs). We have shown, by means of numerical examples, that the binary and the ternary schemes should be preferred above higher splitting factors Q because they suffer much smaller throughput losses under bursty and correlated arrival traffic. Moreover, whenever possible, it is worth to exploit ternary feedback, i.e., implement the modified scheme, for a splitting factor $Q = 2$ or 3. We also demonstrated that it might be very useful to use biased coins when the input traffic is bursty and correlated.

Another important performance characteristic is the mean delay that is suffered when transmitting a packet. Using the QBD MCs that were constructed in this paper it is possible to calculate the mean delay and many other performance characteristics. We demonstrated this for the basic binary CTMV algorithm in [29]. In [29] we studied both the delay and throughput of the basic binary CTMV algorithm under D-BMAP input traffic.

The bit error ratio and capture effects are important characteristics of a wireless channel. It is fairly straightforward to see that one can extend the models presented in this paper in order to evaluate the CTMV algorithm when applied on a channel with markovian capture and errors. For instance, one could easily add the state of the channel as a part of the auxiliary variable of the tree structured QBD MCs. Other

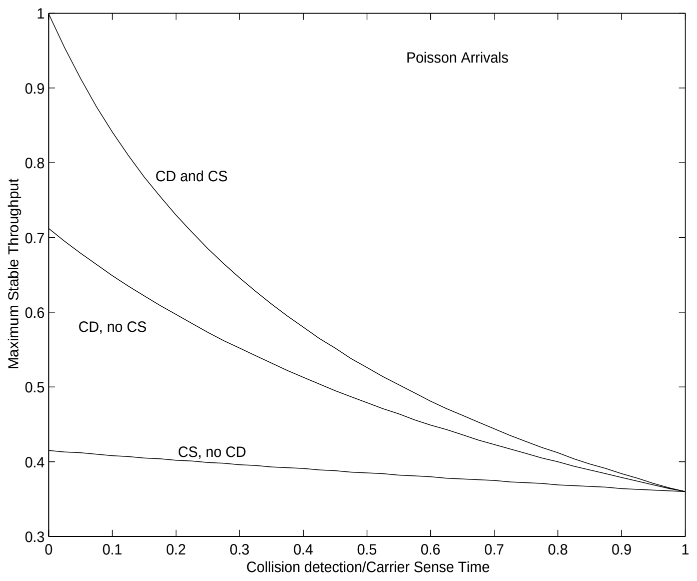


FIG. 2. *Maximum Stable Throughput of the basic binary CTMV algorithm with carrier-sensing and collision detection*

possible extensions are to create interaction between the channel feedback and the arrival process. When doing so one can show that even higher throughputs can be achieved by adapting the input rate in an appropriate manner. We can also shorten the length of the collision and empty slots to study the impact of carrier-sense and collision detection techniques. For instance, Figure 2 indicates the impact of having a carrier-sense (CS) and/or collision detection (CD) mechanism on the maximum stable throughput of the basic binary CTMV algorithm. The length of the carrier-sense window and collision detection times are expressed as a fraction of the packet length. Obviously, the faster we can identify empty and collision slots the better the throughput results. Figure 2 shows that implementing collision detection is more effective to increase the throughput in comparision with a carrier-sense mechanism.

Acknowledgement. B. Van Houdt is a postdoctoral fellow of the FWO-Flanders. The authors would like to acknowledge the reviewers for their helpful comments.

REFERENCES

[1] D. Bertsekas and R. Gallager, *Data Networks.* Prentice-Hall Int. Inc., 1992.

[2] D. A. Bini, G. Latouche, and B. Meini, *Solving nonlinear matrix equations arising in tree-like stochastic processes*, Linear Algebra Appl., 366(2003), pp. 39–64.

[3] C. Blondia, *A discrete-time batch markovian arrival process as B-ISDN traffic model*, Belgian Journal of Operations Research, Statistics and Computer Science, 32:3-4(1993), pp. 3–23.

[4] C. Blondia and O. Casals, *Statistical multiplexing of VBR sources: A matrix-analytical approach*, Performance Evaluation, 16(1992), pp. 5–20.

[5] C. Blondia and F. Geerts, *The correlation structure of the output of an ATM multiplexer*, In: Fifth IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, pages 235–250. Kluwer Academic Publ., 1999.

[6] O. Boxma, D. Denteneer, and J. Resing, *Delay models for contention trees in closed populations*, Performance Evaluation, 53(2003), pp. 169–185.

[7] J. I. Capetanakis, *The multiple access broadcast channel: Protocol and capacity considerations*, Ph.D. dissertation, MIT, also IEEE Trans. Inform. Theory, IT-25(1979), pp. 505–515.

[8] I. Cidon and M. Sidi *Conflict multiplicity estimation and batch resolution algorithms*, IEEE Trans. on Information Theory, IT-34:1(1988), pp. 101–110.

[9] A. Ephremides and B. Hajek, *Information theory and communication networks: an unconsummated union*, IEEE Transactions on Information Theory, 44:6(1998), pp. 2416–2434.

[10] G. Fayolle, P. Flajolet, M. Hofri, and P. Jacquet, *Analysis of a stack algorithm for random multiple-access communication*, IEEE Transactions on Information Theory, Vol. IT-21, No 2, March, 1985.

[11] P. Flajolet and P. Jacques, *Analytic models for tree communication protocols*, INRIA, Technical Report No 648, 1987.

[12] A. G. Greenberg, P. Flajolet, and R. E. Ladner, *Estimating the multiplicity of conficts to speed their resolution in mulitple access channels*, Journal of the Association for Computing Machinery, 34:2(1987), pp. 289–325.

[13] P. A. Humblet, *On the throughput of channel access algorithms with limited sensing*, IEEE Trans. on Comm., COM-34(1986), pp. 345–347.

[14] A.J.E.M. Janssen and M.J.M. de Jong, *Analysis of contention tree-algorithms*, IEEE Trans. on Information Theory, IT-46:6(2000), pp. 2163–2172.

[15] I. Kessler and M. Sidi, *Splitting algorithms in noisy channels with memory*, IEEE Transactions on Information Theory, IT-35:5(1989), pp. 1034–1043.

[16] A. Klemm, C. Lindemann, and M. Lohmann, *Modeling IP traffic using the batch markovian arrival process*, Performance Evaluation, 54(2003), pp. 149–173.

[17] D.M. Lucantoni, *New results on the single server queue with a batch markovian arrival process*, Stochastic Models, 7:1(1991), pp. 1–46.

[18] P. Mathys and P. Flajolet, *Q-ary collision resolution algorithms in random-access systems with free or blocked channel access*, IEEE Transactions on Information Theory, Vol it-32, no 2, March, 1985.

[19] M. L. Molle and G.C. Polyzos, *Conflict resolution algorithms and their performance analysis*, Technical report, University of Toronto, CS93-300, 1993.

[20] M.F. Neuts, *Structured Stochastic Matrices of M/G/1 type and their applications.* Marcel Dekker, Inc., New York and Basel, 1989.

[21] G.C. Polyzos and Molle, *Performance analysis of finite nonhomogeneous population tree conflict resolution algorithms using constant size window access*, IEEE Transactions on Communications, 35:11(1987), pp. 1124–1138.

[22] V. Ramaswami, *The generality of QBD processes*, 2nd Int. Conference on Matrix Analytical Methods, 1998.

[23] M. Seri and M. Sidi, *Splitting algorithms in channels with markovian capture*, European Transactions on Telecommunications and related Technologies, 5:1(1994), pp. 19–26.

[24] M. Seri and M. Sidi, *Splitting algorithms in channels with markovian capture*, European Transactions on Telecommunications and Related Technologies, 5:1(1994), pp. 19–26.

[25] J.A. Silvester, N. Fonseca, and S. Wang, *D-BMAP models for the performance evaluation of multimedia traffic in ATM networks*, Performance Modelling and Evaluation of ATM Networks, ed. Kouvatsos, IFIP Press, 1995.

[26] T. Takine, B. Sengupta, and R.W. Yeung, *A generalization of the matrix M/G/1 paradigm for markov chains with a tree structure*, Stochastic Models, 11:3(1995), pp. 411–421.

[27] A.S. Tanenbaum, *Computer Networks.* Prentice-Hall Int., Inc., 1989.

[28] B. S. Tsybakov and V.A. Mikhailov, *Free synchronous packet access in a broadcast channel with feedback*, Problemy Peredachi Inform, 14:4(1978), pp. 32–59.

[29] B. Van Houdt and C. Blondia, *Stability and performance of stack algorithms for random access communication modeled as a tree structured QBD Markov chain*, Stochastic Models, 17:3(2001), pp. 247–270.

[30] B. Van Houdt and C. Blondia, *Robustness of Q-ary collision resolution algorithms in random access systems*, Performance Evaluation, 57(2004), pp. 357–377.

[31] I. Widjaja and A.I. Elwalid, *Performance issues in VC-merge capable switches for multiprotocol label switching*, IEEE Journal on Selected Areas in Communications, 17:6(1999), pp. 1178–1189.

[32] R.W. Yeung and A.S. Alfa, *The quasi-birth-death type markov chain with a tree structure*, Stochastic Models, 15:4(1999), pp. 639–659.

[33] R.W. Yeung and B. Sengupta, *Matrix product-form solutions for markov chain with a tree structure*, Adv. Appl. Prob. 26(1994), pp. 965–987.