Discrete Mathematics and Theoretical Computer Science DMTCS vol. **10:3**, 2008, 149–160

Waiting Time Distribution for Pattern Occurrence in a Constrained Sequence: an Embedding Markov Chain Approach

Grégory Nuel

MAP5, CNRS, UMR 8145, University Paris Descartes, 45 rue des Saint-Pères, 75270 Paris Cedex 06, France

received Dec 6, 2007, *revised May 13, 2008*, *accepted Nov 25, 2008*.

In this paper we consider the distribution of a pattern of interest in a binary random (d, k)-sequence generated by a Markov source. Such constrained sequences are frequently encountered in communication systems. Unlike the previous approach based on generating function we have chosen here to use Markov chain embedding techniques. By doing so, we get both previous results (sequence constrained up to the r^{th} occurrence), and new ones (sequence constrained up to its end). We also provide in both cases efficient algorithms using basic linear algebra only. We then compare our numerical results to previous ones and finally propose several interesting extensions of our method which further illustrate the usefulness of our approach. That is to say higher order Markov chains, renewal occurrences rather than overlapping ones, heterogeneous models, more complex patterns of interest, and multistate trial sequences.

Keywords: pattern, waiting time, conditional distribution, (d, k)-sequences

1 Introduction

The binary sequences used in communications systems (like magnetic or optical recording systems) are often subject to technical constraints. The simplest of these constraints allows runs of zeros of specific lengths. Several authors (Zehavi and Wolf, 1988; Marcus et al., 1998; Lothaire, 2005) have considered such particular constraint sequences called (d, k)-sequence containing no run of zeros of length smaller than d or greater than k. It is clear that this constraint is equivalent to the case when forbidden patterns are introduced. For instance, these forbidden patterns are '11' and '00000' in a $(1, 4)$-sequence, they are '11', '101' and '0000' in a $(2, 3)$-sequence. In this paper, we consider the distribution of a given pattern of interest (for example '100100100') in a random (d, k)-sequence generated by a Markov source.

Recently, this problem has been considered by Stefanov and Szpankowski (2007) where the authors use a sophisticated approach based on generating functions. In this paper, we introduce an alternative approach to the same problem using the well known technique of Markov chain embedding (Fu, 1996; Chadjiconstantinidis et al., 2000; Antzoulakos, 2001; Fu and Chang, 2002) which will allow us both to get the same results as in a previous paper, but also several interesting extensions. One should note that we use the same notation as that in Stefanov and Szpankowski (2007) to facilitate the comparison.

Formally, let $X = X_1X_2\dots$ be an homogeneous Markov chain over $\{0, 1\}$ with transition matrix π and starting distribution μ_1. We denote by N_i (resp. $\bar{N}_i^{(d,k)}$) the number of occurrences of the pattern of interest

$w = w_1w_2\dots w_m$ (resp. the forbidden patterns $\bar{w} \in \bar{\mathcal{W}}$) in $X_1X_2\dots X_i$ and by Y_r the waiting time of the r^{th} occurrence of w. Then, we consider the following two probabilities of interest:

$$\mathbb{P}(Y_r \leqslant n|\bar{N}_n^{(d,k)} = 0) \quad = \quad \mathbb{P}(N_n \geqslant r|\bar{N}_n^{(d,k)} = 0); \tag{1}$$

the distribution of Y_r given the sequence is constrained up to n, and

$$\begin{aligned} \mathbb{P}(Y_r \leqslant n|\bar{N}_{Y_r}^{(d,k)} = 0) \quad &= \quad \frac{\sum_{i=1}^{n} \mathbb{P}(Y_r = i, \bar{N}_i^{(d,k)} = 0)}{\sum_{i=1}^{\infty} \mathbb{P}(Y_r = i, \bar{N}_i^{(d,k)} = 0)} \\ &= \quad \mathbb{P}(N_n \geqslant r|\bar{N}_{Y_r}^{(d,k)} = 0) \end{aligned} \tag{2}$$

the distribution of Y_r given the sequence is constrained up to Y_r.

This paper is organized as follows: Section 2 presents our main results starting with the embedding Markov chain allowing to keep track of both occurrences of the pattern of interest and of the forbidden patterns (2.1) followed by its applications to probability computations given the constrained sequence up to Y_r (2.2) like it is done in Stefanov and Szpankowski (2007) or up to n (2.3) which is a new result. In both cases, we derive efficient algorithms to compute these probabilities by using basic linear algebra only. Then we give a numerical example, we compare our results to the ones from previous paper (2.4), and we study the practical limits of our algorithm through intense testing (2.5). In Section 3, we propose several extensions of the method: higher order Markov chains (3.1), renewal occurrences rather than overlapping ones (3.2), heterogeneous models (3.3), more complex patterns of interest (3.4) and multistate trial sequences (3.5).

2 Main results

2.1 Embedding Markov Chain

In order to avoid degenerated cases, let us assume that w does not contain any of the forbidden patterns and that neither w nor any $\bar{w} \in \bar{\mathcal{W}}$ belongs to $\{0,1\}$. Let us define $\mathcal{P}$ the union of $\{0,1\}$ and of the set of all (non empty) prefixes of both w and the forbidden patterns. For example, studying $w = 100100100$ in $(1,4)$-sequences, we get:

$$\mathcal{P} = \{0, 1, 11, 00, 000, 0000, 00000, 10, 100, 1001, 10010, 100100, 1001001, 10010010, 100100100\}.$$

Let us consider the transition function $\delta : \mathcal{P} \times \{0,1\} \to \mathcal{P}$ defined for all $p \in \mathcal{P}$ and $a \in \{0,1\}$ by $\delta(p,a)$ is the longest suffix of pa in $\mathcal{P}$. In our example we get: $\delta(00,0) = 000$, $\delta(00,1) = 1$, $\delta(1001,0) = 10010$, $\delta(1001,1) = 11$, and so on.

Theorem 1 *Let $\widetilde{X} = \widetilde{X}_1\widetilde{X}_2\dots$ a random sequence over $\mathcal{P}$ be defined by:*

$$\widetilde{X}_1 = X_1 \quad \text{and} \quad \widetilde{X}_i = \delta(\widetilde{X}_{i-1}, X_i) \quad \text{for all } i \geqslant 2.$$

Then $\widetilde{X}$ is an homogeneous Markov chain whose transition matrix Π is given for all (possibly empty) words p and q and for any $a, b \in \{0,1\}$ such as $pa, qb \in \mathcal{P}$ by:

$$\Pi(pa, qb) = \begin{cases} \pi(a,b) & \text{if } qb = \delta(pa, b) \\ 0 & \text{else} \end{cases}$$

and we obtain the following properties:

i) w *ends in position* i *in* $X \iff \widetilde{X}_i = w$*;*

ii) for all $\bar{w} \in \bar{\mathcal{W}}$*,* $\bar{w}$ *ends in position* i *in* $X \iff \widetilde{X}_i = \bar{w}$*.*

Proof: $\widetilde{X}$ is obviously a Markov chain and the expression of its transition matrix is straightforward to establish. We then remark that the Deterministic Finite Automaton (DFA) defined on the state space $\mathcal{P} \cup \{\varepsilon\}$ (ε denotes the empty word), the finite alphabet $\{0, 1\}$, with ε as starting state, $\{w\} \cup \bar{\mathcal{W}}$ as set of final states and $\widetilde{\delta}$ defined for all $p \in \mathcal{P} \cup \{\varepsilon\}$ and $a \in \{0, 1\}$ by

$$\widetilde{\delta}(p, a) = \begin{cases} a & \text{if } p = \varepsilon \\ \delta(p, a) & \text{else} \end{cases}$$

as transition function is exactly the Aho-Corasick DFA (Aho and Corasick, 1975) that allows us to count simultaneously w and all $\bar{w} \in \bar{\mathcal{W}}$. The properties i) and ii) directly come from those of this DFA. □

It should be noted that we have chosen here to explicitly use the Aho-Corasick DFA in the construction of our Markov chain embedding while most authors working on the subject usually use it implicitely (for instance, see Chang, 2005).

Corollary 2 *For all* $r \geqslant 0$ *the sequence* $(\widetilde{Z}_i^r)_{i \geqslant 1}$ *defined for all* $i \geqslant 1$ *by:*

$$\widetilde{Z}_i^r = \begin{cases} (\widetilde{X}_i, N_i, 0) & \textit{if } \bar{N}_i^{(d,k)} = 0 \\ (\widetilde{X}_i, r+, 0) & \textit{if } N_i \geqslant r \textit{ and } \bar{N}_i^{(d,k)} = 0 \\ (\widetilde{X}_i, \cdot, 1+) & \textit{if } \bar{N}_i^{(d,k)} \geqslant 1 \end{cases}$$

is a Markovian sequence whose starting distribution a_r *and transition matrix* A_r *are defined for all* $0 \leqslant i, j \leqslant r + 1$ *by size* $|\mathcal{P}|$ *blocks with:*

$$a_r(i) = \begin{cases} \mu_1 & \textit{if } i = 0 \\ 0 & \textit{else} \end{cases} \quad \textit{and} \quad A_r(i, j) = \begin{cases} P & \textit{if } 0 \leqslant i < r \textit{ and } j = i \\ Q & \textit{if } 0 \leqslant i < r \textit{ and } j = i + 1 \\ P + Q & \textit{if } i = j = r \\ \bar{Q} & \textit{if } 0 \leqslant i \leqslant r \textit{ and } j = r + 1 \\ \Pi & \textit{if } i = j = r + 1 \\ 0 & \textit{else} \end{cases}$$

where the blocks are ordered as follows: block 0 $(\mathcal{P}, 0, 0)$*, block 1* $(\mathcal{P}, 1, 0)$*, ..., block* $r - 1$ $(\mathcal{P}, r - 1, 0)$*, block* r $(\mathcal{P}, r+, 0)$ *and block* $r + 1$ $(\mathcal{P}, \cdot, 1+)$*; and where we decompose the transition matrix* Π *into:*

$$\Pi = P + Q + \bar{Q}$$

where Q *(resp.* $\bar{Q}$*) contains all transitions ending in* $\{w\}$ *(resp.* $\bar{\mathcal{W}}$*) and* P *the remaining ones.*

Proof: For all $0 \leqslant i < r$, a transition from the block $(\mathcal{P}, i, 0)$ to the same block does not allow any occurrence of w nor any $\bar{w} \in \bar{\mathcal{W}}$ to appear which means that the transition P must be used. If the transition is now from $(\mathcal{P}, i, 0)$ to $(\mathcal{P}, i + 1, 0)$, hence one occurrence of w just occurred, and we have to use the transition matrix $\bar{Q}$. If a forbidden pattern occurs, then the transition from block $(\mathcal{P}, i, 0)$ to $(\mathcal{P}, \cdot, 1+)$ obviously uses $\bar{Q}$. The expression of the remaining transitions simply relies on the same mechanism. □

Require: Two arrays of dimension $r \times |\mathcal{P}|$: $\mathtt{u}$ and $\mathtt{v}$; two real numbers: $\mathtt{sum1}$ and $\mathtt{sum2}$

1: INITIALIZATION:
2: $\mathtt{sum1} = \mathtt{sum2} = 0$
3: $\mathtt{u}[0] = \mu_1$ and $\mathtt{u}[j] = 0$ for all $1 \leqslant j \leqslant r-1$
4: $\mathtt{v}[j] = 0$ for all $0 \leqslant \mathtt{j} < r-1$ and $\mathtt{v}[r-1] = Qe_w^T$
5: FINITE SUM:
6: **for** $i = 2 \dots n$ **do**
7: $\quad \mathtt{sum1} += \mathtt{u} \cdot \mathtt{v}$
8: $\quad$ **for** $j = r-1 \dots 1$ **do** $\mathtt{u}[j] = \mathtt{u}[j]P + \mathtt{u}[j-1]Q$ **end for** and $\mathtt{u}[0] = \mathtt{u}[0]P$
9: **end for**
10: INFINITE SUM:
11: **while** $\mathtt{sum2}$ has not (numerically) converged **do**
12: $\quad \mathtt{sum2} += \mathtt{u} \cdot \mathtt{v}$
13: $\quad$ **for** $j = r-1 \dots 1$ **do** $\mathtt{u}[j] = \mathtt{u}[j]P + \mathtt{u}[j-1]Q$ **end for** and $\mathtt{u}[0] = \mathtt{u}[0]P$
14: **end while**

Output: $\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0) = \mathtt{sum1}/(\mathtt{sum1} + \mathtt{sum2})$

Algorithm 1: Algorithm computing $\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0)$ for any $r \geqslant 1$ and $n \geqslant 2$. All vector $\times$ matrix products (lines 8 and 13) use the sparse structure of matrices P and Q; the expression $\mathtt{u} \cdot \mathtt{v}$ (lines 7 and 12) denotes the scalar product of the two vectors. One can efficiently compute $\mathbb{P}(Y_r > n | \bar{N}_{Y_r}^{(d,k)} = 0)$ with the same algorithm by simply returning $\mathtt{sum2}/(\mathtt{sum1} + \mathtt{sum2})$. Space complexity is $O(r \times |\mathcal{P}|)$ and time complexity is $O(r \times |\mathcal{P}| \times n)$.

2.2 *The sequence is constrained up to* Y_r

Proposition 3 *For all $r \geqslant 1$ and for all $i \geqslant 2$ we have:*

$$\mathbb{P}(Y_r = i, \bar{N}_i^{(d,k)} = 0) = u_r R_r^{i-2} S_r v_r^T \tag{3}$$

where ^{T} denote the transpose operator and the two row-vectors u_r and v_r are defined for size $|\mathcal{P}|$ block $0 \leqslant i \leqslant r-1$ by:

$$u_r(i) = \begin{cases} \mu_1 & \text{if } i = 0 \\ 0 & \text{else} \end{cases} \quad \text{and} \quad v_r(i) = \begin{cases} e_w & \text{if } i = r-1 \\ 0 & \text{else} \end{cases},$$

(For all $p \in \mathcal{P}$, e_p denotes the indicatrix row-vector of state p) and two matrices R_r and S_r and are defined for the block (i,j) with $0 \leqslant i,j \leqslant r-1$ by:

$$R_r(i,j) = \begin{cases} P & \text{if } i = j \\ Q & \text{if } j = i+1 \\ 0 & \text{else} \end{cases} \quad \text{and} \quad S_r(i,j) = \begin{cases} Q & \text{if } i = j = r-1 \\ 0 & \text{else} \end{cases}.$$

Proof: If $Y_r = i$ and $\bar{N}_i^{(d,k)} = 0$ that means that $\widetilde{Z}_{i-1}^r$ belongs to the block $(\mathcal{P}, r-1, 0)$ and that w appears in position i which means that the transition Q is used. R_r is hence the submatrix of A_r corresponding to the blocks $(\mathcal{P}, j, 0)$ with $0 \leqslant j \leqslant r-1$, and S_r is a same dimension matrix allowing the product of the most right-handed block by Q. $\square$

Require: two arrays of dimension $r \times |\mathcal{P}|$: u and v; two arrays of dimension $|\mathcal{P}|$: x and y; three real numbers: sum1, sum2 and den

1: INITIALIZATION:
2: $\texttt{sum1} = 0$
3: $\texttt{u}[0] = \mu_1$ and $\texttt{u}[j] = 0$ for all $1 \leqslant j \leqslant r-1$
4: $\texttt{v}[j] = 0$ for all $0 \leqslant \texttt{j} < r-1$ and $\texttt{v}[r-1] = Qe_w^T$
5: FINITE SUM:
6: **for** $i = 2 \ldots n$ **do**
7: $\quad \texttt{sum1} += \texttt{u} \cdot \texttt{v}$
8: $\quad$ **for** $j = r-1 \ldots 1$ **do** $\texttt{u}[j] = \texttt{u}[j]P + \texttt{u}[j-1]Q$ **end for** and $\texttt{u}[0] = \texttt{u}[0]P$
9: **end for**
10: INFINITE SUM:
11: **for** $j = r-2 \ldots 0$ **do**
12: $\quad \texttt{u}[r-1] = \texttt{u}[r-1] + \texttt{lupower}(\texttt{u}[j], r-1-j)$
13: **end for**
14: $\texttt{sum2} = \texttt{lupower}(\texttt{u}[r-1], 1) \times e_w^T$
15: $\texttt{den} = \texttt{lupower}(\mu_1, r) \times e_w^T$

Output: $\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0) = \texttt{sum1}/\texttt{den}$

Algorithm 2: Algorithm computing $\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0)$ for any $r \geqslant 1$ and $n \geqslant 2$. All vector $\times$ matrix products (lines 8 and 13) use the sparse structure of matrices P and Q; the expression $\texttt{u} \cdot \texttt{v}$ (lines 7 and 12) denotes the scalar product of the two vectors. One can efficiently compute $\mathbb{P}(Y_r > n | \bar{N}_{Y_r}^{(d,k)} = 0)$ with the same algorithm by simply returning sum2/den. Lines 11-14 may be omitted if one is not interested in the computation of sum2. Space complexity is $O(r \times |\mathcal{P}|)$ and time complexity is $O(r \times |\mathcal{P}| \times n + r^2 \times |\mathcal{P}|)$ (remove the $r^2 \times |\mathcal{P}|$ term if lines 11-14 are omitted).

Corollary 4 *For all $r \geqslant 1$ and $n \geqslant 2$ we have:*

$$\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0) = \frac{\sum_{i=2}^{n} u_r R_r^{i-2} S_r v_r^T}{\sum_{i=2}^{\infty} u_r R_r^{i-2} S_r v_r^T}. \tag{4}$$

Proof: This simply results from a combination of Equation (2) and Equation (3). □

Algorithm 1 is a straightforward implementation of (4) using the special structure of R_r to get an efficient iterative computation of vector $\times$ matrix products. The tail sum is here computed numerically which can be a source of error due to the finite level of precision. An alternative could consist in using the inverse of $I - R_r$ to immediately get this tail sum as suggested by the following corollary.

Corollary 5 *If $I - P$ is invertible then for all $r \geqslant 1$ and $n \geqslant 2$ we have:*

$$\mathbb{P}(Y_r \leqslant n | \bar{N}_{Y_r}^{(d,k)} = 0) = \frac{\sum_{i=2}^{n} u_r R_r^{i-2} S_r v_r^T}{u_r T_r S_r v_r^T} \quad \text{and} \quad \mathbb{P}(Y_r > n | \bar{N}_{Y_r}^{(d,k)} = 0) = \frac{u_r R_r^{n-1} T_r S_r v_r^T}{u_r T_r S_r v_r^T} \tag{5}$$

where $T_r = (I - R_r)^{-1}$ is defined on the block (i,j) with $0 \leqslant i,j \leqslant r-1$ by:

$$T_r(i,j) = \begin{cases} [(I-P)^{-1}Q]^{j-i}(I-P)^{-1} & \text{if } j \geqslant i \\ 0 & \text{else} \end{cases}.$$

Require: Two array of size $|\mathcal{P}|$: `x` and `b`
$\texttt{b} = u$
for $i = 1 \dots k$ **do**
  find `x` such as $\texttt{x}(I - P) = \texttt{b}$ *// sparse LU solving*
  $\texttt{b} = \texttt{x}Q$ *// sparse product*
end for
Output: $u[(I - P)^{-1}Q]^k = \texttt{b}$

Algorithm 3: Procedure $\texttt{lupower}(u, k)$ computing $u[(I - P)^{-1}Q]^k$ through sequential LU solving of linear equations. The necessary sparse LU factorization has to be performed only once for all `lupower` calls. Complexities are $O(|\mathcal{P}|)$ in space and $O(|\mathcal{P}| \times k)$ in time.

Proof: Thanks to the definition of R_r it is clear that $[(I-R_r)\times T_r](i,j)(I-P)\times T_r(i,j)-Q\times T_r(i+1,j)$. It is then easy to verify that such a block is either I if $i = j$ or 0 otherwise thus proving that $(I - R_r) \times T_r = I$. A similar approach shows that $T_r \times (I-R_r) = I$. We then just have to replace $\sum_{i=0}^{\infty} R_r^i$ by $T_r = (I-R_r)^{-1}$ in equation (4). □

This result allows us to get Algorithm 2 which is more robust numerically than the previous one, and also faster. However, and unlike Algorithm 1, this algorithm cannot be extended to a heterogeneous model.

2.3 The sequence is constrained up to n

Proposition 6 *For all $r \geqslant 0$ and $n \geqslant 1$ we have:*

$$\mathbb{P}(Y_r \leqslant n, \bar{N}_n^{(d,k)} = 0) = y_r T_r^{n-1} z_r^T \tag{6}$$

where the two vectors y_r and z_r are defined for the size $|\mathcal{P}|$ block $0 \leqslant i \leqslant r$ by:

$$y_r(i) = \begin{cases} \mu_1 & \textit{if } i = 0 \\ 0 & \textit{else} \end{cases} \quad \textit{and} \quad z_r(i) = \begin{cases} 1 & \textit{if } i = r \\ 0 & \textit{else} \end{cases},$$

and the matrix T_r is defined for the block (i, j) with $1 \leqslant i, j \leqslant r$ by:

$$T_r(i,j) = \begin{cases} P & \textit{if } 0 \leqslant i = j < r \\ Q & \textit{if } j = i + 1 \\ P + Q & \textit{if } i = j = r \\ 0 & \textit{else} \end{cases}.$$

Proof: As $\mathbb{P}(Y_r \leqslant n, \bar{N}_n^{(d,k)} = 0) = \mathbb{P}(N_n \geqslant r, \bar{N}_n^{(d,k)} = 0)$ we need to compute the probability that $\widetilde{Z}_n^r$ belongs to blocks $(\mathcal{P}, r+, 0)$. The transition matrix T_r is hence simply a submatrix of A_r. □

Corollary 7 *For all $r \geqslant 1$ and $n \geqslant 1$ we have:*

$$\mathbb{P}(Y_r \leqslant n | \bar{N}_n^{(d,k)} = 0) = \frac{y_r T_r^{n-1} z_r^T}{y_r T_r^{n-1} 1^T} \tag{7}$$

where 1 is a row-vector of ones (of the appropriate dimension).

Proof: This simply results from a combination of Equation (1) and Equation (6). □

A straightforward implementation of (7) gives Algorithm 4. Like in Algorithm 1, we take advantage of the sparse structure of matrices P and Q through a very natural recurrence expression of vector $\times$ matrix products with T_r.

Require: One array of dimension $(r+1) \times |\mathcal{P}|$: `y`; two real numbers: `sum1` and `sum2`

1: INITIALIZATION:
2: $\texttt{sum1} = 0$ and $\texttt{sum2} = 0$
3: $\texttt{y}[0] = \mu_1$ and $\texttt{y}[j] = 0$ for all $1 \leqslant j \leqslant r$
4: MAIN LOOP:
5: **for** $i = 2 \dots n$ **do**
6: $\quad \texttt{y}[r] = \texttt{y}[r](P+Q) + \texttt{y}[r-1]Q$
7: $\quad$ **for** $j = r-1 \dots 1$ **do** $\texttt{y}[j] = \texttt{y}[j]P + \texttt{y}[j-1]Q$ **end for** and $\texttt{y}[0] = \texttt{y}[0]P$
8: **end for**
9: $\texttt{sum1}+= \texttt{y}[r] \cdot \mathbf{1}$ and **for** $j = 0 \dots r-1$ **do** $\texttt{sum2}+= \texttt{y}[j] \cdot \mathbf{1}$ **end for**

Output: $\mathbb{P}(Y_r \leqslant n | \bar{N}_n^{(d,k)} = 0) = \texttt{sum1}/(\texttt{sum1} + \texttt{sum2})$

Algorithm 4: Algorithm computing $\mathbb{P}(Y_r \leqslant n | \bar{N}_n^{(d,k)} = 0)$ for any $r \geqslant 1$ and $n \geqslant 1$. All vector $\times$ matrix products (lines 7 and 8) use the sparse structure of matrices P and Q. One can efficiently compute $\mathbb{P}(Y_r > n | \bar{N}_n^{(d,k)} = 0)$ with the same algorithm by simply returning $\texttt{sum2}/(\texttt{sum1} + \texttt{sum2})$. Space complexity is $O(r \times |\mathcal{P}|)$ and time complexity is $O(r \times |\mathcal{P}| \times n)$.

2.4 Numerical application

We consider the occurrence of $w = 100100100$ in $(1,4)$-sequences where the unconstrained sequence $X = X_1X_2\dots$ is a Markov chain whose transition matrix π may be defined by $\pi_{0,0} = 0.4$, $\pi_{0,1} = 0.6$, $\pi_{1,0} = 1.0$ and $\pi_{1,1} = 0.0$. The set of forbidden patterns is hence $\bar{\mathcal{W}} = \{11, 00000\}$, but can be reduced to $\bar{\mathcal{W}} = \{00000\}$ since 11 cannot appear in the sequence (transition $\pi_{1,1}$ is null).

According to Theorem 1, we consider the Markov chain $\widetilde{X} = \widetilde{X}_1\widetilde{X}_2\dots$ whose states are in

$$\mathcal{P} = \{0, 1, 00, 10, 000, 100, 0000, 1001, 00000, 10010, 100100, 1001001, 10010010, 100100100\},$$

and relies on the following transition matrix

$$\Pi = \begin{pmatrix}
0. & 0.6 & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0.6 & 0. & 0. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. \\
0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0.6 & 0. & 0. \\
0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. \\
0. & 0.6 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.4 \\
0. & 0. & 0. & 0. & 0.4 & 0. & 0. & 0. & 0. & 0. & 0. & 0.6 & 0. & 0.
\end{pmatrix}$$

If we number the states of $\mathcal{P}$ from 1 to 14 we get that 1 (resp. 2) is the starting state corresponding to $X_1 = 0$ (resp. $X_1 = 1$), 9 correspond to the forbidden pattern '00000' and 14 to $w = 100100100$ our pattern of interest.

r	$\mathbb{P}(N_n \geqslant r \| \bar{N}^{(1,4)}_{Y_r} = 0)$	running time	$\mathbb{P}(N_n \geqslant r \| \bar{N}^{(1,4)}_{n} = 0)$	running time
1	0.9998229893	$0.023s$	0.9744091233	$0.024s$
2	0.9988429172	$0.039s$	0.8990552029	$0.033s$
3	0.9957604487	$0.055s$	0.7729881352	$0.042s$
4	0.9885806721	$0.070s$	0.6167698770	$0.052s$
5	0.9748567886	$0.086s$	0.4578925276	$0.061s$
6	0.9521461776	$0.102s$	0.3179937858	$0.070s$
7	0.9185406129	$0.117s$	0.2078334759	$0.079s$
8	0.8731122847	$0.133s$	0.1285845121	$0.089s$
9	0.8161570287	$0.150s$	0.0757066715	$0.099s$
10	0.7491883799	$0.166s$	0.0426162883	$0.109s$

Tab. 1: Results of Algorithm 2 and Algorithm 4 for several values of r with $n = 500$, $w = 100100100$, $\pi_{0,0} = 0.4$, $\pi_{0,1} = 0.6$, $\pi_{1,0} = 1.0$, $\pi_{1,1} = 0.0$ and assuming that the Markov chain starts with $X_1 = 0$. The computational time using a (slow) Scilab implementation and running on a Xeon 1.86 Gz processor are also given.

The results we get in Table 1 for $\mathbb{P}(Y_r \leqslant n|\bar{N}^{(d,k)}_{Y_r} = 0, X_1 = 0)$ are exactly the same as the ones from Stefanov and Szpankowski (2007). This is something rather positive since the two approaches use completely different techniques (Markov chain embedding here and generating functions in the previous paper). It is also interesting to note that even with our slow Scilab implementation (from our personal experience one can expect at least to decrease by a factor 20 the computational time with a pure C implementation of the same algorithms), the computation time remains relatively small. Since working conditionally to the constraint up to Y_r involves an infinite sum while the constraint up to n only requires a finite one, the computational times for the first probability are longer than the second ones, but grows linearly with r in both cases.

2.5 Numerical limits of the presented algorithms

The space and time complexities of all algorithms we have presented in this paper are strongly connected to three parameters: the number r of occurrences for the pattern of interest, the sequence length n, and the size $|\mathcal{P}|$ depending both on the pattern of interest and on the set of forbidden patterns. These complexities are linear in all theses parameters, except for Algorithm 2 where there also is a small quadratic contribution of r when the tail distribution is requested.

In Table 2 we study the practical limits of our algorithms by considering patterns whose size ranges from 9 to 180, and various values of r and n. In this table, we can see that the numerical convergence used in Algorithm 1 provides reliable results (with only two degenerated cases where Algorithm 1 fails to give the correct answer). However, Algorithm 2 proves itself to be faster in all but one case ($j = 2, r = 15, n = 500$).

3 Extensions of the method

3.1 Markov chain of higher order

It is easy to adapt our method to the case where X is an order m Markov chain. To do so, we just have to replace $\mathcal{P}$ by the union of $\{0,1\}^m$ with all the prefixes of w and all $\bar{w} \in \bar{\mathcal{W}}$ whose length is at least m. The algorithms then remain the same except that the starting distribution μ_1 over $\{0,1\}$ must be replaced by a starting distribution μ_m over $\{0,1\}^m$ and hence, the first transition to consider is to position $i = m+1$ rather than $i = 1 + 1 = 2$ (line 6 in Algorithm 1 and Algorithm 4).

j	$\vert\mathcal{P}\vert$	r	n	n_∞	$\mathbb{P}(N_n \geqslant r \vert \bar{N}_{Y_r}^{(1,4)} = 0)$	$\mathbb{P}(N_n < r \vert \bar{N}_{Y_r}^{(1,4)} = 0)$	running time
1	15	15	500	∞	$3.6458298048 \times 10^{-01}$	$6.3541701952 \times 10^{-01}$	$0.251s$
				3494	$3.6458298048 \times 10^{-01}$	$6.3541701952 \times 10^{-01}$	$0.783s$
		150	500	∞	$1.1642613103 \times 10^{-48}$	1.0	$2.591s$
				11774	$1.1642613103 \times 10^{-48}$	1.0	$57.036s$
		10	5000	∞	1.0	$2.3644758572 \times 10^{-28}$	$1.669s$
				7056	1.0	$2.3644758572 \times 10^{-28}$	$2.397s$
		100	5000	∞	$9.8617817984 \times 10^{-01}$	$1.3821820160 \times 10^{-02}$	$16.144s$
				9512	$9.8617817984 \times 10^{-01}$	$1.3821820160 \times 10^{-02}$	$30.623s$
2	24	15	500	∞	$9.6256991047 \times 10^{-01}$	$3.7430089526 \times 10^{-02}$	$0.262s$
				4095	$9.6256991047 \times 10^{-01}$	$3.7430089526 \times 10^{-02}$	$0.172s$
		150	500	∞	$7.2947817992 \times 10^{-03}$	$9.9959925625 \times 10^{-01}$	$2.725s$
				6249	$7.2947817992 \times 10^{-03}$	$9.9959925625 \times 10^{-01}$	$30.530s$
		10	5000	∞	1.0	$1.3627569774 \times 10^{-20}$	$1.827s$
				8309	1.0	$1.3627569774 \times 10^{-20}$	$3.068s$
		100	5000	∞	1.0	$1.6547536595 \times 10^{-13}$	$16.894s$
				8959	1.0	$1.6547536595 \times 10^{-13}$	$30.174s$
10	96	15	500	∞	$9.7610094277 \times 10^{-01}$	$2.3899057230 \times 10^{-02}$	$0.361s$
				3722	$9.7610094277 \times 10^{-01}$	$2.3899057230 \times 10^{-02}$	$0.707s$
		150	500	∞	0.0	1.0	$3.790s$
		10	5000	∞	1.0	$3.5772469492 \times 10^{-22}$	$2.404s$
				8184	1.0	$3.5772469492 \times 10^{-22}$	$3.957s$
		100	5000	∞	1.0	$5.4838685880 \times 10^{-21}$	$23.625s$
				8227	1.0	$5.4838685880 \times 10^{-21}$	$38.643s$
20	186	15	500	∞	$9.4063094425 \times 10^{-01}$	$5.9369055746 \times 10^{-02}$	$0.466s$
				3691	$9.4063094425 \times 10^{-01}$	$5.9369055746 \times 10^{-02}$	$3.407s$
		150	500	∞	0.0	1.0	$4.898s$
		10	5000	∞	1.0	$8.8864498503 \times 10^{-22}$	$3.051s$
				8221	1.0	$8.8864498503 \times 10^{-22}$	$5.072s$
		100	5000	∞	1.0	$1.3622800965 \times 10^{-20}$	$30.375s$
				8196	1.0	$1.3622800965 \times 10^{-20}$	$49.583s$

Tab. 2: Numerical performance of Algorithm 1 and Algorithm 2 to compute the distribution of the pattern $w = (100100100)^j$ (ex: with $j = 2$, $w = 100100100100100100$) in a $(1,4)$-sequence starting with $X_1 = 0$ and with the following transition probabilities: $\pi_{0,0} = 0.4$, $\pi_{0,1} = 0.6$, $\pi_{1,0} = 1.0$ and $\pi_{1,1} = 0.0$. n_∞ is the largest value of n used in the infinite sum; a finite value corresponds to the position where numerical convergence is achieved in Algorithm 1 while an infinite one corresponds to exact result of Algorithm 2. Algorithm 1 failed to produce a result on two occasions ($r = 150$, $n = 500$ for $j = 10$ and $j = 20$).

x_1	x_2	q	$\mathbb{P}(N_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{Y_5} = 0)$	$\mathbb{P}(N_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{500} = 0)$
0	0	0.3	0.8484458330	0.0794428210
0	1	0.3	0.8498548514	0.0800356376
0	0	0.4	0.9745916345	0.4565899323
0	1	0.4	0.9750222448	0.4587174663
0	0	0.5	0.9983578453	0.8752575333
0	1	0.5	0.9984031350	0.8766482482

Tab. 3: Results of Algorithm 1 and Algorithm 4 for $r = 5$ with $n = 500$, $w = 100100100$, with an order 2 Markov model whose transition matrix is $\pi_{00,0} = 0.4$, $\pi_{00,1} = 0.6$, $\pi_{01,0} = 1.0$, $\pi_{01,1} = 0.0$, $\pi_{10,0} = q$, $\pi_{10,1} = (1-q)$, $\pi_{11,0} = 1.0$ and $\pi_{11,1} = 0.0$ where $q \in [0,1]$ and assuming that the Markov chain starts with $X_1 = x_1$ and $X_2 = x_2$.

For example let us consider the order $m = 2$ Markov chain whose transition probabilities are given by: $\pi_{00,0} = 0.4$, $\pi_{00,1} = 0.6$, $\pi_{01,0} = 1.0$, $\pi_{01,1} = 0.0$, $\pi_{10,0} = q$, $\pi_{10,1} = (1-q)$, $\pi_{11,0} = 1.0$ and $\pi_{11,1} = 0.0$ where $q \in [0,1]$. If $q = 0.4$, the order of the model is reduced to 1, and we get the same model as in the previous numerical example. We can see in Table 3 that in this case, the results are very close to those in Table 1. As a validation, one can get exactly the same result as in Table 1 with $q = 0.4$ by considering the starting distribution $\mu_2(00) = 0.4$ and $\mu_2(01) = 0.6$. However, we should emphasize that this result is not a weighted combination of rows 3 and 4 of Table 3 since the computed probabilities are conditional ones. If we now consider a higher value of q (like $q = 0.5$) we shall favor the occurrences of '100' thus resulting in higher probabilities. At the opposite, a lower value of q (like $q = 0.3$) will decrease accordingly our probabilities of interest.

3.2 Renewal occurrences

Up to now, we have studied the number N_i of overlapping occurrences of our pattern of interest (up to position i). It may also be useful to consider the number N'_i of renewal occurrences up to position i instead of N_i. We recall here that a renewal occurrence is only counted if it does not overlap a previously counted one. For instance, if $w = 100100100$, there is three overlapping occurrences of w in '01001001001001001' but only one renewal occurrence (the first one).

It is often a great deal of work to study N'_i rather than N_i, it is however possible here with only a small modification of our transition function δ. The basic idea is just that looking for a renewal occurrence of w after one of its occurrences is exactly the same as looking for an occurrence from the start. Hence, simply setting $\delta(w, 0) = 0$ and $\delta(w, 1) = 1$ ensures that not any unwanted occurrence will be counted.

Using the same model (with $X_1 = 0$) as in Section 2.4 we get

$$\mathbb{P}(N_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{Y_5} = 0) = 0.9748567886 \quad ; \quad \mathbb{P}(N'_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{Y'_5} = 0) = 0.9064203814$$
$$\mathbb{P}(N_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{500} = 0) = 0.4578925276 \quad ; \quad \mathbb{P}(N'_{500} \geqslant 5 \mid \bar{N}^{(1,4)}_{500} = 0) = 0.2996029919$$

for $w = 100100100$ and where N'_{500} and Y'_5 are the renewal version of N_{500} and Y_5. Unsurprisingly, it is more difficult to observe 5 renewal occurrences of w than to observe 5 overlapping ones.

3.3 Heterogeneous models

Like in Stefanov and Szpankowski (2007), we have supposed that our sequence model is homogeneous. However, it is interesting to observe that our results hold with heterogeneous models. The matrices P and Q must be replaced in all algorithms by their actual value at the current position.

For example, we can consider the following model: $X_1 = 0$ and for all $2 \leqslant i \leqslant n$ let

$$\mathbb{P}(X_i = 0|X_{i-1} = 0) = 0.4 \times \left(\frac{n-i}{n-2}\right) + 0.6 \times \left(\frac{i-2}{n-2}\right) \quad ; \quad \mathbb{P}(X_i = 0|X_{i-1} = 1) = 1.0$$
$$\mathbb{P}(X_i = 1|X_{i-1} = 0) = 0.6 \times \left(\frac{n-i}{n-2}\right) + 0.4 \times \left(\frac{i-2}{n-2}\right) \quad ; \quad \mathbb{P}(X_i = 1|X_{i-1} = 1) = 0.0$$

Applying Algorithm 4 for $w = 100100100$ using these matrices P and Q (which here depend on i) in lines 7 and 8 we get:

$$\mathbb{P}(N_{500} \geqslant 5|\bar{N}_{500}^{(1,4)} = 0) = 0.6646404828$$

which should be compared to 0.4578925276 the result using the homogeneous model of Section 2.4 (the higher probability obtained with the heterogeneous model is consistent with the increase of $\pi_{0,0}$ along the sequence). It is of course possible to do the same with Algorithm 1 but in this case, we need to define the transition matrix for all positions (and not like here, up to position $i = n$); this point is left to the reader.

3.4 More complex pattern of interest

Now we consider the occurrence of a set $\mathcal{W}$ of patterns of interest, rather than considering a single word w. In such case, one simply has to add the corresponding prefixes to the set $\mathcal{P}$ to get the Aho-Corasick DFA corresponding to the problem. However, one should note that this approach may not be optimal since a smaller DFA may exist. A more elegant solution could consist to build directly the optimal DFA in replacement to the Aho-Corasick one which is exactly what is proposed in Nuel (2007).

By using this approach, we understand the sequence $\widetilde{X}$ as a PMC allowing to count both the patterns of interest $w \in \mathcal{W}$ and the forbidden patterns $\bar{w} \in \bar{\mathcal{W}}$. Algorithm 1 and Algorithm 4 then remain exactly the same, only change the state space $\mathcal{P}$ and the transition matrices P, Q and $\bar{Q}$.

For example, let us consider the pattern $\mathcal{W} = 10^{(2-4)}10^{(2-4)}10^{(2-4)}$ which is the set of the following 27 words: '100100100', '1001001000', '10010010000', '1001000100', ..., '100001000010000'. The smallest DFA allowing to count both occurrences of $\mathcal{W}$ and '00000' has $L = 20$ states and $F = 5$ final states (which is far smaller than the number of states of the Aho-Corasick DFA). If N_{500} (resp. Y_5) is the number of occurrence (the position where ends the fifth) of $\mathcal{W}$ then we get:

$$\mathbb{P}(N_{500} \geqslant 5|\bar{N}_{Y_5}^{(1,4)} = 0) = 0.9997627166 \quad \text{and} \quad \mathbb{P}(N_{500} \geqslant 5|\bar{N}_{500}^{(1,4)} = 0) = 0.9822759719$$

which are unsurprisingly both larger probabilities than the corresponding ones when we only count occurrences of '100100100'.

3.5 Multistate trial sequences

Up to now, the results presented in this paper focus only binary trial sequences. However, it is important to point out that the results hold with sequences over any finite alphabet. Moreover, any constraints may be used as long as they may be expressed as a set of forbidden patterns.

For instance, let us consider a random i.i.d. sequence of length $n = 500$ over the DNA alphabet $\mathcal{A} = \{\mathtt{a}, \mathtt{c}, \mathtt{g}, \mathtt{t}\}$. Let $w = \mathtt{gacgac}$ be our pattern of interest and $\bar{\mathcal{W}} = \{\mathtt{ttga}, \mathtt{actt}\}$ be our set of forbidden patterns. If we denote by N_i (resp. $\bar{N}_i$) the number of occurrences of the pattern of interest (resp. of the forbidden patterns) up to position i, and if Y_r denote the ending position of the r^{th} occurrence of w then we get:

$$\mathbb{P}(N_{500} \geqslant 1|\bar{N}_{Y_1} = 0) = 0.9796349439 \quad \text{and} \quad \mathbb{P}(N_{500} \geqslant 1|\bar{N}_{500} = 0) = 0.1067240581.$$

with similar computational times as in the previous examples.

4 Conclusion

In this paper, we have proposed several methods based on Markov chain embedding techniques allowing us to study the distribution of a pattern of interest in random (d, k)-sequences. The case where the sequence is constrained up to Y_r has already been treated by Stefanov and Szpankowski (2007) using generating function but the case where the sequence is constrained up to n is a new result.

In both cases, we suggest efficient algorithms using basic linear algebra only (sums, sparse matrix$\times$vector products, etc.) which are both easy to understand and implement. As a validation, we have compared our numerical results to those of Stefanov and Szpankowski (2007), and they are exactly the same.

We also have demonstrated the flexibility and usefulness of our approach by providing several extensions (renewal occurrences, heterogeneous models, complex patterns of interest, etc.) which are here quite straightforward to obtain while some may be hard to get in the generating functions framework.

References

V. A. Aho and M. J. Corasick. Efficient string matching: an aid to bibliographic search. *Communication of the ACM*, 18(6):333–340, 1975.

D. L. Antzoulakos. Waiting times for patterns in a sequence of multistate trials. *J. Appl. Prob.*, 38:508–518, 2001.

S. Chadjiconstantinidis, D. L. Antzoulakos, and M. V. Koutras. Joint distribution of successes, failures and patterns in enumeration problems. *Adv. Appl. Prob.*, 32:866–884, 2000.

Y.-M. Chang. Distribution of waiting time until the rth occurrence of a compound pattern. *Statistics and Probability Letters*, 75(1):29–38, 2005.

J. C. Fu. Distribution theory of runs and patterns associated with a sequence of multi-state trials. *Statistica Sinica*, 6(4):957–974, 1996.

J. C. Fu and Y. M. Chang. On probability generating functions for waiting time distributions of compound patterns in a sequence of multistate trials. *J. Appl. Prob.*, 30:183–208, 2002.

M. Lothaire, editor. *Applied Combinatorics on Words*. Cambridge University Press, Cambridge, 2005.

B. Marcus, R. Roth, and P. Spiegel. *Handbook of Coding Theory*, chapter 20: constrained systems and coding for recording channels. Elvesier Science, Pless, V. S. and Huffman, W. C. edition, 1998.

G. Nuel. Pattern markov chains: optimal markov chain embedding through deterministic finite automata. *J. Appl. Prob.*, 45:226–243, 2007.

V. T. Stefanov and W. Szpankowski. Waiting Time Distributions for Pattern Occurrence in a Constrained Sequence. *Discrete Mathematics and Theoretical Computer Science*, 9(1):305–320, 2007.

E. Zehavi and J. Wolf. On runlength codes. *Transactions on Information Theory*, 34:45–54, 1988.