Growing Binary Trees in a Random Environment

Ilan Kessler and Moshe Sidi, *Senior Member, IEEE*

Abstract—**A class of binary trees that grow in a random environment, where the state of the environment can change at every vertex of the trees is studied. The trees considered are single-type and two-type binary trees that grow in a two-state Markovian environment. For each kind of tree, the conditions on the environment process for extinction of the tree are determined, and the problem of calculating the expected number of vertices of the tree is addressed. Different ways of growing the trees are compared.**

Index Terms— **Random trees, growing trees, random environment, splitting algorithms.**

I. Introduction

Consider a growing tree of which each vertex generates additional vertices according to some probabilistic reproduction law. Growing trees arise naturally in many applications, such as searching and sorting [8], multiaccess comunication [2], and growth of populations [3], [4]. Often, the tree that arises is growing in presence of a stochastic process, the *random environment,* which determines the reproduction law of each vertex. In addition, the tree may consist of vertices of different types, and the reproduction law of each vertex may depend on the type of the vertex.

We study a class of binary trees that grow in a random environment, which arise in multiaccess communication when the communication channel is noisy [6], [9], [12]. In this case, the growing tree describes a splitting algorithm and the random environment corresponds to the noise process. The importance of the trees considered lies in the fact that they determine the stability of the algorithms.

Most previous studies of randomly growing trees do not assume the existence of a random environment, and are based on the assumption that the vertices reproduce independently of each other. Growing trees in a random environment were considered so far only in the context of branching processes in a random environment [4], with the restriction that the state of the environment can change only at every *generation,* so that vertices that belong to the same generation (and are of the same type) have always the same reproduction law [1].

The binary trees considered here are growing in a random environment where the state of the environment can change at every *vertex.* Thus, the reproduction law is chosen separately for each vertex of the tree, and vertices that belong to the same generation need not have

Manuscript received April 10, 1990; revised September 11, 1991. This work supported by a grant from the Office of the Chief Scientist, Ministry of Communications, Israel. This work was presented at INFOCOM'91, Miami, FL, April 1991.

I. Kessler was with the Technion—Israel Institute of Technology, Haifa 32000, Israel. He is now with IBM Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, NY 10598.

M. Sidi is with the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel.

IEEE Log Number 9203043.

the same reproduction law (even if they are of the same type). In the special case where the environment process consists of independent and identically distributed random variables, the vertices reproduce independently of each other, and the growing tree corresponds to a multitype binary Galton–Watson process.

In the multiaccess communication system just discussed as well as in other applications, the performance of the system depends on the statistical behavior of the number of vertices in the trees that describe the system. In this correspondence, we are interested in the conditions on the environment process so that the trees do not grow indefinitely, and in the computation of the expected number of vertices of the trees. As will be evident shortly, the fact that the random environment changes at every vertex, implies that different ways of growing the trees may lead to different results. Thus, we are interested as well in comparing different ways of growing the trees.

II. Problem Formulation

Consider a tree that grows at the discrete times $t = 0, 1, 2, \cdots$. The tree consists of vertices of two types, 0 and 1. Initially, the tree consists of a single vertex of a given type. At each $t \geq 0$ one vertex of the existing tree is selected (according to a predetermined order among the vertices), and this vertex gives rise to new vertices (the *offsprings* of this vertex) according to some reproduction law. The reproduction law of a vertex depends on its type and is governed by a discrete-time homogeneous Markov chain $X = \{X_t, t \geq 0\}$ called the *environment.* The state space of the environment X consists of two states denoted by g and b. Suppose that the selected vertex at t is of type i $(i = 0, 1)$. If $X_t = g$ then with probability 1 the selected vertex does not reproduce at all. If $X_t = b$ then with probability ρ_i the selected vertex gives rise to two offsprings, and with probability $1 - \rho_i$ the vertex does not reproduce at all. A vertex that has been selected at any time $t \geq 0$ is never selected again. If at some instant $0 \leq T' < \infty$ there exists only one vertex that has not already been selected and this vertex does not reproduce at T', then the tree stops growing; if there is no such T' then we put $T' = \infty$. We refer to $T = T' + 1$ as the *length* of the tree. The event $\{T < \infty\}$ is called the *extinction* of the tree.

Since a vertex can be selected only once, it follows that each vertex has either no offsprings or two offsprings. A vertex that has offsprings is referred to as their *parent.* Two offsprings of the same parent are called *siblings;* one of them is referred to as the *left* offspring and the other as the *right* offspring. The growing tree is considered as a *rooted* tree; the initial vertex is the *root* of the tree, and each vertex is the root of a subtree that includes the offsprings of that vertex (if a vertex has not reproduced, then the subtree rooted at that vertex consists of only one vertex). Thus, the growing tree is a complete binary tree. A subtree rooted at a left (right) offspring is referred to as the left (right) subtree of the corresponding parent.

The type of each offspring is determined as follows. A parent of type 0 has only type 0 offsprings. A parent of type 1 has one offspring of type 1 and one offspring of type 0; the type 1 offspring is the right offspring with probability q and it is the left offspring with probability $1 - q$. Thus, if the root of the binary tree is of type 0, then the evolving tree is a *single-type* tree with only type 0 vertices. If the root is of type 1, then the evolving tree is a *two-type* tree with vertices of both types. In this case, every subtree rooted at a type 0 vertex is a single-type tree (with only type 0 vertices).

We now present two different orders for selecting the vertices. The first is referred to as depth first order (DFO) and is defined by the following rules. 1) A parent always precedes its offsprings. 2) The vertices of a right subtree always precede the sibling of the root of this subtree (which is a left offspring). The DFO is used in existing splitting algorithms [2]. The second order is referred to as breadth first order (BFO) and is defined by the following rules. 1) A right offspring always precedes its sibling (which is a left offspring). 2) An older vertex always precedes a younger one; that is, a vertex that was born at time $t' \geq 0$ precedes any vertex that was born at any time $t > t'$. Note that both the DFO and the BFO are independent of the type of the vertices. Examples of growing trees are shown in Fig. 1. In both examples, $\rho_0 = \rho_1 = 1$.

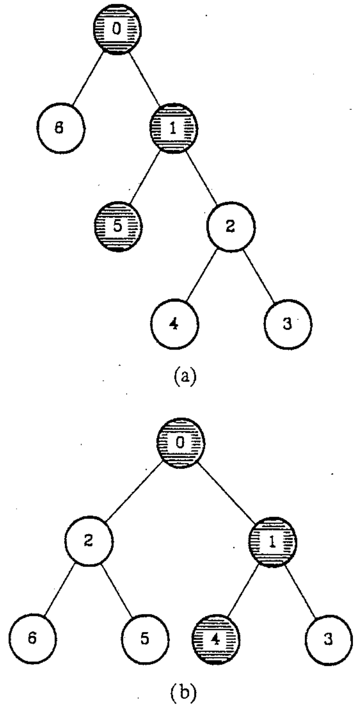

Fig. 1. Examples of binary trees growing in the environment $X(\omega) = \{bbbgggg\cdots\}$. Shaded vertex: Type 1. Unshaded vertex: Type 0. Number inside each vertex is the time at which the vertex is selected. (a) Under the DFO. (b) Under the BFO.

The following notation and assumptions are used in the sequel (similarly to [6]). The state space of the environment X is $\mathcal{X} = \{g, b\}$, and the transition matrix of X is

$$K = \begin{bmatrix} r_g & 1 - r_g \\ r_b & 1 - r_b \end{bmatrix} \tag{1}$$

(where the upper row and the left column correspond to state g). The elements of this matrix are denoted by $K(x, x')$, $x, x' \in \mathcal{X}$. We assume $r_g, r_b \in (0, 1)$, so X is irreducible. The invariant probabilities of X are $\nu(b) = r_b/(1 - r_g + r_b)$ and $\nu(g) = 1 - \nu(b)$. We assume $0 < q < 1$ and $\rho_i > 0$ $(i = 0, 1)$. For $x \in \mathcal{X}$ we denote $P_x(\cdot) = P(\cdot \mid X_0 = x)$ and $E_x(\cdot) = E(\cdot \mid X_0 = x)$. Due to space limitations, all proofs are omitted and the interested reader is referred to [7].

If $X_0 = g$ then the tree is degenerate with only one vertex i.e., $P_g(T = 1) = 1$. Thus, we are interested only in the case where $X_0 = b$.

III. The Single-Type Tree

In this section, we consider a single-type tree with only type 0 vertices. We first determine the condition on the environment X for almost sure extinction of the tree.

Theorem 1: Assume that the type of the root is 0. Then T is independent of the order by which the vertices are selected, and we have: $P_b\{T < \infty\} = 1$, if and only if $\rho_0\nu(b) \leq 1/2$.

For a single-type tree denote $w_x(x') = P_x(X_{T-1} = x')$ for all $x, x' \in \mathcal{X}$. Clearly, $w_g(g) = 1$. The value of $w_b(x), x \in \mathcal{X}$, is given in [6] for $\rho_0\nu(b) \le 1/2$. Since the length T of a single-type tree is independent of the order by which the vertices are selected, we can assume for the calculation of $E_b(T)$ an arbitrary order. In [6], the calculation of $E_b(T)$ is carried out for the DFO. As this result is used in the sequel, for the sake of completeness we state it here.

Theorem 2: Assume that the type of the root is 0. Then,

$$E_b(T) = \begin{cases} 1 + \frac{2\rho_0}{\eta[1/2-\rho_0\nu(b)]}, & \text{if } \rho_0\nu(b) < 1/2, \\ \infty, & \text{if } \rho_0\nu(b) \ge 1/2, \end{cases}$$

where $\eta = \{-2\rho_0(1-r_b)(1-r_g+r_b)\}/\{1+\beta/2-\rho_0[1-r_br_g+(1-r_b)^2] - \sqrt{\beta^2-4\alpha\gamma}/2\} > 0$ for $r_g \ne r_b$ and $\eta = 2$ for $r_g = r_b$, and $\alpha = \rho_0(1-r_b)(r_b-r_g), \beta = \rho_0[(1-r_b)(1-2r_b+r_g)+r_b(1-r_g)] - 1$, and $\gamma = \rho_0 r_b \cdot (1-r_b+r_g)$.

Consider now the more general case in which there are initially $1 \le n < \infty$ *separate* vertices of type 0. The vertices are selected and reproduce according to the rules defined in Section II, thus producing n growing binary trees referred to as a *forest*.

We now calculate $E_x(T)$ of a forest with n initial vertices. Since T is independent of the order by which the vertices are selected, we can assume for this calculation an arbitrary order. Thus, we number the n initial vertices from 0 to $n-1$, and we assume the following. For every $0 \le i \le n-1$, the tree rooted at the initial vertex i is grown under the DFO, and the vertices of this tree precede all the initial vertices with number $j > i$. Let T_i be the length of the tree rooted at the initial vertex i $(0 \le i \le n-1)$. Clearly, $E_x(T) = \Sigma_{i=0}^{n-1} E_x(T_i)$. Since $E_b(T) \ge E_b(T_0)$ and $E_g(T) \ge E_g(T_1) = E_g E_g(T_1 \mid X_1) = r_g + (1-r_g)E_b(T_0)$, it follows by Theorem 1 that if $p_0\nu(b) \ge 1/2$ then $E_x(T) = \infty$. Consider now the case $p_0\nu(b) < 1/2$. Let the expected length $E_x(T)$ of a forest with n roots be denoted by $L_0^x(n)$. Let $r_i (0 \le i \le n-1)$ be the time at which the initial vertex i is selected. Then, for any $0 \le i \le n-1$, we obtain $E_x(T_i) = \Sigma_{x' \in \mathcal{X}} P_x(X_{r_i} = x')E_{x'}(T_0)$. Therefore,

$$L_0^x(n) = L_0^x(1) + \sum_{i=1}^{n-1} \left[P_x(X_{r_i} = g) + P_x(X_{r_i} = b)L_0^b(1)\right], \quad (2)$$

where, by Theorem 2, $L_0^x(1) = 1$ if $x = g$ and $L_0^x(1) = 1 + 2\rho_0/[\eta(1/2-\rho_0\nu(b))]$ if $x = b$. For $P_x(X_{r_i} = x')$, $x, x' \in \mathcal{X}$, $1 \le i \le n-1$, we obtain the following recursive expressions:

$$P_x(X_{r_1} = x') = \sum_{x'' \in \mathcal{X}} w_x(x'')K(x'', x') \quad (3)$$

$$P_x(X_{r_1} = x') = \sum_{x'', x''' \in \mathcal{X}} P_x(X_{r_{i-1}} = x'')w_{x''}(x''')K(x''', x'), \quad 2 \le i \le n-1. \quad (4)$$

Equations (2)–(4) yield $L_0^x(n)$ for any $n < \infty$ and $x \in \mathcal{X}$. This is used in the sequel.

IV. The Two-Type Tree

In this section, we consider a two-type tree that consists of vertices of both type 0 and type 1. Note that in the special case $\rho_0 = \rho_1$, the two-type tree is in fact a single-type tree. We first determine the condition on the environment X for almost sure extinction of the tree.

Theorem 3: Assume that the type of the root is 1, and that the vertices are selected by an arbitrary order (independent of the type of the vertices). Then $P_b\{T < \infty\} = 1$, if and only if $\rho_0\nu(b) \le 1/2$.

The expected length $E_b(T)$ of the two-type tree in the case where the vertices are selected by the DFO is calculated in [6]. We now calculate $E_b(T)$ in the case where the vertices are selected by the BFO. It is not difficult to show that if $\rho_0\nu(b) \ge 1/2$ then $E_b(T) = \infty$ (see [7]). Thus we consider only the case $\rho_0\nu(b) < 1/2$. At any $t \ge 0$ there exists at most one vertex of type 1 that has not been selected at any $t' < t$. Let σ be the time at which such a vertex is selected and does not reproduce (possibly $\sigma = \infty$). Let N_t $(t \ge 0)$ be the number of type 0 vertices existing at time t that have not been selected at any $t' < t$. Let $S_t = \inf\{t' \ge t :$ the type 1 vertex is selected at $t'\}$. Define

$$D_t = \begin{cases} S_t - t + 1, & \text{if } t \le \sigma, \\ 0, & \text{if } t > \sigma, \end{cases}$$

(note that since $\rho_0\nu(b) < 1/2$ then $\sigma < \infty$ a.s.). Denote $L^x(n,d) = E(T \mid X_0 = x, N_0 = n, D_0 = d)$. With this notation, the expected length $E_b(T)$ of the two-type tree is $L^b(0,1)$. We use the following matrix notation:

$$G = \begin{bmatrix} r_g & 1-r_g \\ 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 \\ r_b & 1-r_b \end{bmatrix},$$
$$L(n,d) = \begin{bmatrix} L^g(n,d) \\ L^b(n,d) \end{bmatrix},$$
$$L_0(n) = \begin{bmatrix} L_0^g(n) \\ L_0^b(n) \end{bmatrix}, \qquad \mathbf{1} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

For brevity, in the sequel we write $L(n)$ instead of $L(n,1)$. Denote $G_i = G + (1-\rho_i)B$ and $B_i = \rho_i B, i = 0,1$. Let the set $\{1,2,\cdots,s\}$ be denoted by N_s. For $\Lambda \subseteq N_s$, denote by $\boldsymbol{H}_s(\Lambda)$ the product $H_1H_2\cdots H_s$ where $H_k = G_0$ for $k \in \Lambda$ and $H_k = B_0$ otherwise. Let i,j be nonnegative integers. Define $(G_0B_0)^{i,j} = \Sigma_{\Lambda \subseteq N_{i+j}:|\Lambda|=i} \boldsymbol{H}_{i+j}(\Lambda)$ if $i+j > 0$, and $(G_0B_0)^{0,0} = I$.

Theorem 4: The expected lengths $L(n), n \ge 0$, satisfy the following infinite system of equations:

$$Z(n) = B_1 \sum_{i=0}^{2(n+1)} A_i^{(n)} Z(i) + C^{(n)}, \qquad n \ge 0, \quad (5)$$

where $A_{2k}^{(n)} = (1-q)(G_0B_0)^{n+1-k,k}$ for $0 \le k \le n+1$, $A_{2k+1}^{(n)} = q(G_0B_0)^{n-k,k}$ for $0 \le k \le n$, and $C^{(n)} = G_1L_0(n) + \left[\frac{1}{1+\rho_1(n+1-q)}\right]$.

The following is useful for computations.

Corollary 1: For $n \ge 0$,

$$A_0^{(n)} = (1-q)G_0^{n+1}, \qquad A_1^{(n)} = qG_0^n,$$
$$A_{2n+1}^{(n)} = qB_0^n, \qquad A_{2(n+1)}^{(n)} = (1-q)B_0^{n+1}.$$

For $2 \le i \le 2n$, the following recursive relation holds: $A_i^{(n)} = G_0A_i^{(n-1)} + B_0A_{i-2}^{(n-1)}$.

From Theorem 4, it follows that $L^g(n) = 1 + r_gL_0^g(n) + (1-r_g)L_0^b(n)$ for all $n \ge 0$, which could be obtained directly as well.

To obtain $L(0)$ (and thus $L^b(0)$), we need to find the particular solution $\{L(n), n \ge 0\}$ of the infinite system (5). In general, an exact expression for this solution is difficult to obtain. In the special case where $\rho_0 = 0$, the system can be solved exactly and we discuss this case shortly. In the general case, we have a computable lower bound by the following.

Proposition 1: If system (5) has a finite positive solution $\{Z^*(n), n \ge 0\}$, then for all $N \ge 0$, the finite system

$$Z(n) = B_1 \sum_{i=0}^{N} A_i^{(n)} Z(i) + C^{(n)}, \qquad 0 \le n \le N, \quad (6)$$

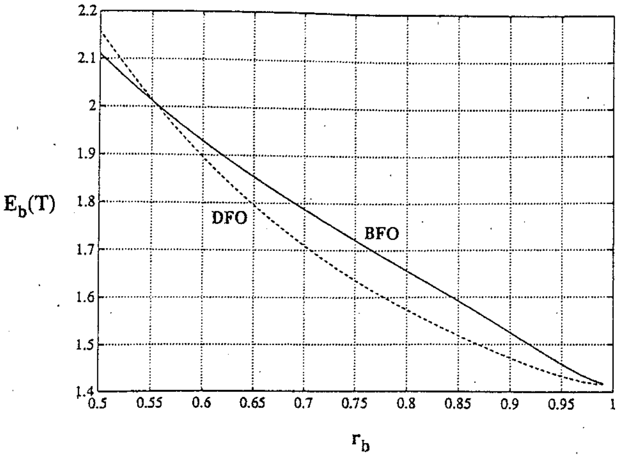

Fig. 2. Expected length versus r_b for $\nu(b) = 0.5$, $\rho_0 = 0.9$, $\rho_1 = 0.1$, and $q = 0.1$ (lower bound for the BFO and exact value for the DFO).

has a unique solution $\{Z_N(n), 0 \le n \le N\}$ such that $0 \le Z_N(n) \le Z^*(n)$ for all $0 \le n \le N$.

Since $L(n) > 0$ for all $n \ge 0$, it follows by Proposition 1 that by solving the finite system (6) for any $N \ge 0$, we obtain a lower bound on $L(n), 0 \le n \le N$, and in particular on $L^b(0)$. This enables us to show numerically that for certain values of the parameters, the expected length of the two-type tree under the BFO is strictly greater than the expected length under the DFO. An example that demonstrates this is given in Fig. 2, where the bound for the BFO is obtained by taking $N = 2$.

The Special Case $\rho_0 = 0$: The case $\rho_0 = 0$ corresponds in the multiaccess communication system discussed in Section I, to the likely situation where the noise can garble messages but cannot cause the receiver to misinterpret a silence as a simultaneous transmission of a number of transmitters. We first calculate $E_b(T)$ and then compare between the expected lengths under the BFO and the DFO.

Proposition 2: Assume that the type of the root is 1 and that $\rho_0 = 0$. Then, for any order by which the vertices are selected, we have $E_b(T) < \infty$.

When $\rho_0 = 0$, the first two equations of system (5) become

$$Z(0) = B_1[(1-q)KZ(0) + qZ(1)] + C^{(0)},$$
$$Z(1) = B_1[(1-q)K^2Z(0) + qKZ(1)] + C^{(1)}.$$

Solving these equations, we easily obtain $L^b(0)$ (it can be shown that these equations have a unique solution).

We now address the question which order, the DFO or the BFO, yields a shorter expected length of the two-type tree. The complete answer is given by the following.

Theorem 5: Assume that the type of the root is 1 and that $\rho_0 = 0$. Let $\theta = r_g - r_b$. Then, we have

$$\text{if } 0 < \theta < 1 \quad \text{or} \quad -1 < \theta < -\frac{q}{1-q},$$
$$\text{then } E_b^{\text{BFO}}(T) < E_b^{\text{DFO}}(T);$$
$$\text{if } -\frac{q}{1-q} < \theta < 0, \quad \text{then } E_b^{\text{BFO}}(T) > E_b^{\text{DFO}}(T);$$
$$\text{if } \theta = 0 \quad \text{or} \quad \theta = -\frac{q}{1-q}, \quad \text{then } E_b^{\text{BFO}}(T) = E_b^{\text{DFO}}(T).$$

V. Discussion

The trees considered in this correspondence arise in multiaccess communication, when splitting algorithms are used to access a noisy communication channel. The expected length of these trees is directly related to the performance of such communication system. This motivates the interest in comparing between the expected lengths under the BFO and the DFO. In the special case where vertices of type 0 do not reproduce ($\rho_0 = 0$), the complete answer to this question is given (Theorem 5). The importance of this result lies in the fact that it demonstrates the property that the superiority of the DFO over the BFO (or vice versa) depends on the type of the memory of the environment process. Consider, for instance, the case where $q \ge 1/2$. In this case, we obtain that the expected length is smaller under the BFO if $r_b < r_g$, is smaller under the DFO if $r_b > r_g$, and is the same under both orders if $r_b = r_g$. When $r_b < r_g$ the process X is said to have a *persistent* memory, and when $r_b > r_g$ the process is said to have an *oscillatory* memory (when $r_b = r_g$ the process is memoryless) [10]. Thus, the result obtained is that the BFO is superior (i.e., yields a shorter expected length) when the memory is persistent, and the DFO is superior when the memory is oscillatory.

An intuitive explanation to the above phenomenon is as follows. A process having an oscillatory memory would typically alternate frequently between the two states b and g, whereas a process having a persistent memory would typically stay for a long period in a state before alternating to the other state. Suppose now that the memory is persistent. As the state of the environment at the root of the tree is b, subsequent states would typically be also b. Under the DFO, these b states will typically occur at vertices of type 1, since $q \ge 1/2$ and therefore, it is more likely that the type 1 vertex would be the right offspring than the left offspring. On the other hand, under the BFO some of these b states will occur at vertices of type 0 (which are not affected by the state of the environment ($\rho_0 = 0$)), since vertices are selected by advancing also to the breadth of the tree. Therefore, the expected length under the BFO is smaller. Suppose now that the memory is oscillatory. In this case, the burst of b states is short, and under the DFO a subsequent g state will typically terminate the growth of the tree. On the other hand, under the BFO such g state may be wasted on a type 0 vertex, thus deferring the termination of the growth of the tree. Therefore, the expected length under the BFO is greater.

References

[1] D. Aldous and P. Shields, "A Diffusion limit for a class of randomly-growing binary trees," *Probab. Theory Related Fields*, vol. 79, pp. 509–542, 1988.
[2] D. Bertsekas and R. Gallager, *Data Networks.* Englewood Cliffs, NJ: Prentice-Hall, 1987.
[3] T.E. Harris, *The Theory of Branching Processes.* Berlin: Springer-Verlag, 1963.
[4] P. Jagers, *Branching Processes with Biological Applications.* London: Wiley, 1975.
[5] L.V. Kantorovich and V.I. Krylov, *Approximate Methods of Higher Analysis.* Groningen: P. Noordhoff, 1964.
[6] I. Kessler and M. Sidi, "Splitting algorithms in noisy channels with memory," *IEEE Trans. Inform. Theory*, vol. 35, pp. 1034–1043, Sept. 1989.
[7] ——, "Growing binary trees in a random environment," in *Proc. INFOCOM'91*, Bal Harbour, Fl, Apr. 1991, pp. 638–647.
[8] D.E. Knuth, *The Art of Computer Programming: Sorting and Searching (vol. 3).* Reading, MA: Addison-Wesley, 1973.
[9] J.L. Massey, "Collision resolution algorithms and random-access communications," in *Multi-User Communications Systems*, (CISM Courses and Lectures Series), G. Longo, Ed. New York: Springer-Verlag, 1981, pp. 73-137.
[10] M. Mushkin and I. Bar-David, "Capacity and coding for the Gilbert–Elliot channel," *IEEE Trans. Inform. Theory*, vol. 35, pp. 1277–1290, Nov. 1989.
[11] W.C. Torrez, "The birth and death chain in a random environment: Instability and extinction theorems," *Ann. Probab.*, vol. 6, pp. 1026–1043, 1978.
[12] N.D. Vvedenskaya and B.S. Tsybakov, "Random multiple access of packets to a channel with errors," *Probl. Inform. Transm.*, vol. 19, pp. 131–147, 1983.