European Journal of Operational Research 201 (2010) 791–798

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Stochastics and Statistics

On the link between Markovian trees and tree-structured Markov chains

Sophie Hautphenne [a,*], Benny Van Houdt [b]

[a] Université Libre de Bruxelles, Boulevard du Triomphe, 1050 Brussels, Belgium
[b] University of Antwerp-IBBT, Middelheimlaan 1, 2020 Antwerpen, Belgium

ARTICLE INFO

Article history:
Received 17 July 2008
Accepted 28 March 2009
Available online 10 April 2009

Keywords:
Stochastic processes
Markovian multi-type branching processes
Markovian trees
Tree-like quasi-birth-and-death processes
Extinction probability
Newton's iteration

ABSTRACT

In this paper, we describe a link between Markovian binary trees (MBT) and tree-like quasi-birth-and-death processes (TLQBD) by associating a specific TLQBD to each MBT. The algorithms to compute the matrices G_k in the TLQBD then correspond to the algorithms calculating the extinction probability vector of the MBT. This parallelism leads to a new quadratic algorithm, based on the Newton iteration method, which converges to the extinction probability of an MBT.

We also present a one-to-one correspondence between a general Markovian tree (GMT) and a scalar tree-structured M/G/1-type Markov chain. This allows us to prove the equivalence between the main result on the positive recurrence, null recurrence or transience of a scalar tree-structured M/G/1-type Markov chain and the criticality of a GMT.

1. Introduction

This paper links the theory of Markovian trees to a class of discrete-time Markov chains with a tree-structured state space and a matrix product form solution. Markovian binary trees (MBTs) form a subclass of the general Markovian trees (GMTs) [12,2] and belong to a particular class of continuous-time Markovian multi-type branching process [1]. An MBT is initiated by a phase i branch for some $i \in \{1,\ldots,n\}$. After an exponential amount of time, three events can occur: (a) the branch dies, (b) the branch changes its phase to $j \in \{1,\ldots,n\}$, $j \neq i$, or (c) it splits into two branches with initial phases j and k, both in $\{1,\ldots,n\}$. Given their initial phases j and k, both the phase j and k branches (in case the latter exists) will evolve independently from each other (and from their parent phase i branch) in an analogous manner. That is, each remains within the same phase for an exponential amount of time, after which one of the above three events will take place. For a formal definition, we refer to Section 2.1.1. If, at some point, all of the branches have died, we state that the MBT is extinct. An important problem of MBTs exists in determining the extinction probability vector $\boldsymbol{q}$, where its ith entry holds the probability that an MBT starting in phase i becomes extinct. Several algorithms to compute $\boldsymbol{q}$ with linear convergence [12,2,5] and one with quadratic convergence based on a Newton iteration [6] have been developed.

Various types of tree-structured discrete-time Markov chains with a matrix product form solution have been introduced [25,17,24]. Our focus will go to two specific subclasses of these Markov chains: (i) tree-like Quasi-Birth–Death processes (TLQBD) [3,19] and (ii) scalar M/G/1-type Markov chains [8]. Formal definitions are given in Sections 2.1.2 and 4, respectively. The key in determining the steady state vector of the former exists in computing a set of matrices G_k, for which various algorithms have been proposed [24,3].

In this paper, we will demonstrate that we can associate a TLQBD to any MBT such that the computation of the set of G_k matrices of the TLQBD coincides with the computation of the extinction probability $\boldsymbol{q}$ of the MBT. We further show that the two natural fixed-point iterations for tree-structured QBDs [24], when applied to the associated TLQBD, give rise to the depth and (both) order algorithms for MBTs (see Section 2.1.1). The Newton iteration for TLQBDs however reduces to a different type of Newton iteration for MBTs than the one discussed in Hautphenne et al. [6]. We analyze the convergence properties of this novel Newton iteration, and we show that it needs less iterations than the existing Newton algorithm for MBTs.

Furthermore, we present a one-to-one relation between scalar tree-structured M/G/1-type Markov chains and GMTs. Using this relation we prove the equivalence between the main theorem on the criticality of GMTs [1, Section V.3, Theorem 2] and the main result on the ergodicity of scalar tree-structured M/G/1-type Markov chains [8].

The paper is structured as follows. Before we proceed with the introduction on MBTs and TLQBDs, we like to highlight some of the application areas of both paradigms. In Section 2 we start with a brief introduction on MBTs and TLQBDs. Then, we demonstrate how to associate a TLQBD to any MBT, we show the equivalence

* Corresponding author.
E-mail addresses: shautphe@ulb.ac.be (S. Hautphenne), benny.vanhoudt@ua.ac.be (B.V. Houdt).

0377-2217/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2009.03.052

between the various fixed-point iterations, and we generate the new Newton iteration for MBTs. In Section 3, we show how to obtain the new Newton iteration directly from the extinction equation for MBTs, and we discuss its convergence properties. Finally, Section 4 focusses on the link between GMTs and scalar tree-structured M/G/1-type Markov chains. All the vectors are supposed to be column vectors and all the entries of vectors and matrices are supposed to be real finite.

1.1. Applications of MBTs and TLQBDs

Continuous-time multi-type branching processes have applications in a large number of fields such as biology and epidemiology [4,11], but also in telecommunication systems [7,23]. For instance, the relationship between animal species can be represented on a tree diagram called a phylogenetic tree; Kontoleon [12] showed that many of the current models of the macroevolutionary process are subsumed by the MBT. Hautphenne et al. used the MBT to model the spread of a file in a peer-to-peer network and to compute the probability that the sharing process of this file eventually ends [7].

Tree-structured Markov chains have been used to study particular classes of queueing systems [25,17,10,9,21], as well as random access communication protocols [18,20,15]. Most of the work on queueing systems focussed on last-come-first-served service disciplines [25,17,10,9], but also includes more classical priority queues with first-come-first-served service [21,22]. The contributions on the random access protocols [18,20,15] are mainly concerned with the study of the maximum stable throughput of various classes of splitting and tree algorithms.

2. Markovian binary trees and tree-like QBD processes

2.1. Basic definitions and properties

2.1.1. Markovian binary trees

An MBT is characterized by an $n \times n$ matrix D_0, with entries $(D_0)_{i,j} \geqslant 0$ for $i \neq j$, an $n \times n^2$ matrix $\overline{B}$ with entries $\overline{B}_{i,jk} \geqslant 0$, for $1 \leqslant i,j,k \leqslant n$ and an $n \times 1$ vector $\bar{\boldsymbol{a}} \geqslant \mathbf{0}$. The diagonal entries of D_0 are such that $(-D_0)_{i,i}$ holds the positive parameter of the exponentially distributed sojourn time of a phase i branch. After this exponential sojourn time, the branch changes to phase $j \neq i$ with probability $(D_0)_{i,j}/(-D_0)_{i,i}$, splits into a left branch in phase j and a right branch in phase k with probability $\overline{B}_{i,jk}/(-D_0)_{i,i}$ or dies with probability $\bar{\boldsymbol{a}}_i/(-D_0)_{i,i}$. Hence, $(-D_0)_{i,i} = \sum_{j\neq i}(D_0)_{i,j} + \sum_{j,k}\overline{B}_{i,jk} + \bar{\boldsymbol{a}}_i$.

In other words, a branch that starts in phase i dies without splitting, with probability $a_i = ((-D_0)^{-1}\bar{\boldsymbol{a}})_i$, while it will eventually split into a left branch in phase j and a right branch in phase k with probability $B_{i,jk} = ((-D_0)^{-1}\overline{B})_{i,jk}$. Notice, the vector $\boldsymbol{a}$ and matrix B are such that $\mathbf{0} \leqslant \boldsymbol{a} \leqslant \mathbf{1}$, $\boldsymbol{a} \neq \mathbf{0}$, $B \geqslant 0$, and $\boldsymbol{a} + B\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ denotes a vector of 1's. An illustrative example of an MBT is depicted on Fig. 1.

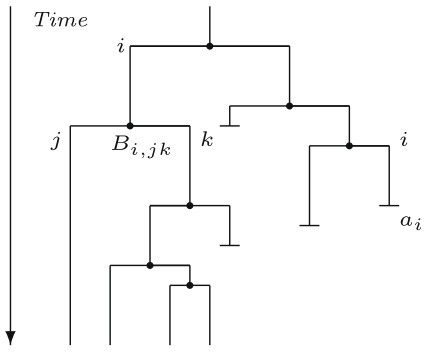

Fig. 1. The evolution of a Markovian binary tree over time.

We are interested, for instance, in determining the probability q_i that the tree originated by a branch in phase i dies completely. The $n \times 1$ vector $\boldsymbol{q}$ holding these probabilities is the smallest nonnegative solution to the extinction equation [5]

$$\boldsymbol{x} = \boldsymbol{a} + B(\boldsymbol{x} \otimes \boldsymbol{x}). \tag{1}$$

In Bean et al. [2], the authors analyze two linearly convergent algorithms to solve (1). The first one is named the *depth* algorithm and is obtained by using fixed-point iterations on (1). The second algorithm is called the *order* algorithm, and is based on two equivalent rewrites of (1) as

$$\boldsymbol{x} = [I - B(\boldsymbol{x} \otimes I)]^{-1}\boldsymbol{a} \tag{2}$$

$$= [I - B(I \otimes \boldsymbol{x})]^{-1}\boldsymbol{a}. \tag{3}$$

The order algorithm uses fixed-point iterations on (2) (*order*-1 algorithm), *or* on (3) (*order*-2 algorithm). A third linearly convergent algorithm, called the *thicknesses* algorithm, is presented in Hautphenne et al. [5]; it is obtained by using fixed-point iterations *alternatively* on (2) *and* on (3). It offers some advantages in that it better exploits possible dissymmetries in the structure of the matrix B. Finally, a quadratic algorithm [6], called the *Newton* algorithm, is obtained using Newton's iteration method on (1).

An MBT is called subcritical, supercritical or critical if the spectral radius of the nonnegative matrix $M = B(\mathbf{1} \otimes I + I \otimes \mathbf{1}) = B(\mathbf{1} \oplus \mathbf{1})$ is respectively strictly less than one, strictly greater than one, or equal to one (Athreya and Ney, [1, Section V.3, Theorem 2]). In the subcritical and critical cases, $\boldsymbol{q} = \mathbf{1}$, while in the supercritical case $\boldsymbol{q} \leqslant \mathbf{1}$, $\boldsymbol{q} \neq \mathbf{1}$. A useful property is when the MBT is *positive regular* [5]. This means that, for the underlying continuous-time process, the transition graph between the phases is irreducible. Then the matrix M has a single essential class and either $\boldsymbol{q} = \mathbf{1}$, or $\boldsymbol{q} = \mathbf{0}$, or $\mathbf{0} < \boldsymbol{q} < \mathbf{1}$.

2.1.2. Tree-like QBD processes

Consider a discrete-time bivariate Markov chain $\{(X_t, N_t), t \geqslant 0\}$ in which the values of X_t are represented by nodes of a d-ary tree, for $d \geqslant 2$, and where N_t takes integer values between 1 and m. We will refer to X_t as the *node* and to N_t as the *auxiliary* variable of the Markov chain at time t. The root node of the d-ary tree is denoted as $\emptyset$ and the remaining nodes are denoted as strings of integers, where each integer takes a value between 1 and d. For instance, the kth child of the root node is represented by k, the lth child of the node k by kl, and so on. Throughout this paper, we use the '+' to denote the concatenation on the right and the '-' to represent the deletion from the right. For example, if $J = k_1 k_2 \ldots k_n$, then $J + k = k_1 k_2 \ldots k_n k$. Let $f(J,k)$, for $J \neq \emptyset$, denote the k rightmost elements of the string J, then $J - f(J,1)$ represents the parent node of J.

The following restrictions need to apply for a Markov chain $\{(X_t, N_t), t \geqslant 0\}$ to be a tree-like QBD process (TLQBD). At each step the chain can only make a transition to its parent (i.e., $X_{t+1} = X_t - f(X_t,1)$, for $X_t \neq \emptyset$), to itself ($X_{t+1} = X_t$), or to one of its children ($X_{t+1} = X_t + s$ for some $1 \leqslant s \leqslant d$). Moreover, the state of the chain at time $t+1$ is determined as follows:

$$P[(X_{t+1}, N_{t+1}) = (J',j) | (X_t, N_t) = (J,i)] = \begin{cases} f^{i,j} & J' = J = \emptyset, \\ c^{i,j} & J' = J \neq \emptyset, \\ d_k^{i,j} & J \neq \emptyset, f(J,1) = k, J' = J - f(J,1), \\ u_s^{i,j} & J' = J + s, s = 1, \ldots, d, \\ 0 & \text{otherwise.} \end{cases}$$

Remark that the transition probabilities between two nodes depend only on the spacial relationship between the two nodes and not on their specific values.

We can now define the $m \times m$ matrices D_k, C, F and U_s with respective (i,j)th elements given by $d_k^{i,j}$, $c^{i,j}$, $f^{i,j}$ and $u_s^{i,j}$, for $k,s = 1,\dots,d$. This completes the description of the TLQBD; such a process is fully characterized by the matrices D_k, C, U_s and F.

Next, we introduce a number of matrices that play a crucial role when studying the stability and stationary behavior of a TLQBD. The fundamental period of a TLQBD starting in state $(J+k,i)$ is defined as the first passage time from the state $(J+k,i)$ to one of the states (J,j), for $j = 1,\dots,m$. Let G_k, for $1 \leqslant k \leqslant d$, denote the matrix whose (i,v)th element is the probability that the Markov chain is in state (J,v) at the end of a fundamental period which started in state $(J+k,i)$. Let V denote the matrix whose $(i,v)^{th}$ element is the taboo probability that starting from state $(J+k,i)$, the process eventually returns to node $J+k$ by visiting $(J+k,v)$, under the taboo of the node J. Note that, due to the restrictions on the transition probabilities, the matrix V does not depend on k. Yeung and Alfa [24] showed that the following expressions hold for these matrices:

$$G_k = (I-V)^{-1}D_k, \tag{4}$$

$$V = C + \sum_{s=1}^{d} U_s G_s.$$

Combining these equations, we have the following relation:

$$V = C + \sum_{s=1}^{d} U_s (I-V)^{-1} D_s.$$

2.2. The link between MBTs and TLQBDs

In this section we show how to associate a TLQBD to an MBT characterized by a birth matrix B and a death vector $\boldsymbol{a}$, such that the computation of the G_k-matrices of the TLQBD process coincides with the computation of $\boldsymbol{q}$ in the MBT.

Let $(\Omega, \mathscr{A}, \mathbb{P})$ be a probability space. The MBT with characteristics $\boldsymbol{a}$ and B is a random process which associates to each $\omega \in \Omega$ a specific MBT realization (that is, a certain binary tree picture). This realization of the MBT may be finite or infinite. What we call the extinction probability $\boldsymbol{q}$ of an MBT, is actually the measure $\mathbb{P}$ of the subset of Ω containing all the ω corresponding to a finite realization of an MBT.

Now, we shall associate a path in the TLQBD corresponding to a given realization of an MBT, in such a way that, if the MBT is finite (respectively infinite), then the first passage time from the initial state, let us say $(J+k,i)$, in the associated TLQBD, to one of the parental states (J,j), is finite (respectively infinite).

For a given realization of an MBT, let us visit all the edges and vertices of the binary tree in a *pre-order* way, that is, a *depth-first* manner. With each step of this exploration, we will associate a transition in the TLQBD. As mentioned above, if the MBT is finite, then, starting from its root, we will cover all the nodes of the tree in a finite time; this means that in the TLQBD, we will return to the parent of the starting node in a finite time. We will consider two depth-first traversals: one where we first explore the *left* side of the tree and a second where the *right* is explored first.[1]

[1] Other traversals are also possible, for instance, we might explore either the right or left branch first depending on the phase i of the parent. Various numerical experiments seem to indicate that exploring the right branch first for phase i parents whenever the ith entry of $(I - B(I \otimes \boldsymbol{a}))^{-1}\boldsymbol{a}$ is larger than or equal to the ith entry of $(I - B(\boldsymbol{a} \otimes I))^{-1}\boldsymbol{a}$, results in a linear algorithm that outperforms the depth, both order and the thicknesses algorithms.

The idea of the correspondence between the path in the MBT and that in the TLQBD is the following: whenever a branch splits into a left and right branch in the MBT, if we use the left (respectively right) traversal, then we freeze the evolution of the right (respectively left) branch, by storing its initial phase in the node variable of the TLQBD, and we allow the left (respectively right) branch to evolve. Whenever a branch dies in the MBT, we unfreeze the last branch that was frozen during a split operation. Thus, a right (respectively left) branch remains frozen as long as the tree originating from its left (respectively right) sibling is still alive. As the tree-like process is a discrete-time process, we will only observe the depth-first evolution of the tree at the split and death instances. This correspondence is illustrated in Fig. 2 for the left traversal of the MBT.

Let us now construct the TLQBD $\{(X_t, N_t), t \geqslant 0\}$ with $d = m = n$ to further clarify this idea. Define

$$(U_k^{(l)})_{i,j} = B_{i,jk},$$

for the left traversal and

$$(U_j^{(r)})_{i,k} = B_{i,jk},$$

for the right one, meaning when a phase i branch splits into a left branch in phase j and a right branch in phase k, then we may forget the phase i of the parent, and if we use the left traversal, then we add the phase k of the right branch (respectively the phase j of the left branch for the right traversal) to the node variable X_t (where it remains stored for later use) and set the auxiliary variable N_{t+1} equal to the phase j of the left branch (respectively equal to the phase k of the right branch), which is now the current branch. Also, note that we can write $B = [U_1^{(r)} U_2^{(r)} \dots U_m^{(r)}]$. Further, define

$$D_k = D_k^{(l)} = D_k^{(r)} = \boldsymbol{a}\boldsymbol{e}_k,$$

where $\boldsymbol{e}_k$ is a $1 \times m$ vector with all its entries equal to zero except for its kth entry which equals 1. Thus, when a branch dies, we forget its phase, remove the rightmost integer $f(X_t, 1) = k$ of X_t and set it equal to the auxiliary variable N_{t+1}. It means that we simply move to the next branch of the depth-first traversal, which is the one we most recently stored in X_t. Finally, the C matrix of the TLQBD is set equal to zero.

Due to the specific structure of the D_k matrices, and Eq. (4), we immediately have the following structure for the G_k-matrices

$$G_k = \boldsymbol{q}\boldsymbol{e}_k, \quad k = 1,2,\dots,m$$

for $\boldsymbol{q} = (I-V)^{-1}\boldsymbol{a}$. Hence, computing G_k for all k is reduced to computing a single $m \times 1$ vector $\boldsymbol{q}$. By carefully looking at the stochastic interpretation of the G_k matrices and the construction above, one

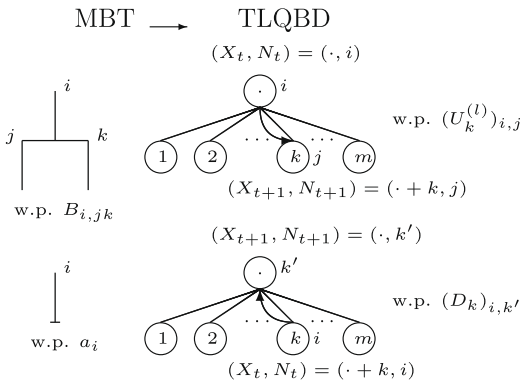

Fig. 2. The correspondence between possible transitions in an MBT and the corresponding transitions in the TLQBD, for the left traversal of the MBT (w.p. stands for *with probability*).

can also see that the ith entry of $\boldsymbol{q}$, is identical to the extinction probability of an MBT originating from a phase i branch, so that $\boldsymbol{q}$ is exactly the extinction probability vector defined in Section 2.1.1. This can also be derived algebraically as shown below.

2.3. Equivalence of the algorithms

Next, we demonstrate that the standard iteration for the G_k-matrices in TLQBDs, of both the left and right traversals, corresponds to the depth algorithm for MBTs, while the V oriented algorithm coincides with the two order algorithms. Finally, we look at the sequence corresponding to the Newton iteration for TLQBDs developed in Bini et al. [3]. This will be the starting point of the next section.

The standard G_k-iteration starts with $G_{k,0} = D_k$ for all k and computes $G_{k,n+1}$ as

$$G_{k,n+1} = D_k + CG_{k,n} + \left(\sum_{j=1}^{m} U_j G_{j,n}\right) G_{k,n}.$$

By plugging in the specific structure of the $G_{k,n} = \boldsymbol{x}_n \boldsymbol{e}_k$ matrices and the fact that $C = 0$, we find $\boldsymbol{x}_0^{(l)} \boldsymbol{e}_k = \boldsymbol{a}\boldsymbol{e}_k$, and for $n \geqslant 0$,

$$\boldsymbol{x}_{n+1}^{(l)} \boldsymbol{e}_k = \left(\boldsymbol{a} + B(\boldsymbol{x}_n^{(l)} \otimes I)\boldsymbol{x}_n^{(l)}\right)\boldsymbol{e}_k,$$

as $\sum_j U_j^{(l)} \boldsymbol{y}\boldsymbol{e}_j = B(\boldsymbol{y} \otimes I)$ for any $\boldsymbol{y} \in \mathbb{R}^m$, and

$$\boldsymbol{x}_{n+1}^{(r)} \boldsymbol{e}_k = \left(\boldsymbol{a} + B(I \otimes \boldsymbol{x}_n^{(r)})\boldsymbol{x}_n^{(r)}\right)\boldsymbol{e}_k,$$

as $\sum_j U_j^{(r)} \boldsymbol{y}\boldsymbol{e}_j = B(I \otimes \boldsymbol{y})$. In other words, we find that the two sequences $\boldsymbol{x}_n^{(r)}$ and $\boldsymbol{x}_n^{(l)}$ of vectors are identical and computed according to the depth algorithm for the extinction probability of MBTs.

The V-iteration starts by letting $V_0 = C$ and continues as

$$V_{n+1} = C + \sum_{j=1}^{m} U_j (I - V_n)^{-1} D_j.$$

By denoting $(I - V_n^{(l)})^{-1}\boldsymbol{a}$ as $\boldsymbol{x}_n^{(l)}$, meaning $\boldsymbol{x}_0^{(l)} = \boldsymbol{a}$ (as $C = 0$), we find

$$V_{n+1}^{(l)} = \sum_{j=1}^{m} U_j^{(l)} \boldsymbol{x}_n^{(l)} \boldsymbol{e}_j = B(\boldsymbol{x}_n^{(l)} \otimes I),$$

which means that $\boldsymbol{x}_{n+1}^{(l)} = (I - V_{n+1}^{(l)})^{-1}\boldsymbol{a}$ equals

$$\boldsymbol{x}_{n+1}^{(l)} = (I - B(\boldsymbol{x}_n^{(l)} \otimes I))^{-1}\boldsymbol{a}.$$

This iteration is identical to the *order*-1 algorithm for MBTs developed in Kontoleon [12]. Similarly, we find $\boldsymbol{x}_0^{(r)} = \boldsymbol{a}$ and

$$\boldsymbol{x}_{n+1}^{(r)} = (I - B(I \otimes \boldsymbol{x}_n^{(r)}))^{-1}\boldsymbol{a},$$

which is the *order*-2 algorithm.

The algorithms presented in Bini et al. [3] for computing the set of G_k matrices of a TLQBD also include a Newton iteration. This iteration determines the smallest nonnegative solution

$$\overline{G} = \begin{bmatrix} G_1 \\ \vdots \\ G_d \end{bmatrix}$$

of $\mathscr{F}(\overline{G}) = 0$, by setting $\overline{G}_0 = 0$ and solving

$$\overline{G}_{n+1} = \overline{G}_n - \left(\mathscr{F}'_{\overline{G}_n}\right)^{-1} \mathscr{F}\left(\overline{G}_n\right), \tag{5}$$

where $\mathscr{F}$ is the following mapping from $\mathbb{R}^{md\times m}$ to $\mathbb{R}^{md\times m}$:

$$\mathscr{F}\begin{bmatrix} X_1 \\ \vdots \\ X_d \end{bmatrix} = \begin{bmatrix} X_1 \\ \vdots \\ X_d \end{bmatrix} - \left(I \otimes \left(I - \left(C + \sum_{j=1}^{d} U_j X_j\right)\right)^{-1}\right) \begin{bmatrix} D_1 \\ \vdots \\ D_d \end{bmatrix},$$

and $\mathscr{F}'_X Y$ is the image of Y under the Fréchet derivative[2] of $\mathscr{F}$ at X. As discussed in Bini et al. [3], each step of the Newton iteration either requires an iterative procedure or the solution of a linear system of m^2 equations and m^2 unknowns. For the TLQBDs stemming from the MBTs, a single step can be performed more easily by exploiting the structure of the matrices involved. Let us focus on the right traversal of the branches. First note that $\overline{G} = (I \otimes \boldsymbol{q})$, $\overline{D} := [D_1, D_2, \ldots, D_d]' = (I \otimes \boldsymbol{a})$ and let us take $\overline{G}_0 = (I \otimes \boldsymbol{a}) = (I \otimes \boldsymbol{x}_0)$ (indeed, we can show that the Newton sequence is still convergent when $G_0 = (I \otimes (I - C)^{-1})\overline{D} = -\mathscr{F}(0)$). Further, when we look at the image of $(I \otimes \boldsymbol{x})$ with $\boldsymbol{x} \in \mathbb{R}^m$ under $\mathscr{F}$, using $\overline{D} = (I \otimes \boldsymbol{a})$ and $C = 0$, we find

$$\mathscr{F}(I \otimes \boldsymbol{x}) = I \otimes \left(\boldsymbol{x} - \left(I - \sum_{j=1}^{d} U_j^{(r)} \boldsymbol{x}\boldsymbol{e}_j\right)^{-1} \boldsymbol{a}\right) = I \otimes (\boldsymbol{x} - S_{\boldsymbol{x}}\boldsymbol{a}),$$

with $S_{\boldsymbol{x}} = (I - B(I \otimes \boldsymbol{x}))^{-1}$. The image of $(I \otimes \boldsymbol{z})$ under $\mathscr{F}'$ at $(I \otimes \boldsymbol{x})$ equals

$$\mathscr{F}'_{(I\otimes \boldsymbol{x})}(I \otimes \boldsymbol{z}) = I \otimes (\boldsymbol{z} - S_{\boldsymbol{x}} B(I \otimes \boldsymbol{z}) S_{\boldsymbol{x}}\boldsymbol{a}) = I \otimes (I - S_{\boldsymbol{x}} B(S_{\boldsymbol{x}}\boldsymbol{a} \otimes I))\boldsymbol{z}.$$

Therefore, Eq. (5), with $\overline{G}_n = (I \otimes \boldsymbol{x}_n)$, reduces to

$$\begin{aligned}(I \otimes \boldsymbol{x}_{n+1}) = \overline{G}_{n+1} &= (I \otimes \boldsymbol{x}_n) - (\mathscr{F}'_{(I\otimes \boldsymbol{x}_n)})^{-1} \mathscr{F}(I \otimes \boldsymbol{x}_n) \\ &= I \otimes (\boldsymbol{x}_n - (I - S_{\boldsymbol{x}_n} B(S_{\boldsymbol{x}_n}\boldsymbol{a} \otimes I))^{-1}(\boldsymbol{x}_n - S_{\boldsymbol{x}_n}\boldsymbol{a})).\end{aligned}$$

Hence, the Newton iteration of Bini et al. [3] reduces to the simple iteration

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - (I - S_{\boldsymbol{x}_n} B(S_{\boldsymbol{x}_n}\boldsymbol{a} \otimes I))^{-1}(\boldsymbol{x}_n - S_{\boldsymbol{x}_n}\boldsymbol{a}) \tag{6}$$

to determine the extinction probability vector $\boldsymbol{q}$ of the MBT, provided that the necessary inverse matrices exist. The convergence of this iteration follows indirectly from the convergence result in Bini et al. [3] and the link established above, and is discussed in the next section. More importantly, we also show that the new iteration converges at least as fast as the Newton iteration introduced in Hautphenne et al. [6].

3. A faster Newton algorithm for MBTs

One might also have obtained Eq. (6) directly from a reformulation of the extinction equation for MBTs. We first recall the way the Newton algorithm developed in Hautphenne et al. [6] for MBTs has been obtained, then we derive the new Newton sequence. Subsequently, we show that the convergence rate is better than that of the former Newton algorithm.

3.1. A direct derivation of the Newton sequences

In what follows, we suppose that the MBT is positive regular, which implies that either $\boldsymbol{q} = \mathbf{1}$, or $\boldsymbol{q} < \mathbf{1}$. We may also assume that the MBT is supercritical. Otherwise, the extinction probability is exactly equal to the vector $\mathbf{1}$ and we do not need algorithms to solve the extinction equation. We denote by $\langle \boldsymbol{u}, \boldsymbol{v}\rangle$ the set of vectors $\boldsymbol{x}$ such that $\boldsymbol{u} \leqslant \boldsymbol{x} \leqslant \boldsymbol{v}$, where we use the natural partial order such that $\boldsymbol{x} \leqslant \boldsymbol{y}$ if $x_i \leqslant y_i$ for all i.

In Hautphenne et al. [6], the authors develop a quadratic algorithm to compute the extinction probability of an MBT. This algorithm is based on Newton's iteration method for the solution

[2] The mapping $F : D \subset \mathbb{R}^n \to \mathbb{R}^m$ is *Fréchet-differentiable* at $x \in \text{int}(D)$ if there exists a linear operator A from $\mathbb{R}^n$ to $\mathbb{R}^m$ such that

$$\lim_{h\to 0}(1/||h||)||F(x + h) - F(x) - Ah|| = 0.$$

The linear operator A is denoted by F'_x and is called the *Fréchet derivative* of F at x (see Ortega and Rheinboldt [14]).

of Eq. (1) rewritten as $\boldsymbol{x} - \boldsymbol{a} - B(\boldsymbol{x} \otimes \boldsymbol{x}) = \boldsymbol{0}$. The quadratic algorithm obtained goes as follows: $\boldsymbol{x}_0 = \boldsymbol{a}$ and for $k \geqslant 1$,

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - [I - B(\boldsymbol{x}_k \oplus \boldsymbol{x}_k)]^{-1}[\boldsymbol{x}_k - \boldsymbol{a} - B(\boldsymbol{x}_k \otimes \boldsymbol{x}_k)]. \tag{7}$$

Now, observe that Eq. (1) may be equivalently rewritten as $\mathscr{F}(\boldsymbol{x}) = \boldsymbol{0}$, with

$$\mathscr{F}(\boldsymbol{x}) = \boldsymbol{x} - [I - B(I \otimes \boldsymbol{x})]^{-1}\boldsymbol{a}, \tag{8}$$

via (3).

For any matrix norm, $\mathbb{R}^m$ is a Banach space, and the function $\mathscr{F}$ is a mapping from $\mathbb{R}^m$ into itself. The Fréchet derivative of $\mathscr{F}$ at $\boldsymbol{x}$ is a linear map $\mathscr{F}'_{\boldsymbol{x}} : \mathbb{R}^m \to \mathbb{R}^m$ given by

$$\mathscr{F}'_{\boldsymbol{x}} : \boldsymbol{z} \mapsto [I - S_{\boldsymbol{x}} B(S_{\boldsymbol{x}}\boldsymbol{a} \otimes I)]\boldsymbol{z} = [\boldsymbol{z} - S_{\boldsymbol{x}} B(S_{\boldsymbol{x}}\boldsymbol{a} \otimes \boldsymbol{z})],$$

with $S_{\boldsymbol{x}} = [I - B(I \otimes \boldsymbol{x})]^{-1}$, as in the previous section.

For a given $\boldsymbol{x}_0$, the new Newton sequence for the solution of $\mathscr{F}(\boldsymbol{x}) = \boldsymbol{0}$ is

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{x}_k - (\mathscr{F}'_{\boldsymbol{x}_k})^{-1}\mathscr{F}(\boldsymbol{x}_k) \\ &= \boldsymbol{x}_k - [I - S_{\boldsymbol{x}_k} B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes I)]^{-1}[\boldsymbol{x}_k - S_{\boldsymbol{x}_k}\boldsymbol{a}]\end{aligned} \tag{9}$$

for $k = 0, 1, \ldots$, provided that $\mathscr{F}'_{\boldsymbol{x}_k}$ is invertible for all k. Notice, this sequence is identical to the one given in Eq. (6).

We can easily show that the inverse $[I - S_{\boldsymbol{x}_k} B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes I)]^{-1}$ exists for all $\boldsymbol{x}_k \leqslant \boldsymbol{q}$, by using an interpretation of the equation $\mathscr{F}(\boldsymbol{x}) = 0$ as an extinction equation for a new branching process embedded in the MBT: the leftmost branch of the MBT represents the lifetime of the first particle of this branching process, and each right branch coming from the leftmost branch represents a new child from the initial particle, which may itself generate new particles, and so on. For this process, we see that the progeny generating function of a particle is given by

$$\boldsymbol{G}(\boldsymbol{x}) = \sum_{n \geqslant 0} [B(I \otimes \boldsymbol{x})]^n \boldsymbol{a} = [I - B(I \otimes \boldsymbol{x})]^{-1}\boldsymbol{a}.$$

Then, we repeat the proof of Theorem 5.1 in Hautphenne et al. [5] with this new progeny generating function, and in the supercritical case, we get $\mathrm{sp}[S_{\boldsymbol{q}} B(S_{\boldsymbol{q}}\boldsymbol{a} \otimes I)] < 1$. This implies that for all $\boldsymbol{x}_k$ in $\langle \boldsymbol{0}, \boldsymbol{q} \rangle$,

$$\mathrm{sp}[S_{\boldsymbol{x}_k} B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes I)] < 1$$

since $S_{\boldsymbol{x}_k} \leqslant S_{\boldsymbol{q}}$, and thus $S_{\boldsymbol{x}_k} B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes I) \leqslant S_{\boldsymbol{q}} B(S_{\boldsymbol{q}}\boldsymbol{a} \otimes I)$.

For any $\boldsymbol{x}_0$ in $\langle \boldsymbol{0}, \boldsymbol{a} \rangle$, the new Newton sequence (9) is well defined, is monotone increasing, and converges (at least) quadratically to the extinction probability $\boldsymbol{q}$ of the MBT. To prove these properties, we can follow the same idea as in Latouche [13], Bini et al. [3], and Hautphenne et al. [6], and adapt the series of preliminary lemmas as well as their proof in the context of MBTs. Another way to be convinced is by the link established between TLQBDs and MBTs: the new Newton sequence for MBTs inherits all the properties of the Newton algorithm analyzed in Bini et al. [3]. For these reasons, and the sake of brevity, the direct proof of the quadratic convergence of the new sequence in the MBT context will be omitted in the present paper.

3.2. Comparison between the two Newton algorithms for MBTs

Let us write $\tilde{\boldsymbol{x}}_k$ for the approximations of the Newton algorithm (7) from Hautphenne et al. [6], to distinguish from these of the new Newton algorithm (9).

Let us denote the difference between the kth iteration in the two Newton algorithms by $\Delta_k = \boldsymbol{x}_k - \tilde{\boldsymbol{x}}_k$. Let us show that at each iteration, the new Newton algorithm is closer to the solution $\boldsymbol{q}$ than the first algorithm, which implies that the new algorithm converges faster towards the solution than the former in terms of the number of iterations.

Proposition 3.1. *The difference Δ_k is nonnegative for all $k \geqslant 0$.*

Proof. The proof goes by induction. First, $\Delta_0 = \boldsymbol{x}_0 - \tilde{\boldsymbol{x}}_0 = \boldsymbol{a} - \boldsymbol{a} = \boldsymbol{0}$. Now, suppose that $\Delta_k \geqslant \boldsymbol{0}$. Let us show that we still have $\Delta_{k+1} \geqslant \boldsymbol{0}$.

The new Newton sequence may be rewritten as

$$\boldsymbol{x}_{k+1} = S_{\boldsymbol{x}_k} B_{\boldsymbol{x}_{k+1}} S_{\boldsymbol{x}_k}\boldsymbol{a} - S_{\boldsymbol{x}_k} B_{\boldsymbol{x}_k} S_{\boldsymbol{x}_k}\boldsymbol{a} + S_{\boldsymbol{x}_k}\boldsymbol{a} \tag{10}$$

by multiplying both sides of Eq. (9) on the left by $[I - S_{\boldsymbol{x}_k} B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes I)]$. Multiplying both sides of Eq. (10) on the left by $(I - B(I \otimes \boldsymbol{x}_k))$ yields

$$\boldsymbol{x}_{k+1} = B(\boldsymbol{x}_{k+1} \otimes \boldsymbol{x}_k) + B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes \boldsymbol{x}_{k+1}) - B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes \boldsymbol{x}_k) + \boldsymbol{a}. \tag{11}$$

By doing the same for the Newton algorithm from Hautphenne et al. [6], we have

$$\tilde{\boldsymbol{x}}_{k+1} = B(\tilde{\boldsymbol{x}}_k \otimes \tilde{\boldsymbol{x}}_{k+1}) + B(\tilde{\boldsymbol{x}}_{k+1} \otimes \tilde{\boldsymbol{x}}_k) - B(\tilde{\boldsymbol{x}}_k \otimes \tilde{\boldsymbol{x}}_k) + \boldsymbol{a}. \tag{12}$$

So, by subtracting Eq. (12) from Eq. (11) we obtain

$$\begin{aligned}\Delta_{k+1} =& B(\boldsymbol{x}_{k+1} \otimes \boldsymbol{x}_k) + B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes (\boldsymbol{x}_{k+1} - \boldsymbol{x}_k)) - B(\tilde{\boldsymbol{x}}_{k+1} \otimes \tilde{\boldsymbol{x}}_k) \\ & - B(\tilde{\boldsymbol{x}}_k \otimes (\tilde{\boldsymbol{x}}_{k+1} - \tilde{\boldsymbol{x}}_k)).\end{aligned}$$

By adding and subtracting the terms $B(\boldsymbol{x}_{k+1} \otimes \tilde{\boldsymbol{x}}_k)$ and $B(S_{\boldsymbol{x}_k}\boldsymbol{a} \otimes (\tilde{\boldsymbol{x}}_{k+1} - \tilde{\boldsymbol{x}}_k))$, and regrouping the terms, we get

$$\begin{aligned}\Delta_{k+1} =& [B(S_{\boldsymbol{x}_k}\boldsymbol{a} \oplus \tilde{\boldsymbol{x}}_k)]\Delta_{k+1} + B[(\boldsymbol{x}_{k+1} - S_{\boldsymbol{x}_k}\boldsymbol{a}) \otimes I]\Delta_k \\ & + B[(S_{\boldsymbol{x}_k}\boldsymbol{a} - \tilde{\boldsymbol{x}}_k) \otimes (\tilde{\boldsymbol{x}}_{k+1} - \tilde{\boldsymbol{x}}_k)] \\ =& \underbrace{[I - B(S_{\boldsymbol{x}_k}\boldsymbol{a} \oplus \tilde{\boldsymbol{x}}_k)]^{-1}}_{(I)} \Bigg\{ B[\underbrace{(\boldsymbol{x}_{k+1} - S_{\boldsymbol{x}_k}\boldsymbol{a})}_{(II)} \otimes I] \underbrace{\Delta_k}_{(III)} \\ & + B[\underbrace{(S_{\boldsymbol{x}_k}\boldsymbol{a} - \tilde{\boldsymbol{x}}_k)}_{(IV)} \otimes \underbrace{(\tilde{\boldsymbol{x}}_{k+1} - \tilde{\boldsymbol{x}}_k)}_{(V)}] \Bigg\} \geqslant \boldsymbol{0}.\end{aligned}$$

Indeed, (I) is a positive matrix, since it is equal to a infinite sum of positive terms. (II) is positive by Eq. (10) and the increase of the new Newton sequence. (III) is positive by induction assumption. (IV) is positive since $\tilde{\boldsymbol{x}}_k \leqslant \boldsymbol{x}_k$ by induction assumption, and because $\boldsymbol{x}_k \leqslant S_{\boldsymbol{x}_k}\boldsymbol{a}$ for all $k \geqslant 0$. Indeed, in the proof of the quadratic convergence of the first Newton sequence (Theorem 2.1 in Hautphenne et al. [6]), it is notably shown that if $\boldsymbol{0} \leqslant \tilde{\boldsymbol{x}}_0 \leqslant \boldsymbol{a}$, then $\widetilde{\mathscr{F}}(\tilde{\boldsymbol{x}}_k) \leqslant 0$, for all k (where $\widetilde{\mathscr{F}}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{a} - B(\boldsymbol{x} \otimes \boldsymbol{x})$). Here, the same argument may be used to show that the new Newton sequence also satisfies $\mathscr{F}(\boldsymbol{x}_k) \leqslant 0$, for all k, whenever $\boldsymbol{0} \leqslant \boldsymbol{x}_0 \leqslant \boldsymbol{a}$. Finally, (V) is positive by the increasing nature of the first Newton sequence. □

3.3. Other alternatives

We could as well have applied the Newton iteration method on the equation $\mathscr{F}(\boldsymbol{x}) = \boldsymbol{0}$, where

$$\mathscr{F}(\boldsymbol{x}) = \boldsymbol{x} - [I - B(\boldsymbol{x} \otimes I)]^{-1}\boldsymbol{a}.$$

In that case, the Fréchet derivative of $\mathscr{F}$ at $\boldsymbol{x}$ is a linear map $\mathscr{F}'_{\boldsymbol{x}}$ given by

$$\mathscr{F}'_{\boldsymbol{x}} : \boldsymbol{z} \mapsto [I - T_{\boldsymbol{x}} B(I \otimes T_{\boldsymbol{x}}\boldsymbol{a})]\boldsymbol{z} = [\boldsymbol{z} - T_{\boldsymbol{x}} B(\boldsymbol{z} \otimes T_{\boldsymbol{x}}\boldsymbol{a})],$$

where $T_{\boldsymbol{x}} = [I - B(\boldsymbol{x} \otimes I)]^{-1}$.

These two alternate Newton algorithms may be seen as "Newton-order" sequences, as the structure of $\mathscr{F}(\boldsymbol{x}) = \boldsymbol{0}$ reminds us of the structure of the two order algorithms, while the Newton algorithm described in Hautphenne et al. [6] may be seen as a "Newton-depth" sequence. We can also construct the "Newton-thicknesses" sequence, by applying the Newton iteration alternatively on the two equations

$$\mathscr{F}(\boldsymbol{x}) = \boldsymbol{x} - [I - B(I \otimes \boldsymbol{x})]^{-1}\boldsymbol{a} \quad \text{and} \quad \mathscr{F}(\boldsymbol{x}) = \boldsymbol{x} - [I - B(\boldsymbol{x} \otimes I)]^{-1}\boldsymbol{a}.$$

In terms of convergence rate expressed in the number of iterations, the fastest Newton algorithm among the four presented above depends on the example under consideration. Furthermore, in general, we may *not* assume that if one of the linear algorithms, let us say A, is better than another one, B, for one particular example, then the corresponding Newton-A algorithm will converge faster than the corresponding Newton-B algorithm on the same example. This is illustrated on Fig. 3, where, taking Example 1 with one parameter δ and $n = 9$ from Hautphenne et al. [6], we plotted the number of iterations needed to compute $\boldsymbol{q}$ for each linear and each Newton algorithm, as a function of the parameter δ. We see that the thicknesses algorithm is the best linear algorithm to compute $\boldsymbol{q}$, but the Newton-thicknesses algorithm does not perform very well, it is even worse than the Newton-depth algorithm on some occasions.

Remark 3.2. Our algorithms were optimized by using the particular structure of the matrices involved at each iteration, in particular the Kronecker product with the identity matrix. We obtain a complexity of $(20/3)n^3 + \mathcal{O}(n^2)$ flops per iteration for the Newton-depth algorithm, against $8n^3 + \mathcal{O}(n^2)$ flops for the three other Newton algorithms. We thus expect the Newton-depth algorithm to be somewhat faster than the other algorithms, even if it requires one or two iterations more. This is confirmed when looking at Fig. 4 where we plotted the CPU time (with Intel 2.4 GHz) for the Newton-depth and the Newton-order 2 algorithm (the curve for the Newton-order 1 and the Newton-thicknesses algorithms almost coincide with that of the Newton-order 2 algorithm).

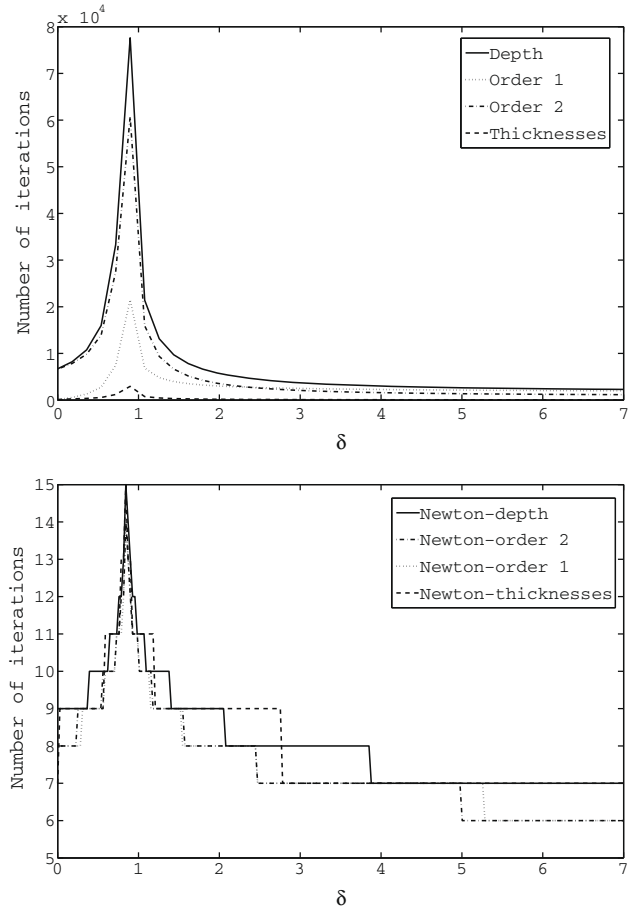

Fig. 3. Number of iterations needed to compute $\boldsymbol{q}$ as a function of a parameter δ, with the four linear algorithms (top), and with the four Newton algorithms (bottom).

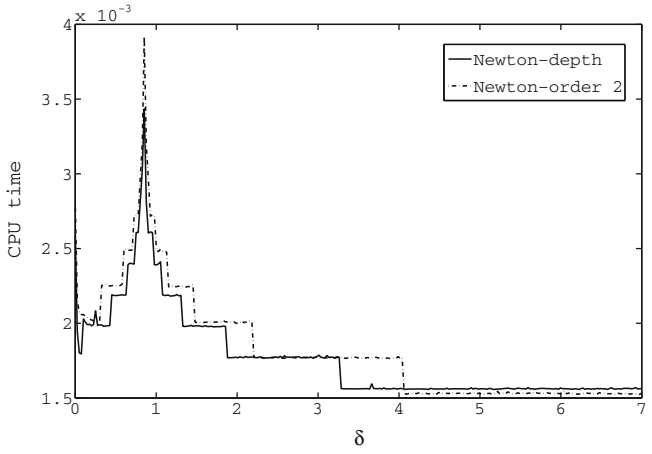

Fig. 4. CPU time to compute $\boldsymbol{q}$ with the Newton-depth and the Newton-order 2 algorithms.

4. The general Markovian tree and its relation with scalar tree-structured M/G/1-type Markov chains

When observed at the split and death occasions only, a general Markovian tree (GMT) can be characterized by a set of matrices B_j, for $j > 0$, where B_j has dimension $n \times n^j$, and a single $n \times 1$ vector $\boldsymbol{a}$. Entry $(B_j)_{i,h_1h_2\ldots h_j}$ gives the probability that a phase i branch splits into j branches, with the phase of the ith branch (from the left) equaling h_i. Notice that having $j = 1$ is also allowed (thus, a branch can *split* into a single other branch, possibly having the same phase). The ith entry of $\boldsymbol{a}$ gives the probability that a phase i branch dies without splitting.

A *scalar* tree-structured M/G/1-type Markov chain is a discrete-time Markov chain $(X_t)_{t\geqslant 0}$ taking values in a d-ary tree, meaning its state space Ξ equals

$$\Xi = \{J | J = j_1 \ldots j_n, j_k \in \{1,\ldots,d\}, k = 1,\ldots,n, n > 0\} \cup \emptyset.$$

Moreover, the state of the chain at time $t+1$ is determined as follows:

$$P[X_{t+1} = J' | X_t = J] = \begin{cases} a(\emptyset,\emptyset) & J' = J = \emptyset, \\ a(\emptyset,H) & J' = H, J = \emptyset, \\ a_1(k,H) & J \neq \emptyset, f(J,1) = k, J' = J - f(J,1) + H, \\ 0 & \text{otherwise}. \end{cases}$$

Thus, transitions can only occur between a node and any descendant of its parent.

We first indicate that there exists a one-to-one correspondence between all possible $\boldsymbol{a}$ and B_j matrices ($j > 0$) that characterize a GMT and the $a_1(k,H)$ values characterizing the behavior of a scalar tree-structured M/G/1-type Markov chain when it is away from the boundary $\emptyset$. As in the previous section, we will associate a transition in the Markov chain to each step in the (left) depth-first traversal of a given GMT realization. The correspondence between a GMT and a scalar tree-structured M/G/1-type Markov chain is easy to establish by setting $d = n$,

$$a_1(k,H) = (B_{|H|})_{k,h_{|H|}\ldots h_2h_1},$$

for $H = h_1h_2\ldots h_{|H|} \neq \emptyset$ and $a_1(k,\emptyset) = a_k$. Fig. 5 helps to visualize the correspondence for two types of transitions.

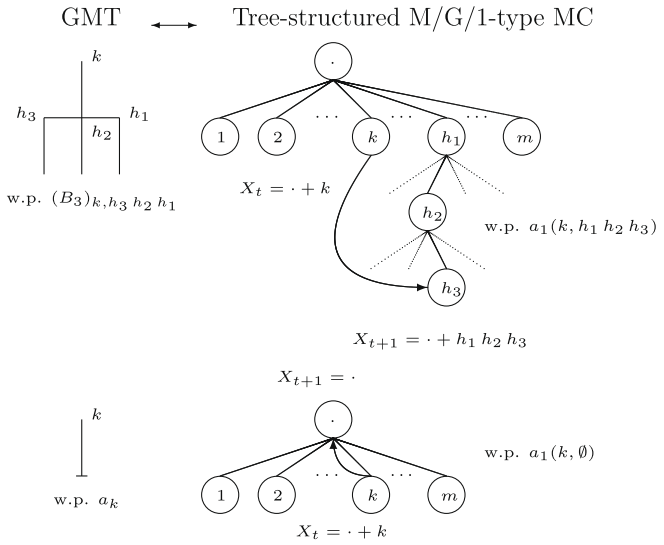

Fig. 5. The one-to-one correspondence between two possible transitions in a GMT (for $|H| = 3$ and for $|H| = 0$), and in a scalar tree-structured M/G/1-type Markov chain.

Analogue to the previous section, one sees that the evolution of the tree-structured Markov chain corresponds to a depth-first evolution of the multi-type branching process, where all the branches, except for the leftmost, are immediately frozen. In this case all the phases, including the one of the leftmost branch, are stored by adding them to the variable X_t. When a branch in phase i dies, which happens with probability $a_1(i,0) = a_i$, we unfreeze the last branch that we froze (where a set of branches is frozen from right to left).

The GMT realization is finite if and only if, starting from a node in the corresponding scalar tree-structured M/G/1-type Markov chain, we return to its parent in a finite time. However, this link between the multi-type branching processes and the scalar tree-structured M/G/1-type Markov chains does not provide us with a new efficient means to compute the extinction probability vector $\boldsymbol{q}$ of the branching process. But, it does allow us to show that the main result on the positive recurrence, null recurrence or transience of a scalar tree-structured M/G/1-type Markov chain as proven in He [8, Theorem 3.2], is equivalent to Athreya and Ney [1, Section V.3, Theorem 2], because the extinction probability q_i equals the probability that the Markov chain eventually returns to state $J = \emptyset$ given that it started in state $J = i$.

Indeed, the result in Athreya and Ney [1, Section V.3, Theorem 2] states that it suffices to compute the spectral radius $\mathrm{sp}(Q)$ of a nonnegative matrix Q in order to determine whether the branching process is subcritical $(\mathrm{sp}(Q) < 1)$, critical $(\mathrm{sp}(Q) = 1)$ or supercritical $(\mathrm{sp}(Q) > 1)$. The i,jth entry of Q represents the expected number of phase j branches born from a phase i branch. In our Markov chain notation we can write this as

$$Q_{i,j} = \sum_{J\in\Xi} a_1(i,J)N(J,j),$$

where $N(J,j)$ counts the number of occurrences of the integer j in the string J.

On the other hand, for a scalar tree-structured M/G/1-type Markov chain, He [8] defines the matrix $P = M^{-1}\Lambda$, whose spectral radius $\mathrm{sp}(P)$ determines whether the chain is positive recurrent $(\mathrm{sp}(P) < 1)$, null recurrent $(\mathrm{sp}(P) = 1)$ or transient $(\mathrm{sp}(P) > 1)$. Without going into detail and by noting that $a_0(k,J)$ was used to denote $a_1(k,k+J)$ (with $J \neq \emptyset$), and $a_2(k)$ to denote $a_1(k,k)$ in He [8], it can be seen that

$$\Lambda_{i,j} = \sum_{J\in\Xi} a_1(i,J)N(J,j) - \sum_{J\in\Xi} a_1(i,i+J),$$

and M is a diagonal matrix with its ith entry equal to $1 - \sum_{J\in\Xi} a_1(i,i+J)$. Hence, we have the relationship

$$Q = (I - M) + \Lambda.$$

The following result shows the equivalence between the main result on the criticality of a GMT and the positive recurrence, null recurrence or transience of a scalar tree-structured M/G/1-type Markov chain.

Theorem 4.1. *For the GMT defined above and the corresponding scalar tree-structured M/G/1-type Markov chain, one of the three relations between* $\mathrm{sp}(Q)$ *and* $\mathrm{sp}(P)$ *must hold*

(a) $1 = \mathrm{sp}(Q) = \mathrm{sp}(P)$,
(b) $1 < \mathrm{sp}(Q) < \mathrm{sp}(P)$,
(c) $0 < \mathrm{sp}(P) < \mathrm{sp}(Q) < 1$.

Proof. We first argue that

1. $\mathrm{sp}(Q) < 1 \iff \mathrm{sp}(P) < 1$,
2. $\mathrm{sp}(Q) = 1 \iff \mathrm{sp}(P) = 1$.

The first statement is immediate as $(I-P)^{-1}M^{-1} = (M-\Lambda)^{-1} = (I-Q)^{-1}$, meaning if $(I-P)^{-1}$ exists then so does $(I-Q)^{-1}$ and vice versa. The second follows from the Subinvariance Theorem in Seneta [16, Theorem 1.6] as the dominant eigenvector of Q is also an eigenvector of P. This establishes statement (a).

Now, analogue to Seneta [16, Theorem 2.5], we can prove that if $\mathrm{sp}(Q) \neq 1 \neq \mathrm{sp}(P)$, then the first statement can be refined to $1 < \mathrm{sp}(Q) < \mathrm{sp}(P)$ or $0 < \mathrm{sp}(P) < \mathrm{sp}(Q) < 1$. This completes the proof. □

Notice, as $\mathrm{sp}(P)$ lies further from 1 than $\mathrm{sp}(Q)$, it might be numerically preferable to compute $\mathrm{sp}(P)$ when the chain is (very) close to being null recurrent.

Acknowledgements

We thank the referees for their relevant comments, as well as Guy Latouche for some useful discussions, which helped to improve the presentation of the paper. The first author is an Aspirant of the Fonds National de la Recherche Scientifique (F.R.S.-F.N.R.S.), part of her research has been supported by a F.R.I.A. Grant.

References

[1] K.B. Athreya, P.E. Ney, Branching Processes, Springer-Verlag, New York, 1972.
[2] N. Bean, N. Kontoleon, P. Taylor, Markovian trees: Properties and algorithms, Annals of Operations Research 160 (2008) 31–50.
[3] D.A. Bini, G. Latouche, B. Meini, Solving nonlinear matrix equations arising in tree-like stochastic processes, Linear Algebra and its Applications 366 (2003) 39–64.
[4] P. Haccou, P. Jagers, V.A. Vatutin, Branching Processes: Variation, Growth, and Extinction of Populations, Cambridge University Press, 2005.
[5] S. Hautphenne, G. Latouche, M.-A. Remiche, Algorithmic approach to the extinction probability of branching processes, submitted for publication.
[6] S. Hautphenne, G. Latouche, M.-A. Remiche, Newton's iteration for the extinction probability of a Markovian binary tree, Linear Algebra and its Applications 428 (2008) 2791–2804.
[7] S. Hautphenne, K. Leibnitz, M.-A. Remiche, Extinction probability in peer-to-peer file diffusion, ACM SIGMETRICS Performance Evaluation Review 34 (2) (2006) 3–4.
[8] Q. He, Classification of Markov processes of M/G/1 type with a tree structure and its applications to queueing models, O.R. Letters 26 (2000) 67–80.
[9] Q. He, A.S. Alfa, The MMAP[K]/PH[K]/1 queues with a last-come-first-serve preemptive service discipline, Queueing Systems 28 (1998) 269–291.
[10] Q. He, A.S. Alfa, The discrete-time MMAP[K]/PH[K]/1/LCFS-GPR queue and its variants, in: Proceedings of the Third International Conference on Matrix Analytic Methods, Leuven (Belgium), 2000, pp. 167–190.
[11] M. Kimmel, D.E. Axelrod, Branching Processes in Biology, Springer-Verlag, New York, 2002.
[12] N. Kontoleon, The Markovian Binary Tree: A Model of the Macroevolutionary Process, Ph.D. Thesis, The University of Adelaide, 2005.

[13] G. Latouche, Newton's iteration for non-linear equations in Markov chains, Journal of Numerical Analysis 14 (1994) 585–598.
[14] J.M. Ortega, W.C. Rheinboldt, Iterative solution of nonlinear equations in several variables, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000 (Republication of work first published by Academic Press in 1970).
[15] G.T. Peeters, B. Van Houdt, C. Blondia, A multiaccess tree algorithm with free access, interference cancellation and single signal memory requirements, Performance Evaluation 64 (2007) 1041–1052.
[16] E. Seneta, Nonnegative Matrices and Markov Chains, Springer-Verlag, New York, 1981.
[17] T. Takine, B. Sengupta, R.W. Yeung, A generalization of the matrix M/G/1 paradigm for Markov chains with a tree structure, Stochastic Models 11 (3) (1995) 411–421.
[18] B. Van Houdt, 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) 247–270.
[19] B. Van Houdt, C. Blondia, Tree-structured QBD Markov chains and tree-like QBD processes, Stochastic Models 19 (4) (2003) 467–482.
[20] B. Van Houdt, C. Blondia, Throughput of Q-ary splitting algorithms for contention resolution in communication networks, Communications in Information and Systems 4 (2) (2005) 135–164.
[21] B. Van Houdt, C. Blondia, Analyzing priority queues with 3 classes using tree-like processes, Queueing Systems 54 (2006) 99–109.
[22] J. Van Velthoven, Analyzing Queues with Customer Differentiation using Matrix Analytic Methods, Ph.D. Thesis, The University of Antwerp, 2008.
[23] X. Yang, G. de Veciana, Service Capacity of Peer-to-Peer Networks, Technical Report, The University of Texas at Austin, 2004.
[24] R.W. Yeung, A.S. Alfa, The quasi-birth–death type Markov chain with a tree structure, Stochastic Models 15 (4) (1999) 639–659.
[25] R.W. Yeung, B. Sengupta, Matrix product-form solutions for Markov chains with a tree structure, Advances in Applied Probability 26 (1994) 965–987.